INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI®

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

NORTHWESTERN UNIVERSITY

New Product Development Processes: Creation of a Dynamic Analysis Tool

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Industrial Engineering/Management Sciences

by

David Michael Redszus

EVANSTON, ILLINOIS

June 1995

VOLUME I

ABSTRACT

NEW PRODUCT DEVELOPMENT PROCESSES:
CREATION OF A DYNAMIC ANALYSIS TOOL

David M. Redszus

This study is an examination of processes by which organizations actually conduct New Product Development (NPD). Processes were found to exhibit non-linear characteristics, which trigger substantial delays, higher costs, deteriorating quality, and ultimately, lost market opportunity.

With the assistance of NPD participants, we documented their processes in over 100 US and German industrial organizations, and the US. Army Materiel Command. IDEF0 functional modeling was employed at several sites. The functional models (and process flow charts derived from these models) did not reveal dynamic behavior, but demonstrated that NPD processes are composed of an enormous number of functions (i.e., *complicated*) whose relationship are not well understood, even by most participants. Further, these processes were found to be highly iterative (i.e., *non-linear*). This conflicts with common linear process views. Current NPD management tools largely assume process linearity.

When considered dynamically, NPD could be characterized as a ***complex system***, sharing characteristics with certain deterministically chaotic systems. Further, NPD was identified as a collection of ***information churning processes*** which often stifle engineering progress.

A new dynamic analysis tool, the ***Complex Process Path (CPP),*** was developed from existing manufacturing simulation techniques. It provides for high, previously inconceivable, levels of non-linear information processing. It also incorporates varying

degrees of functional concurrence, behavioral contingency of human resources, priority policies, and variable service times. Though prototype flow was sequential in the current model, prototype iteration can also be incorporated.

Fifty-two variations of a simple CPP model were simulated, using parameters derived from field observations. Incremental changes in functional efficiency or structure of the information system could produce unpredictable, non-intuitive behaviors and significantly change overall performance. ***Further, it often resembles behavior seen in the field studies.***

To help rectify problems of unpredictable complex behavior, a three component framework has been established. Based on field studies and CPP model dynamics, several managerial suggestions have been forwarded

A deficiency of dynamic management tools and measures precludes immediate exploitation of non-linear behavior *to improve* system performance. Research focused on dynamic management techniques is warranted and is expected to be useful beyond the domain of NPD in other complicated/complex organizational systems.

ACKNOWLEDGMENTS

Where do I start? As a tumultuous process, this research has been a roller coaster of "fits and starts," largely resembling the processes under analysis. I can, in retrospect, take blame for process slowdowns. Without the help of others, however, there may not have been any velocity nor acceleration. So many people have influenced this research that I cannot list them all. A few have had outstanding positive influence, and I wish to acknowledge them here.

Many parents are proud of their children. I am proud of my parents. Dad, you have been the most influential and caring person throughout my personal development. Sometimes I wonder where I might have been without your insight and advice. From your endless reading of my manuscripts to continuous inquiry over how to increase the value of my research to the real world, you have always been there. I'm sure you couldn't possibly enjoy so many late night crises, but you never complained. Mom, you showed me that it is possible to remain cheerful, no matter the situation. I'm still learning this process; you've mastered it. Also, I don't know how you've managed to stay 38 for the past four years! Thank you both for your tireless support.

Daniel, my brother, you hold a special place in my heart. As a critical reviewer of that original monstrosity I had called my thesis, as a fierce racquetball rival, and (mostly) as a close friend, you are the greatest. Our blitzkrieg run through Germany put much in focus, albeit with a little amber (Weiss) tint. By the way, how did you *really* get to stay over there, anyway? Diane, as you've grown through your college years, I see myself all over again. Though you might not have realized it, you've helped me realize that youth can be eternal, despite Mother Nature's might. Your entertaining personality has been a breath of fresh air for me during this process. Everyone should have a sister like you.

The commitment of the committee members for both my qualifying and final examination has been impressive. Thank you, Drs. Frey, Rath, and Cassell. I had not anticipated your level of involvement in such roles. I have difficulty imagining a better,

more diverse committee. You have all taught me well, in your own ways. Perhaps your most salient lesson: *nothing* is quite the way it seems.

Much thanks go to Charles Thompson, Al Rubenstein, Barry O'Neill, and Wally Hopp from the IE/MS department for their guidance in formulating a research agenda and outlining how "research really works," before I even entered graduate school. Allen Batteau, as a friend and professional colleague, you offered me excellent advice and examples early in the process, which I shall continue to use for a long time.

To Dennis Wisnosky, thank you and the Wizdom crew for supporting my research. Your ability and willingness to engage in this experiment is truly rare in business today. I think we have all learned a great deal, and hope that you can put it to good use. I've had fun working with the "wizards." The EPIC, NGEDM (AMC), TAPOI, and other projects would never have existed without your work. Many others should also be thankful to you.

I am indebted to the variety of personnel at each of the companies I visited and interviewed. Their candor was both surprising and appreciated. Without them, this study would likely have followed a more traditional trajectory, and concluded the same old thoughts. These people really made me address the whole subject in a different light.

To the Beckmeister, now that I'm done with this chapter, you owe me a pizza (or is it the other way around?)! Thanks for the 24 years of friendship. Over the years, several others have helped me get through this seemingly eternal study. My hat is off to Bill and Ellen B., Boris A., Craig W., Dan W., Mike S., Joe F., Karen D., Kevin G., Gary S., Jeff L., Les W., Owen F., Paula H., Steve M., Tom G., and Virginia R. for their emotional and intellectual support.

TABLE OF CONTENTS

LIST OF EXHIBITS

LIST OF TABLES

PREFACE

Describing one's research is never a straightforward process, since so many iterative and interactive deliberations are made during the research process. It is not unusual to lose sight of the unfamiliarity of the research to others, and hurriedly delve into complex, difficult-to-understand details. This is a technique that I have personally practiced far too many times. Such practice notoriously impairs communication. As more "fragments" of the research are relayed, the overall "picture" becomes more clouded, rather than more clear. One could describe this phenomenon as a decrease in the "signal to noise ratio" as cumulative communication increases. ***Noise*** in this instance can be characterized as excessive detail in communication. Using terms introduced in our work, ***interface complexity*** rises faster than true ***information transfer capability***.

To alleviate this problem, this thesis is presented in two parts. A "short report" of the thesis is presented in the remainder of "Volume I". It is a condensed overview of the research, and attempts to highlight the most important findings and efforts, which should be of interest to practical development executives. We deliberately avoided the normally prevalent use of equations and academic vernacular in this overview. Yet, we have tried to retain the rigor and content of the research. This "short report" references many detailed findings, descriptions, methodologies, modeling results, and other conclusions. More information on these can be found in "Volume II" of this report, ***Appendices***, which are presented for researchers who are interested in furthering this field of study and for executives and managers who are interested in more detail.

CHAPTER I: INTRODUCTION

"Discovery consists of seeing what everybody has seen and thinking what nobody has thought."

- Albert Szeat-Gyorgyi[1], 1937 Nobel Laureate in Medicine

Many years ago, I became interested in how new products were developed. I made it my mission to find out. I visited many companies throughout my childhood, teens, and college years. As a member of an entrepreneurial-intense family, dinner table conversations touched on all facets of business. Through my college years, I engaged in the development of several industrial and consumer products, from voltage regulators to re-design and fabrication of production-based racing cars. I was not interested in just learning about a wide array of topics, but in learning as much as possible about each one. Interdisciplinary skills were a prerequisite for an objective mind, I had been told. After completing my undergraduate Industrial Engineering curriculum in three years, I spent the better part of my senior year obtaining another major (Economics) and taking graduate courses at Northwestern's Kellogg business school. I was looking for the holy grail of innovation and development. I soon discovered it was not to be found there.

I went to graduate school and worked under Dr. Donald Frey, a professor who has more practical experience developing new products than perhaps any other academic worldwide. I visited and examined more companies, as part of my schooling, business, and preliminary research. As a teaching assistant to Northwestern University's MEM professional degree program, I spoke with many of my students about the development

---

[1] As relayed by Royston M. Roberts in Serendipity, New York:Wiley, 1989, p.245.

conditions within their companies. Yet, there was still a significant piece of the product development puzzle missing. It was like that classic cartoon that shows nothing but chaos in a process and then...poof!...out jumps a new product. Though stories, recollections, academic theories, and assertions abounded, nobody could ***explain*** this. At least not to my satisfaction.

I had to go find out for myself. I studied a very large automotive engineering center at a very well-known company which has been known for creating hundreds of products over most of this century. I stayed there for over nine months, probing around, asking questions, watching, asking more questions, making statements and watching the nature of the responses, asking more questions. I talked to engineers. I talked to managers. I talked to secretaries. I talked to accountants. I talked to artistic (design) types. I talked to technical types. I talked to executives, planners, production-line workers, union workers, non-union workers. I talked to almost 300 people within the company (and to several of its development suppliers) about how the product development process works. Throughout all of this, I came to believe that there were many different impressions of how their process worked, and that their impressions did not inter-link very well.

As part of this investigative process, I worked daily with a team of twenty experienced engineers in this company to help document the process. We, as a team, spent over one million dollars examining the process. We created an IDEF0 functional model so large that it took over 700 pages[2]. When we strung the model out into a process, it filled three

[2] Actually, our use of IDEF0 to model the engineering process was a new, unique use of the methodology, for it had been used virtually exclusively for manufacturing processes in the past. For a brief overview of IDEF0, refer to Appendix D.

walls of our 1500+ sqft. war-room. When we did this, we felt as if we understood the company ***less*** than we did before. At least, our perspectives had changed. Many experienced engineers got a big eye-opener. Some of their comments of surprise over this period would make a curious toddler seem like a cool, calm, collected, and experienced professor. When they tried to communicate their findings to their managers, however, many mangers refused to accept these "controversial views." They had lived through the process themselves, damnit, and nobody was going to tell them that it was different than their past experiences and current perceptions.

Managers and engineers in the know, however, were pleasantly surprised and intrigued. This new, unorthodox view incorporated important aspects and explanations of the development process, which had not been seen before. Many studies of their processes had been conducted before. None had shown this amount of clarity. When the chief of engineering, an executive VP of the company, saw the potential of this type of modeling for his own use, he quickly adopted it as a standard. Yet, he realized the political problems of middle managers' insecurities. Better, he felt, for the project to be broken up into many more, albeit smaller, projects. Each of these projects could help local managers overcome their mental-paradigmatic hurdles, without high public exposure and embarrassment. Further, several smaller models would be easier to "analyze."

Wearing my "researcher's hat", I realized that a reductionist attitude was still prevalent. Simultaneous, continuous, incremental, local improvements would not necessarily help solve the *overall* aches of the engineering system of development. They would temporarily help managers accept and resolve their local problems of process mis-understanding. Being interested in large-scale system dynamics, I felt it was important to

retain a more holistic approach to product development improvement. Seeing this trend, I helped bring another person up to speed to lead this project. I would have to go find another organization that was willing to look at the problem from a more holistic view.

Within three months, I found myself coordinating an existing project team with an intimidating and important mission: examine and evaluate the operations of the world's largest product development organization, the US Army Materiel Command (AMC). Our focus was on the dynamic development, distribution, storage, retrieval, and modification of engineering data. Our mission was to identify current practices, potential technologies, and develop a practical vision for future Engineering Data Management (EDM) processes. First-hand process examination extended from internal Army practices at six Major Subordinate Commands (MSC's) to the practices within several major Army contractors (visits to Boeing (Philadelphia), Computervision (Bedford, MA), FMC (Santa Clara, CA), General Dynamics (Warren, MI), General Electric (Schenectady, NJ), McDonnell Douglas (Mesa, AZ), Martin Marietta (Orlando, FL), Raytheon (Andover, MA), United Technologies (San Jose, CA)). Major IDEF0 models were developed for the Army sites, while extensive interviews were performed at all sites. We developed and administered two surveys: one for Army sites and one for contractor sites. This major study took place over the course of two years. Though much larger in scope than the earlier project, this project was completed at little more than half the expense. As an investigator, I had progressed along a methodological learning curve, but also had considerably more freedom than before. Since the problem was actually less structured, we largely let the findings lead us.

Over the course of this project, we documented a variety of engineering data development, storage, retrieval, and change systems. Many of these were shown to conflict with one another, as each sought improvements with local objectives in mind. We observed and documented many connectivity and automation-related problems, problems of both inadequate and obsolete engineering data standards, and problems with developing stable, non-restrictive design requirements. We observed massive amounts of decentralized data duplication which, even using current technologies, is a prerequisite for lackluster configuration management practices. As a further result of this problem, there exist a plethora of problems with data completeness, consistency, inaccuracy, and obsolescence. EDM processes themselves exhibited non-concurrency, proprietary-based lack of cooperation, poor technical integration, some technical delays, and massive administrative-based delays.

We integrated our research efforts with research teams in several other areas, including logistics, production engineering, requirements definition, and standards development. Other teams came from within all branches of the armed forces, industry, and the academic sectors. As a result, the recommendations forwarded in this project were in alignment with emerging technologies and practices. This came to a head when our study was reviewed by the Commanding General of AMC and his staff in Washington, DC. Many of our recommendations are being further developed for incorporation into the next generation of engineering data management throughout the armed forces.

As an outgrowth of that project, I was asked to help lead a project to examine the processes of the personnel management functions within the Army and National Guard. Again utilizing a team approach, we enlisted the assistance of active and reserve officers

(1 Colonel, 5 Lieutenant Colonels, 1 Captain), and several Sergeant Majors from within the Army and the National Guard. While this was off the "research path" of product development per se, this project offered an opportunity to refine the modeling methodology of service-related concepts in IDEF0, as well as develop various performance measures using IDEF0 modeling methods. Specifically, the first intensive use of Activity-based cost accounting via IDEF0 was conducted during this project.

Perhaps more importantly, however, we employed another new concept in IDEF0 modeling of real organizations, which has yet to be applied to industrial applications: functional comparison of an organization in "Wartime" versus "Peacetime" modes. Because this project was conducted shortly after the Gulf War, fresh data and experiences from both modes were readily available. In essence, we developed two markedly different models for each respective organizational mode. Remarkably, the functionality (i.e., *effectiveness*) of the wartime organization was better than the peacetime organization, while the peacetime organization was presumed to be more *efficient* from a cost and "stability" perspective. The basic difference between these organizations? In one, administrative functions were cut to the bone; for the other, administration thived.

The team concluded that organizations could likely remain in wartime mode in peacetime conditions, with few adverse effects. Administrative types would wail their disapproval, however, because there was an implicit loss of administrative *control* in such a mode. For political reasons, such a recommendation could not be accepted. As a researcher, I continued to learn about the problems of separating technical solutions from political solutions. It was ever more apparent that people's feelings can be more important in the management of organizations than technical solutions. Also, for the first time in my life, I

had the opportunity to formally present findings to a Major General and two Brigadier Generals. It didn't take long for them to realize that we, as a team, had learned more about PERSCOM (Army Personnel Command), ARPERCEN (Army Reserve Personnel Center), and GUARDPERCEN's (National Guard Bureau Personnel Center) functional operations in a few months than most *ever* learn, regardless of time.

With the experiences of these large-scale projects in mind, I wondered whether the problems I had observed were limited to these organizations only. Since the consulting firm I was working with was mostly interested in large accounts (given their product, there was good reason for this), I sought to find out more about product development activities at other venues. Much of this was done while simultaneously working on the large projects mentioned above. Over the course of four years, I visited over three dozen product development organizations throughout the US and within Germany. While I did not spend as much time at each of these organizations, I observed many similarities between these product development organizations and their larger brethren.

Some of the additional sites visited included large engineering centers (BMW FIZ, Ford Body Engineering, GM Advanced Vehicle Engineering, Mercedes-Benz, Porsche, Boeing), research facilities (AMOCO, BMW Motorsport, General Motors Research Laboratories, MAHLE, Mercedes-Benz Sport-Technik, Motorola, NALCO, National Academy of Sciences), skunkworks (AMG, M1 Toolworks, McKee Engineering), software development organizations (Computervision, EDS, PRC, Wizdom), the crash test center at a major US automaker (Ford), product preparation & distribution facilities (BMW, M-B, Penray, Robert Bosch, Jorgensen Steel), parts distribution centers (Mercedes, Bosch), and production plants (BMW, Mercedes, Porsche, Boeing

Commercial Aircraft, MAHLE, Penray). In addition, I spoke with many customers, potential customers, salespersons, and former employees of many of these companies. Some major development organizations with which I spoke, but did not visit first hand included AT&T, CACI, CERC, CSC, OSD CALS, Hewlett-Packard, GM-MTG (Motorsports Technology Group), Honda Motorsport, IBM, Intergraph, JCALS, Matsushita, Microsoft, MathWorks, and Siemens. In consideration of the various international trade shows I've attended, there are easily several dozen more.

Though I observed a variety of cultures at these sites, I found that there were many process similarities between them. Even the smallest companies had recurring, albeit ever changing, problems of information processing dominating over engineering functions. It seemed as if an underlying, compelling force (driver?) caused people at these sites to subordinate their engineering tasks to administrative tasks. Though I had personally felt that this might derive from a human tendency to wish for reduced task ambiguity (i.e., "document things so we, as a group, better understand things"), this pet theory was not confirmed. It seems that people do not substantially *delay themselves* with administrative burdens, but rather *are delayed by* administrative tasks that have been forced upon them. When in this mode, people are in a response mode, rather than a pro-active mode. In some cases, it seems that certain people relinquish control over the system and let the system control them.

My early academic and professional experiences had not adequately prepared me for such observations. After observing a multitude of organizations and their behaviors, I found it difficult to verbally describe their behaviors. Managers and engineers who live with these dynamic processes often feel that there is no systematic order to their local "fire-fighting"

operations. Seeing the processes from a functional viewpoint, however, there *was* an underlying order. To better communicate this view, I felt that others need to ***see***, not just read, what I had seen. There was a problem with this, however. Activity architectures which I had seen and helped develop were both proprietary and extremely large. The proprietary restriction is self-explanatory, and easily fixed. I could merely "genericize" existing models. The size restriction, however, was one which I would have to work around, for demonstration purposes.

Thus, I embarked on the development of a demonstration device which could represent some of the ***types*** of dynamic behavior seen in these organizations. By creating a simple, yet easy to understand model of a development process, perhaps I could also learn about the *causes* of some of the observed behaviors in larger, more complicated, real systems. Though I did not wish to create controversy in developing such a model, I was struck by the lack of sophistication that existing managements were using to control their processes. Maybe the demonstration device we created could help management better understand and perhaps, eventually, control their real processes.

First, I reviewed existing demonstration and analytical techniques. Perhaps an existing modeling methodology could be applied with parameters which I supplied. Among the first methods I reviewed were PERT networks. These looked appealing because of the pseudo-simultaneous nature of many of the observed processes. PERT is well understood by many systems analysts and trained management. However, our observation of looping behavior was incompatible with PERT. I considered an automated process flow technique (with subsequent simulation) which could be utilized from existing IDEF0 functional models. This seemed (and still seems) very promising, but for the fact that they are

oriented around linear views of processes and were still embryonic in their development. I considered a variety of Markov processes. These accommodated looping processes, but were not conducive to parallel activity processing. After discussing this with several professors, I considered a state-augmented Markov process. Unfortunately, the number of transient states necessary to process as the system grew would be frighteningly large. Besides, probability structures do not provide a decent view of people's decision-making. I had found that people throughout the visited development organizations were very rational and thoughtful about their operations. Their main problem was that they were asked to respond to the requests of "the system", and fulfilling such requests could hurt their productivity. They had implicit rules for their behavior, with task selection priorities and policies for information generation and dissemination. I needed contingency-based, response-oriented, non-linear structure that had the ability to incorporate varying internal efficiencies and external influences. I was surprised when I could not find a suitable structure among existing work.

So I created one. Using an existing simulation program (SIMFACTORY, a shell program for SIMSCRIPT), I developed a simple model of information interchange. I treated both information and prototype assemblies as forms of work-in-process. Self-created rules (via priority structures) would dictate when stations would switch from information to prototype. Originally developed as a manufacturing-based model, SIMFACTORY had not conceived of information flow in multiple directions (have you ever pushed WIP backwards through *your* manufacturing process?). After discussing this at length with CACI system developers, we decided to consider circular processes as re-work processes. I essentially had developed a manufacturing-based simulation model with extraordinarily high re-work rates. The CACI technicians thought the system might crash. I tried it.

Sometimes it did crash. Within other parametric regimes, however, it did not crash. In fact, it worked extremely well. It worked scarily well. I was only slightly less than shocked. I named this new structure the ***Complex Process Path (CPP)*** structure.

I ran a variety of "crazy" runs, to probe the limits of the CPP structure. I learned about the influences (and non-influences) of parameters on the behavior of this system. Hey, this research stuff wasn't all work--this could be fun! After a few dozen different investigations, I was happy with the basic elements of this system. It was not complete, but seemed to carry enough realism to get the proverbial ball rolling. I developed a specific model which developers could visualize as a highly simplified view of their development process. It had four basic departments, sequentially linked with a path for an ever evolving design. Between every department, I incorporated information channels. In each department, there existed four stations, where specific activities were to be conducted: three for *information* processing, one for *prototype* processing. To conduct the work at these stations, I incorporated human resources which, naturally, were in short supply. I could change the number, ability, and teamwork needs of each human resource at each department. Each department could have different characteristics in this regard; I tried and observed a wide variety of these characteristics.

Because the model was a computer-based simulation, automatic data collection was conducted as the dynamic model was "run." Real-time visual observation was also possible. Though I recorded extensive statistics on 52 official runs with varying parameters, the most educational part of this model was visual observation. In roughly 30 minutes, I could watch ten years of product development activities take place. I could watch huge inventories of information gather at a department's "front door", while other

departments were stagnant, waiting for that department to finish. I could see activity progress through the model in waves, as the prototype materials move through the system. I could see highly erratic release intervals from one prototype to another, for a given set of constant parameters. I watched human resources switch from information processing to prototype processing back to information processing over and over again. I was watching the "fire-fighting" activities that engineers go through in the real world. All with a simple, four department model. It was an overall view that most executives only wish they could have.

Often, this model behaved strangely. Increase the efficiency of one station and the system's behavior improves. Increase it some more and the system's behavior deteriorates. Wow! Just because I'm improving local efficiencies, I'm not helping overall effectiveness! What if I improve all the efficiencies by the same amount? Then the overall system should improve, right? Not necessarily! Transient tidal waves of information can exist which, if not countered by extraordinary processing improvements in other departments, can cause the entire system to halt its prototype activities; the system turns into an administrative (information) processing system. And who's information are the participants processing? Their own! In some extreme cases, the entire system shuts down both information processing and prototype development activities. This occurs when each department, or just one station within each department, is simultaneously waiting for an input from the other. They have successfully caught their own tail, swallowed it, and died. From a research perspective, this model isn't going to look good. If this model is reflective of reality, it makes managers and engineers look like uncoordinated drones in a beehive (actually, mother nature seems to enforce much better coordination).

Let's see.

- Recirculating information channels? Real systems have those. Check.
- Localized efficiency objectives among management? Yes, we saw that. Check.
- Low engineering activity time among engineers? Unfortunately that happens. Check.
- High proportions of administrative processing? Check.
- Extensive needs for better information management? Check.
- Limited human resources? Check.
- Overworked human resources? Check.
- Long development time (and cost)? Check.
- Irregular product release times? Check.
- High degree of waiting for others? Check.
- Huge backlogs of work to do? Check.

Gee. I'm not going to be very popular at all with this model. But it seems to be more reflective of reality than any other development models I've witnessed or developed. It has hints of IDEF, but is dynamic. It's not perfect. It's not complete. But it is relevant. It shows some basic real-life principles. The work has been objective (painfully so, just ask me!). What's more, I have a bunch of ideas of how to make this model much better, more vivid, but it will be even more painful to management. How do I present this to development managers without creating sweat on their brows and redness in their eyes? Maybe objective executives can comprehend the nuances, subtleties, and underlying truths of the development process. Maybe they know this already. If so, it may become our little (big?) secret.

Based upon both the field research and the CPP model, I do have some basic (nice) suggestions for developers and their management. We shall discuss these in our final chapter of this work. If your impatience level is high, then turn to page 239. Be prepared to see some surprises, however!

An over-riding theme that should be derived from this study is that we desperately need *better analysis tools* for managing our development processes. As technologies evolve and product life cycles continue to shorten, developers and their managers everywhere will feel more, not less, pressure to perform. Without the correct tools to do so, however, we will continue to manage by intuition and experience. Intuition and experience can be very useful. For managing such complex processes as new product development, we need something more. If we do not find it, then we may be bound to the whims of those that do. This is not merely a corporate priority; it is a National Priority.

There is much more research to do in this field. I see formal chaos and anti-chaos analyses. Fractal representations of development processes. Continuously dynamic CPP structures. Self-organizing system algorithms. Semi-automated CPP analysis systems. Virtual reality simulations within CPP models of organizations. There is the fundamental problem of expanding the CPP structure to larger scales. We've done a lot of work on this, so far. Yet, we are beginning to see that there is much more work to be done, and much more understanding to be had. We need to start helping to improve the processes out there, not merely demonstrating how messed up they are. After many years of warm-up....I guess we're really just getting started!

CHAPTER II: REVIEW OF THE LITERATURE

"Knowledge is the small part of ignorance that we arrange and classify."

--Ambrose Bierce

"Perplexity is the beginning of knowledge."

--Kahlil Gibran

"Experience may be defined as the acquired capability to recognize when you've made the same mistake again."

--Anonymous

2.1. Background

An acute finding of this research is that new product development can be a highly confusing, complicated array of tasks. At times, "local" development activities appear to pursue contradictory objectives. Some activities are redundant, performed without knowledge of their "activity clones." The departmentalization of engineering personnel does not seem to help. Rather than becoming more worldly and outward-looking, many development personnel are christened as "specialists"; they often have less integrative, more inward (local) focus than their historical counterparts. Without a doubt, such individuals carry more ***depth*** of technical understanding. Yet, and perhaps as a direct result, the number (i.e., the ***breadth***) of disciplines which are involved in new product development today dwarf the scope of engineering curricula in our universities. As many educational institutions (specifically, the curricula within departments) are patterned after their own specific (and often abstract) research orientations, our engineers of the future seem to become even more specialized and isolated. The implication should be clear: ***We***

are trying to develop more complicated, more interdisciplinary products with less integrative, less-relevant training than before.[3]

Perhaps, one argument goes, we are doing well enough with our specialization strategy. After all, much of the unprecedented growth of our nation was built on the efficiencies of manufacturing specialization. Such specialization was a gateway for the advent of mass production (which was well matched for an exploding population base). Further, how could we have developed such astounding technologies as solid-state electronics, organic chemistry, or even harnessed nuclear power without technical specialization?

Naturally, the acceleration of technology would be difficult to imagine without some degree of specialization. If one considers the mental capacity of the human brain (more specifically, the ***learning*** capacity of the brain), it is clear that there are limits to one's global, integrative knowledge. None of us possess complete knowledge bases, even as rudimentary as can be found in a grammar-school level encyclopedia. Rather, we find ourselves orienting about a limited, comfortable set of disciplines and probing deeper and deeper into but a handful of these disciplines.

---

[3] Lest one think that these observations and judgements are a bit extreme to be presented in a doctoral dissertation, much the same conclusion has been forwarded by others including, but not limited to, the MIT Commission on Industrial Productivity (see Dertouzos et al, 1989), the Stanford Design Forum (see Stanford, 1989), and the National Research Council (see NRC, 1991).

As many participants in technologically-intense projects can attest[4], progress is not solely (and sometimes not significantly) dependent on technological depth. Most ***applied*** technology is, indeed, far less sophisticated than the cutting edge of knowledge in a given discipline. Rather, significant efforts are made to interface existing (common) technologies and communicating those capabilities to non-technologists. ***Integration of capabilities*** has been observed as a key success factor, not the level of technology, per se. If this is true for technology-intense programs/projects, it seems reasonable to assert that the contribution of technological depth to other programs is even less significant.

This does not diminish the importance of ever-advancing technology. Rather, it only emphasizes the need for adequately harnessing the technologies that we have developed. Until we can incorporate and integrate new technology into applicable, beneficial products, such new technology is of little practical use. This study has been oriented around this specific concern.

It is a conclusion of this research that the act of balancing specialization and holism is a factor in the success or failure of innovation and new product development. This perspective suggests that detailed expertise (which arises from specialization) ***as well as*** the integrative skills (which are commonly associated with general management) are both necessary. At risk of being too colloquial, we submit that successful innovation depends upon "seeing the forest, as well as each tree."

---

[4] Based upon conversations with several members of U.S. Space programs since the early 1960's, as well as individuals involved in current defense systems projects and technology-intense industrial development projects, including aircraft design, manufacturing automation systems, and automotive research and development facilities.

With this conclusive perspective in mind, we consider the work of some past research in the remainder of this chapter. We deliberately do not delve into all of the detailed research findings which have been reviewed. This decision was made on the basis that, as an integrative study, many detailed findings proved to be both esoteric and contradictory (because of the variety of local, limited, and/or controlled conditions under which they were developed).

The purpose of these reviews was not limited to merely reviewing past studies of innovation and product development. Because of the normative nature of this study, first-hand observations of development activities and decisions required some interpretation. Throughout the course of the field studies, an uncountable number of reasons and conditions for particular activities and decisions were observed. Each unveiled a unique set of disciplines which could potentially be used for analysis.

As outlined in Appendix A, the "Tree of Knowledge" is presupposed to flourish best through translocation, or integration of previously unconnected disciplines. Thus, this study was not developed by encircling a previously determined "branch" of the academically structured knowledge tree and gradually focusing on a single pre-established discipline. Rather, this study examined highly complex *and* complicated[5] phenomena and delved into specific disciplines and discipline-specific methods which

[5] Despite their similarities according to Webster, we distinguish between the terms ***complex*** and ***complicated***. Complex, as we use the term (and in accordance with modern non-linear dynamics venacular), describes a non-linear, high-feedback condition. We use the term complicated to describe a system with many components, whether complex or not. We shall address this issue more in Chapter V.

most vividly impacted or described this phenomena. This discipline selection methodology sent us on a trek to many different sectors of the knowledge tree. The chronology of this trek closely paralleled our observations of local decision-making, physical activities, and processes at each field site.

Given that we have traversed to so many areas of the knowledge tree during this study, it is useful to consider some key areas which we have found particularly insightful in examining new product development. Currently, let us consider the following areas:

- Innovation and Product Development
- The Cognitive Sciences
- Mathematics
- The Management of Science

2.2. Innovation and Product Development

A significant amount of past research has surrounded the topics of innovation and new product development. For reasons which should become evident, the term "surrounded" is used here because past research has not quite captured the essence of ***how*** the activities of innovation and new product development actually proceed. Rather, it has reiterated the ***need for,*** the ***effects of***, and various ***prescriptions for improving*** these two areas. First, we will consider some of the *innovation* literature. Then, we will look at some past discussions on the field of *product development*.

2.2.1. Innovation Research

As alluded to in the introductory chapter of this work, the landmark discussion of innovation may be considered to have been forwarded by economist Joseph Schumpeter (Schumpeter, 1927, 1928, 1934, 1939, 1962). In these discussions on the state and stability of capitalism, innovation is considered a non-continuous process which not only enables, but actually ***drives*** the creation of credit in an economy. In this view, innovation may have nothing to do with the creation of a physical product, per se, but rather is the successful act of "putting an untried method into practice" in an effort to maximize profit. Further, innovation was not considered to be a battle of intellect, but rather a battle of will, which he coined as a special case of social leadership.[6] Though the Schumpeterian business cycle was considered a direct consequence of innovation, and innovation was considered a necessity for any surviving economy, no mechanistic theory for innovation itself was ever forwarded. He adamantly asserted, however, that invention and innovation were distinct economic and social processes, and should be analyzed as separate processes.

An appealing taxonomy for defining "innovation" has been forwarded by Frey[7] . In his work, innovation is broken into two major classifications: ***seminal*** innovation and ***incremental*** innovation. Seminal innovations tend to be associated with landmark breakthroughs in technology, science, or a whole new direction of thought (particularly

---

[6] J. Schumpeter, "The Instability of Capitalism," Economic Journal, 1928. (as reprinted in Rosenberg, N., The Economics of Technological Change, Middlesex (UK):Penquin Books, 1971, pp 13-42.)

[7] Numerous discussion with Donald Frey at Northwestern University in the period 1988-1994.

for non-physical innovations). Incremental innovations tend toward the continuous improvement of existing products, albeit with some "new" twist.

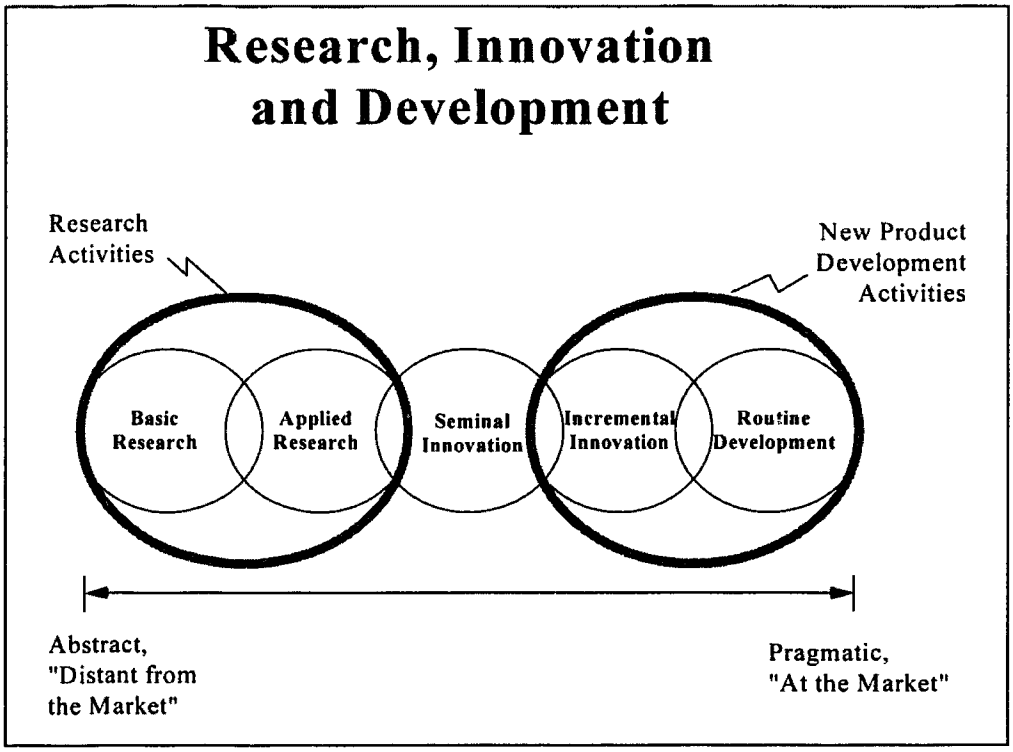
One possible reaction to this classification is that seminal innovations are "larger" in some sense than incremental innovations. Yet, the examples which have been introduced in this taxonomy indicate that seminal innovations are much more rare, and somewhat distant from the marketplace. Incremental innovation, on the other hand, may utilize technology derived from one or more seminal innovations to improve a number of existing products, or cascade new products, using such existent technologies. As such, incremental innovation looms larger in frequency; seminal innovation points to a more global direction over the long haul.

A similar, though more narrow taxonomy has also been proposed (Johne, 1985). In this case, innovation is broken into two categories: ***radical*** and ***incremental***. Radical product innovation is that which applies advances in technology to develop a new line of products. Incremental product innovation uses established technology to extend or improve a current product line. Using such a "close-to-market" definition of innovation, it may be forwarded that Johne is not referring to innovation, per se, but rather "incremental development." This naturally begs the question of where and how to draw the lines between research and seminal innovation, as well as between incremental innovation and routine development.

As demonstrated in Exhibit II.1., overlap can exist between the activities of research, innovation, and development. It may be conceivable to think of innovation (both "seminal/radical" and "incremental") as an activity bridge between research and

development. This graphic might be thought of as a two-way sliding scale, not a sequential time-based process. Naturally, very little basic research ever proceeds beyond that characterization. In fact, it may be convenient to consider that some abstract basic research concepts owe their very existence to the observation of fairly routine, highly pragmatic activities.

EXHIBIT II.1.



With this visualization, Johne's radical innovation can be considered larger steps (leaps!) between applied research and practical application. To this effect, new product development can be defined as ***the coordination and execution of activities which are***

necessary to create a completely new product. This includes elements of seminal innovation, incremental innovation, *and* routine development.

It is useful to consider another important distinction between two broad classes of innovation. As has been illustrated by Johne (1985), these are ***product innovation*** and ***process innovation***. Product innovation is the act of providing customers with either radically new or incrementally improved products[8]. Process innovation is the act of radically or incrementally changing internal organizational practices, with the intent of improving organizational performance. Though both product and process innovation carry beneficial roles, Johne asserted that product innovation has more long-run importance, because of its immediacy with the primary driver of the organization's revenue--the customer. As the marketplace becomes more "turbulent," it is even more important that product innovation be carried out successfully.

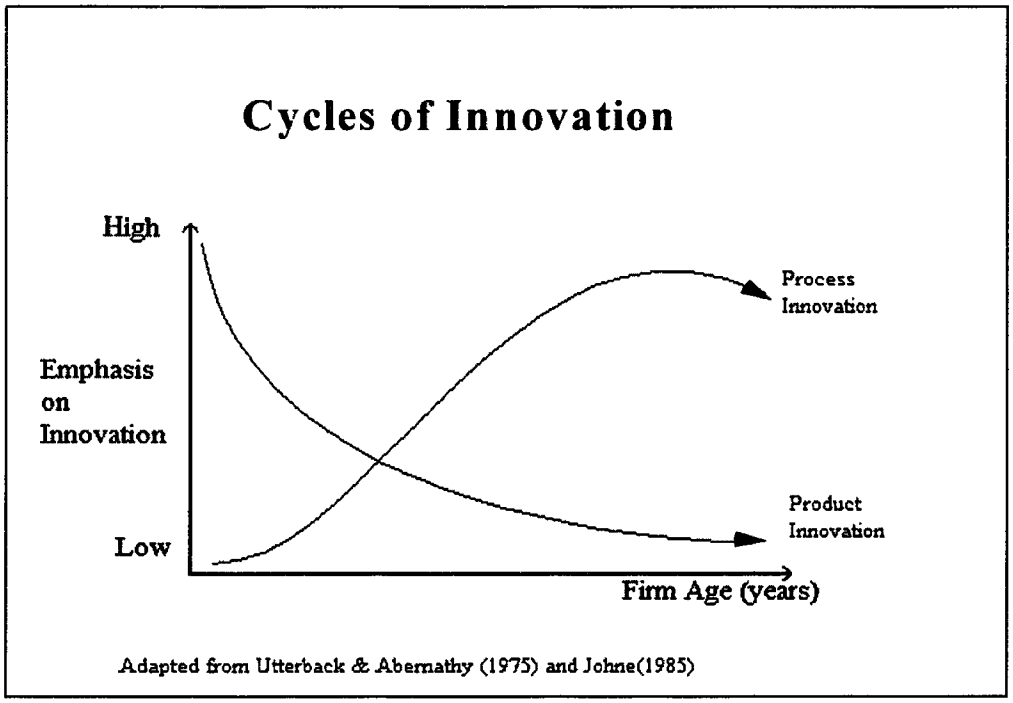
The degree of effort and expense to be allocated to each of these types of innovation is still an unresolved dilemma. The fundamental kernel of this issue is how far into the future an organization must (or can) plan, to stay sufficiently ahead of competitors. Non-reimbursed R&D expenses sacrifice short-term profitability, with unknown *a priori* results. Conversely, insufficient R&D puts the firm into a reactive position, unable to control its destiny and, ultimately, its livelihood.

---

[8] Although Johne considered product to be a tangible item, and was focused on technologically oriented innovations, we prefer to think of "product" as both physical and non-physical (e.g., service) consignments which are made to the customer, regardless of the actual level of technology presented. Naturally, clear-cut definitions of "customer" can cloud this analysis. We address these concepts further in chapter V.

Utterback (1979) asserts that there exist two predictable cycles for companies participating in process and product innovation; both of these cycles are relevant to the firm's age. Refer to Exhibit II.2. The ***process innovation cycle*** is very similar to the cycle proposed by Schumpeterian product life cycle theory. The ***product innovation cycle***, on the other hand, is a decreasing function of firm age.

EXHIBIT II.2.

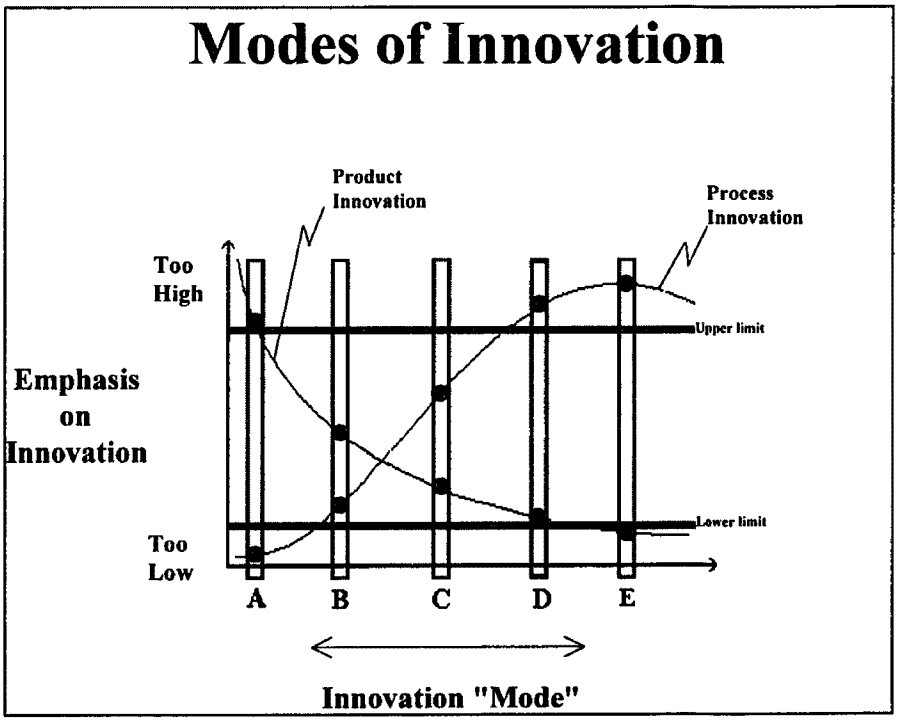


If the assertions of these cycles are true, then some observable tradeoff between process and product innovation should be found to work best for an organization. Though the abscissa of the above graph is indexed as the age of the firm, it may be overly convenient to consider these curves to be unidirectional in time. Perhaps it is possible for a firm to

sway left and right on these curves, according to a variety of managerial, market, and structural changes.

Further, is it fair to assume that the heights of these curves are additive algebraic indicators of innovation? For example, does 5 emphasis "units" of product innovation, coupled with 5 emphasis units of process innovation give the same results as a 7:3, 2:8, or 9:1 product/process ratio? It seems that there are some fundamental minimums and maximums, below or above which the firm cannot sustain itself. Of course, the values of these thresholds may very well depend upon a number of innovation independent factors, such as liquid assets, marketing capabilities, social trends, etc. Thus, it may be more reasonable to consider Utterback's drawing with a few changes, as shown in Exhibit II.3.

EXHIBIT II.3.



In this visualization, a firm may pass from mode to mode as non-development conditions dictate. For a particular organization, it may be found that mode "C" works better than mode "B". In time, this "optimal" modal position may change, as well. The point here is that innovative behavior may not be strictly time dependent, but rather controllable with appropriate organizational "innovation attitude". According to this visualization, modes "A", "D", or "E" would be inappropriate, as product innovation or process innovation activities reside outside viable economic limits (either too high to be internally affordable or too low to stay market competitive).

A Harvard colleague of Schumpeter, A.P. Usher, developed an organizing framework for the concept of invention and linked it to innovation and technology (Usher, 1951, 1954, 1955). The "process" of invention was described as four basic, sequential steps:

1. Perception of the problem (i.e., need recognition)
2. Setting the Stage (i.e., environmental conditioning or "fertilizing")
3. The Act of Insight (i.e., the proverbial light bulb!)
4. Critical Revision (i.e., feedback and assessment)

Usher termed this process, derived from insights of Gestalt psychology, the ***cumulative synthesis*** theory of invention. This was in response to, and in direct conflict with, the ***transcendentalist*** approach and the ***mechanistic process*** theory of invention[9]. In the

---

[9] The ***transcendentalist*** approach alleges that invention is the result of inspiration from the occasional "genius" who intuitively achieves correct knowledge of essential truth. The ***mechanistic*** process theory says that invention proceeds under the stress of necessity, regardless of whether the person can be considered a genius. Usher refuted both of these approaches toward invention, though he did show slightly

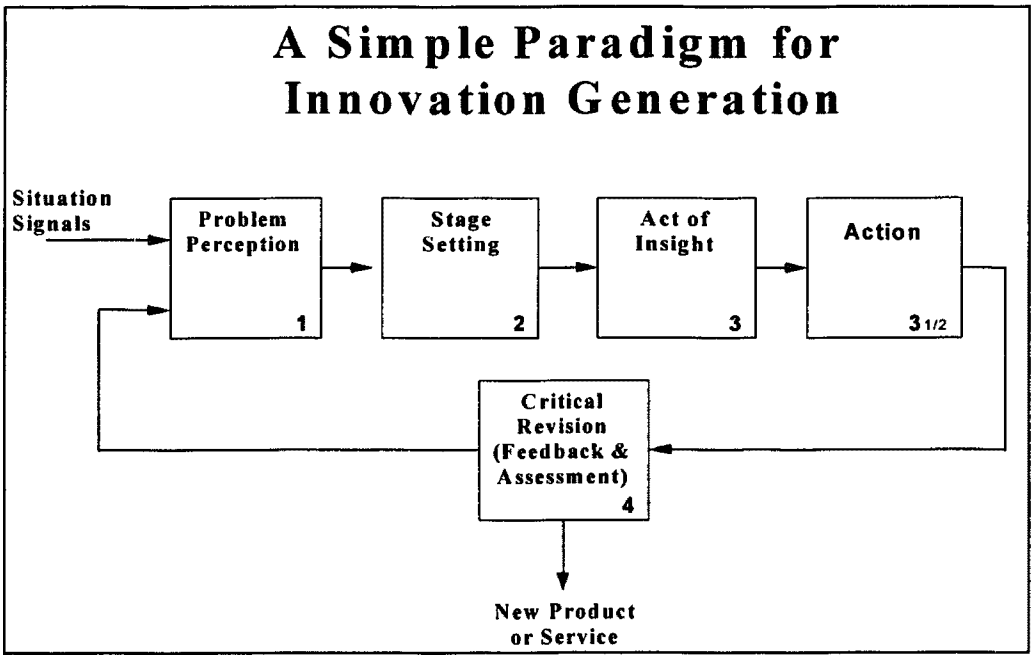
cumulative synthesis approach, repeated sequences of these steps automatically result in new "things" or "acts of instinct." These could encompass a wide range of ideas, including inventions and "innovations". Using this approach, Usher maintained that processes of invention and innovation need *not* be distinct. Further, he recognized that individual inventions could set the stage for a major invention. When this happens, and the market responds favorably, such an act could be recognized as "innovation."

Unfortunately, there seems to be a significant gap between steps 3 and 4 in Usher's framework, particularly if one wishes to move from a mere invention process to a more encompassing ***innovation*** cycle. This may be filled with physical ***action***, which converts the bright idea into a tangible object (e.g., prototype) or process (e.g., service). The nature of this gap filling activity (step 3-1/2) is the major focus of the current study. As a continuous cycle, this modified framework is shown in Exhibit II.4.

---

more compassion for the mechanistic hypothesis (largely because of the extensive empirical results of the Chicago sociologists who directed this cause). For more description of these alternative theories of invention, see Ruttan (1959).

EXHIBIT II.4.



Innovation cycles which carry similar features to this framework have been offered by many authors. A more comprehensive framework, incorporating steps for organizational change, was presented by Zaltman, Duncan and Holbek (1973). Their framework carries additional complexity about certain internal psychological and social resistances which are a way of life in any organization. If one wishes, their framework may be encapsulated *within* the simple representation forwarded here.

James March and Herbert Simon (March and Simon, 1958) paid particular attention to the processes by which individuals search for solutions of identified problems (i.e., step 3 in the above framework), and acknowledged the oft assumed idea that the kernel of new ideas are generated from both within and outside the firm. In the current work, we postulate that "within the organization" does not necessarily translate into adequate

communication among organizational members. Simon seems to have come to much the same conclusion, for he has been intimately associated with the development of cognitive science, particularly the "branch" of cognitive science that involves Artificial Intelligence for improved problem solving.

Burns and Stalker (1961) discussed the internal decision-making activities of organizations attempting innovation, and contemplated processes by which existing alternatives are incorporated or rejected. They seem somewhat passive about the process by which ***needs*** and potential ***solutions*** are identified, however.

In a comparative review of the work of Schumpeter and Usher, Vernon Ruttan (Ruttan, 1959) attempted to clarify the concepts of ***invention***, ***innovation***, and ***technological change***.[10] Drawing elements from Schumpeterian (economic development) and Usherian (cumulative synthesis) theories, Ruttan suggested a semantic framework. Specifically, he suggested "invention" be deemed a special subset of "technical innovation"--the subset of activities which result in products for which patents can be obtained. "Innovation" should be a more encompassing term which covers all the steps of the cumulative synthesis approach. In this definition, there are numerous kinds of innovation--scientific innovation, technical innovations, organizational innovation, etc. Finally, "technological change" should be any activity which contributes to changing the efficiency of creating

---

[10] While this is not a dissertation on the importance of technological change *per se*, technology has often been associated with innovation and new product development. Thus, it was inevitable that technology would become a non-trivial, albeit tangential issue in this study.

outputs for any level of input (i.e., changes of coefficients in the production function)[11]. While these adoptions seem to be an endorsement of Usher's theory, Ruttan maintained that Shumpeter's theory of economic development should remain intact, particularly with innovation as the overriding theme of central importance.

Edwin Mansfield (Mansfield, 1961, 1968, 1979, 1980, 1984, 1988) has conducted exhaustive analysis on the impact of technological change on industrial performance. He deliberated on a variety of possible production functions which could be utilized to estimate economic prowess as a result of technological development. Results of these studies indicated that no single function could adequately describe the impact of the multiple, amorphous, continuously changing technological states in an economy. The works of Mansfield did, however, delineate the contributions of technological change to capital (equipment) and labor savings, for several major industries.

Perhaps the foremost modern day authority on the practice of innovation is Peter Drucker (Drucker, 1985, 1989, 1992). Certainly he is among the most well known of current management gurus, and carries quite a following among managers throughout corporations world-wide. As with many other researchers on the subject, Drucker has emphasized the need for innovation; more pointedly, he advocated the ***need*** for entrepreneurial activity within organizations as the enabling mechanism for innovation. In this sense, his focus may be considered to be "step 2" (stage setting) of the Usherian framework[12]. In an attempt to help entrepreneurs organize their search for innovative

---

[11] "Technological change" in this context is very similar to the concept of "process innovation." For a more thorough review of the roles and impacts of technological change, refer to Rosenberg (1971), Giaini and Louberge (1978), Stoneman (1983), or Scherer and Perlman (1991).

opportunities, he recognized seven sources of innovation: These were classified into four internal (within an organization or industry) sources and three external (social, political, or philosophical) sources:

(I) ***Unexpected***: The circumstance in which the "innovator" is not trying to innovative, but *accidentally stumbles* on a huge market success.

(I) ***Incongruities***: A newly perceived, yet long-standing discrepancy between what is existent in a process or product, and what is currently *possible.*

(I) ***Process Need***: A discrepancy between what is currently being delivered in a process and what is *needed* from that process.

(I) ***Industry/Market Structure***: The condition under which *standards* of the market and how competitors operate *are changing*, whether because of market growth, convergence of new technology, or new paradigms.

(E) ***Demographics***: Changes in the *sociological makeup* of the market.

(E) ***Perception***: Changes in *opinions* or *attitudes*, regardless of actual facts.

(E) ***New Knowledge***: The development or discovery of a *new idea* or technology, usually coupled with other existing knowledge, which synthesizes into a new, *previously unconsidered opportunity.*

Such sources of innovation were identified in an attempt to develop an enabling paradigm for innovation, specifically for entrepreneurs. Drucker made extensive use of reflective

---

[12] Drucker's discussions of innovation touch on all aspects of the framework. In fact, Drucker developed his own framework for entrepreneurship-innovation relationship. Drucker is probably best known for his observations and deliberations of companies and industries inadequately performing need recognition, or honestly asking "What is our business?". This, of course, may be considered step 1 of our modified Usherian framework.

case studies for illustration of each of these sources. Based upon observations or research, he offered some do's and don'ts for innovators.

Unfortunately, Drucker's work has not been considered "scientific enough" for many segments of the research community, largely because of the holistic, reflective nature of his case reviews. While it may be true that such work does not carry the theoretical rigor which many researchers might hope for, his work does provide a practical lean, which seems necessary for any popularization of a study in this very important field.

For all the suggestions and cases which Drucker presented, however, we still have no theory or algorithm for successful innovation. In fact, Drucker himself asserts that innovation is a highly ***unpredictable*** activity, conducted by a wide spectrum of individuals. No real-time diagnostic methods for the innovation process are presented. All that we may hope for, it seems, is that we may keep the organizational grounds fertile for new ideas. In the current thesis, we explore this unpredictability, as it is manifested within existing organizations conducting new product development.

Documented economic impacts of innovation have proliferated, particularly in the past decade. Perhaps the most vivid demonstration of innovation's economic efficacy, however, was created more than 30 years ago. Zvi Griliches extensively studied the diffusion of innovation, particularly that of hybrid corn across farming communities in the central U.S. (Griliches, 1957, 1958, 1960). Although he was not the first to consider the characteristics of innovative diffusion (see, for instance, Pemberton (1936, 1937, 1938), Bowers (1937a, 1937b), McVoy (1940)), nor was he the first to consider hybrid corn diffusion (see Ryan and Gross (1943)), Griliches added practical, "hard" data to the

discussions. After tracking the distribution of corn yields and profitability of farms which used improved hybrid seed, a non-linear, non-uniform (i.e., "non-smooth" or "quantum") diffusion process was discovered. Affected by a number of factors, including geography, local existing economic conditions,[13] communication profiles among neighbors, education, and existing technological level (i.e., tractors and other equipment in use), the documented diffusion resembles the diffusion effects seen in biological and chemical (gaseous) phenomena. Geographic maps in this study show locally isolated "pockets", surrounded by areas of intense innovation acceptance and use. Such maps resemble certain complex-math derived fractal mappings. Moreover, the ***dynamic*** nature of this diffusion process has a curious resemblance to the popular "neighborhood" models in cellular automata.

In a particularly interesting treatise dedicated to the topic of innovation diffusion, Lawrence Brown (Brown, 1981) offered a taxonomy for the diffusion of innovative products or services. He forwarded four perspectives, offering several case reviews for each. The viewpoints presented included ***adoption, market/infrastructure, economic history,*** and ***development***. They are briefly described as follows:

---

[13] One socio-economic conclusion of this hybrid corn study was that the newer hybrids were more readily implemented by farmers who could afford to experiment with them. Thus, the benefits (i.e., profits) of such innovation were only realized by those with better existing economic standing. This, Griliches asserted, could attenuate regional economic disparities over the longer run.

- The ***adoption*** perspective, in which the geographic spread of an innovation is considered as the result of a communication and learning process. This can also be characterized as the "demand development" aspect of diffusion.
- The ***market and infrastructure*** perspective, which considers enabling mechanisms and existent conditions for adoption. Often asserted to be the "supply" view of innovation diffusion, this perspective considers both the establishment of diffusion "agencies" and the strategies by which such enabling organizations operate.
- The ***economic history*** perspective considers the stability of the innovation under consideration (i.e., how much the innovative product itself changes during the diffusion process). Such adaptation of the innovation could be the result of further developmental refinement/improvement or customization for specific uses. Because this facet of the diffusion paradigm can have the effect of stalling diffusion (e.g., because the adopter waits in anticipation of further refinements), the economic history perspective is considered a *precondition* for diffusion.
- The ***development*** perspective contemplates the impact of diffusion on the welfare of the individual, the region, or the diffusion infrastructure. As Brown pointed out, the reverse condition also needs consideration: existent development conditions can affect the diffusion process itself.

While the innovation diffusion literature at large[14] has focused on the adoption perspective, Brown deliberated on the other three perspectives. Presumably, this was

[14] See also, for instance, Rogers and Shoemaker (1971) and Hagerstrand (1967, 1974).

done to "balance" the stage of innovation diffusion research. Nevertheless, Brown emphasized that each of these perspectives are complimentary to one another--none stands alone as the "most" important. In fact, Brown considered the interdependence which may exist among these perspectives as an important point that has often been overlooked in the literature.

Baumol (1990) proposed the hypothesis that innovation, more specifically entrepreneurship, could have productive or destructive consequences for society, depending on the nature of the innovations. The overall nature of innovations in a society, he proposed, were highly (if not entirely) dependent on the "rules of the game", as defined by official or unofficial public policies. This proposition, though not proven per se, was well illustrated by selected cases of innovation in Ancient Rome, Medieval China, the Early and Late Middle Ages, and the current Industrial Revolution.

Much ado has been made in the business literature about the effectiveness of small firms to innovate more adeptly than their larger counterparts. Often, such determination has been made in the light of singular case reviews rather than unbiased industry-wide studies which consider more complete competitive conditions.[15] Nonetheless, the importance of small firms in job creation, technological innovation, and general economic

---

[15] Typically, such studies compare the activities of two or more firms with which the researcher(s) are quite familiar. The case analyses which ensue typically are reflective of *ex post facto* knowledge of the firms' market performances. Given a typical researcher's familiarity with a variety of firms, it is tempting for authors to conveniently "select" firms which fulfill their hypothesis. Even "random sampling" schemes can introduce biases of self-selection, which are often ignored. In our field studies, references to such case "cherry-picking" were numerous (and, sometimes, surprising!). To us, this indicates an unfortunate field attitude of blind research bashing, even if a large portion of the research under question was well conducted.

"rejuvenation" is accepted by many economists, public-policy makers, and academic-based management analysts[16].

But what does it mean to "behave like" a small firm? It is widely observed that many efforts to "act small" have resulted in major market disappointments. Clearly, small is not the end-all to successful innovative behavior. Perhaps there are particular characteristics among innovative small firms from which all firms could learn.

In a study of 50 small Texas manufacturing firms (Khan & Manopichetwattana, 1989), 42 managing characteristics were correlated with innovative behavior. Results revealed five distinct types of firms: two "innovative" and three "non-innovative." That less than half (48%) of the small firms could be considered innovative was not surprising. It *was* revealing that these small firms carried quite diverse characteristics. As a correlate with the level of innovation across all firms, the following characteristic activities showed *no* significant relationship to successful innovation:

---

[16] For instance, see the account of the National Science Foundation's experiences (Colton, Ryan, and Senich, 1985) and Roy Rothwell's deliberations on the direct and indirect nature of such public policies (Rothwell, 1985).

- available resources
- R&D spending
- environmental hostility
- explicitness of firm strategies
- functional communication
- executive locus of control
- executive demographic profile
- decentralization of activities
- level of functional controls
- environmental scanning
- firm age

Though innovative firms covered the entire spectrum on each of these characteristics, it is interesting to note that particular levels of these characteristics were not necessarily antecedents to successful innovation. Many, but not all, innovative firms carried common characteristics; however, many non-innovative firms carried the *same* characteristics. Certain attributes were found to be significantly associated with innovation. In descending order of statistical significance, they included the following:

- ***Proactiveness***: The characteristic of being ready for new challenges *before* the actual challenges arise (i.e., attempts at market leadership).
- ***Risk-taking***: The characteristic of attempting to try a new idea or product, *in light of apprehension* of others (i.e., not afraid of failure)[17].
- ***Integrated decision-making***: When a firm engages in decision-making that is based upon *consideration of all functions* within the firm (i.e., non-reductionist decision-making)

[17] It is interesting to note that risk-taking seems to actually be a perceptually-based activity. Drucker (1985) attests that innovation is actually a risk-adverse, "opportunity-focused" activity which only looks risky to outsiders who don't see the overall picture of the potential risks. Of course, this does not imply that entrepreneurs always see all risks... they merely do not consider themselves to be engaging in unreasonable risks. Drucker asserts that entrepreneurs tend to be very conservative in their exploitation of opportunities.

- ***Environmental dynamism***: The presence of a *constantly changing* market or industrial environment. This seems to have the effect of forcing innovation on the organization, just to survive (i.e., innovate or die!).
- ***Product differentiation***: When a firm's products or services are *distinct* (and/or marketed as distinct) from traditional competitive products. More differentiation correlated with more innovation.
- ***Less executive experience within the firm***: Although an executive's age, per se, seemed to have little or no bearing on innovativeness, in-firm experience was inversely related to innovative behavior. This supports the concept of "bringing in fresh blood" to overcome stagnation.
- ***Environmental heterogeneity***: Seemingly related to environmental dynamism, this aspect considers the level of *diversity* of competitive products or services in the industry.
- ***Strategic integration***: The coordination or *interactive capability* of a firm's products or services.
- ***Less emphasis on technology development***: While technology may ultimately prove important, it should be the *result* of market need, not the *antecedent* of that need. This supports Schumpeter's inclination that "innovativeness" is not related to the ability to invent.

Roy Rothwell and Walter Zegveld have considered the roles of small and medium sized enterprises (specifically, manufacturing firms with less than 500 employees) in economic growth (Rothwell and Zegveld, 1982). They observed that firms play various roles, depending upon the social attitudes, national cultures, and their individual past performance. They advocated a much more sophisticated consideration of the issue over the "innovativeness" of small firms vs. large firms. In reviewing the development of the semiconductor industry, it was demonstrated that small and large firms *each carried significant roles* at different stages of the industry's development. They remarked that the instantaneous status of the industry and marketplace could have both enabling and stifling

affects on small and large firms; small firms were better suited for certain conditions; other conditions required the resources or distribution capabilities of larger firms. Generally, smaller firms demonstrated better capabilities to respond to the market when an industry is in its early, "fluid" stages. As an industry grows in size (i.e., "matures"), greater barriers tend to exist for smaller innovative firms, relegating the exploitation of new concepts to the larger firms. Such time-dependency and industry maturity aspects of innovative capability has, unfortunately, been ignored by too many authors since Schumpeter. If one has the stamina[18], this reference should be reviewed by anyone interested in the dynamic tradeoffs of innovation between small and large firms.

R&D expenses of government and industry have been tracked, with an interesting complementary relationship apparent. Specifically, Mansfield and Switzer (1984). found that government research in energy provided an incentive structure for private firms to also conduct (and pay for) energy research. Levy and Terleckyi (1983) had also found such a relationship; they determined that every dollar of government expenditure resulted in 27 cents of additional private spending. No other studies reviewed have been able to attribute such a precise spillover effect from public to private spending. In fact, there has been much debate about the existence of specific incentive structures among private firms, with respect to public spending (see, for instance, Branch, 1974; Kamien and Schwartz, 1982; Jaffe, 1986; Ulrich, 1986). Measurement of such a phenomena is admittedly difficult, because of time lags and the propensity for imprecise estimates of

---

[18] My edition of the Rothwell and Zegveld book was printed in "mice-type" and was quite dense in its use of tables and graphs, making reading it less than pleasurable. Likely, a reader will find it necessary to re-read this reference, as there is a wealth of information within, which does not jump out during an initial pass.

unavailable data. Yet, there seems to be a resounding feeling that spillover effects between government and private R&D can exist, under certain conditions. From the literature referenced above, it appears that private and public R&D projects offer highly divergent investment returns, with more favorable (& consistent) returns tending to arise from privately funded and conducted projects.

Al Rubenstein has considered the various internal roles of decentralized firms in conducting research, development, and innovation (Hertz and Rubenstein, 1953; Rubenstein, 1957, 1980, 1985; Rubenstein et al, 1970; Rubenstein and Ettlie, 1979; Rubenstein and Geisler, 1988). Although his works have covered many different bases with regard to technology development and the practices of R&D activities in relation to production and innovation, perhaps his most significant contribution, in relation to our study, has been in assessing the communication architecture (and resulting communication practices) among engineers, as well as between "techies" and "non-techies". The concept of communication "gatekeepers" has been based upon the observance of hundreds of companies over the course of more than 40 years. It has been observed in organizations large and small, centralized and decentralized, public and private, even autocratic and autonomous. Time and time again, the insufficiency of un-garbled communication has been pointed out as a key driver for disappointing developmental organization performance, morale, turnover, and inter-divisional support.

As related in his manifesto work (Rubenstein, 1989), the issue of centralization vs. decentralization has had a pendulum-like effect over time: he observed that there seem to be continual pressures among decentralized organizations to become more centralized; conversely, centralized organizations realize pressure to behave in a more decentralized

fashion. Often, he observed that changes from one mode to another tend to be explosive or highly disruptive to the existing participants[19]. According to his studies, there are few progressive firms which have been able to retain stability at the fulcrum of the centralization-decentralization teeter-totter. It is difficult to imagine that such firms will be able to retain such stability forever, given the perennial flux of the marketplace, competing technologies, and internal managerial careers.

Based upon the observed characteristics of communication among engineers, much of which is in accordance with such past research, the current thesis attempts to expand the research frontier of communication dynamics and structural dynamics within development organizations.

As we conclude this cursory review of some relevant work on innovation, let us return to the observations of economist Joseph Schumpeter. In his 1939 treatise, *Business Cycles*, he developed a paradigm for the natural, regular growth and contraction of complete economic systems. He considered three major cycles:

- ***The Kitchin Cycle*** (lasting approximately 40 months)
- ***The Juglar Cycle*** (a period of roughly 10 years)
- ***The Kondratieff Cycle*** (Approximating a 52 year period)

---

[19] I liken this characteristic to the dangerous bane of beginner sailors, the ***accidental jibe***. This is the condition when a wind shift allows higher pressure air to get behind the leech of the sail and causes it to violently swing to an opposite tack (sometimes by more that 180°). The primary danger, of course, is that of the crew being physically (and quite surprisingly!) struck by the swiftly-moving boom. In extreme cases, such a maneuver can result in broken rigging (e.g., the mast!) or even capsizing of the craft. The popular press is full of examples where such analogous results occur in real organizations.

Of these three, Schumpeter considered the first to be a relatively harmless, self-correcting cycle. He felt that it would only be noticed (and feared) by short-term oriented economists, who did not have the foresight or desire to see more prevalent longer-term fluctuations in the business cycle. The Juglar cycle was considered to be somewhat more serious, because its ramifications could be felt by larger sectors of the economy and would last for more time. Yet, even such an economic cycle tended to self-right itself...except in those periods where it was superimposed with the Kondratieff cycle.

The Kondratieff cycle (or Kondratieff Wave, as it is often referred to) was deemed to be the most serious of all, however. Initially recognized in the 1920's by a Russian economist, Nikolai Kondratieff (who was subsequently arrested by the Marxist-oriented secret police as an anti-government subverter, because of remarks about this discovery, shipped to Siberia and, ultimately, executed), this cycle warns of a huge, systematic tendency for real prices to fluctuate in an economy, and for preoccupied participants (i.e., consumers, businesses, even the government) to engage in behavior degenerative to the greater economic system. At the peaks of the Kondratieff cycle, the economy booms, employment levels rise, and the standard of living rises at rates higher than average for that economy. At the troughs, the economy undergoes serious depression.

Schumpeter found these cycles to be characteristic of the aggregate level of innovation in an economy. As new innovations were introduced, he hypothesized, they could "eat" production from less innovative firms in the market. This could have the net effect of reducing the employment levels in the economy. Such a paradox, wherein improvements in firm performance actually hurt an economy by way of increased unemployment, was

coined the "gale of creative destruction" by Schumpeter. It is easy to see that unbridled successes in an economy, characteristic near the peak of a Kondratieff wave, could spell the deceleration and ultimate slowing of that economy. While economies are at or near a trough, remaining players re-structure themselves for survival, their improved practices permit them to flourish, and the economy slowly rises with them again. Schumpeter was declaring, as he had done nearly thirty years earlier[20], that dynamic disequilibrium in an economy *was created by* innovation and that continued disequilibrium is *normal* for a healthy economy. This was in conflict with theories of classical economics, which involve searching for optimization or establishing equilibrium based upon technological improvement and subsequent price adjustment. Is it any wonder that Schumpeter's views were not well accepted by the greater economic community?

It is interesting that Schumpeter's work has been re-addressed so fervently over just the past few years. Throughout the middle forty years of the twentieth century, Schumpeter's theories fell on deaf ears in the economics community. This has been asserted to be the direct result of the rise of Keynesian economics during this period[21]. Yet, as dynamics of Schumpeterian cycles seem to remain valid through turbulent times, they have become more and more popular. Even the interest in Kondratieff and long wave economic theory have exploded in recent years[22].

---

[20] According to Drucker (1985), p27, Schumpeter came to this same conclusion as early as 1911, in a work titled The Theory of Economic Dynamics.

[21] See, for instance, Drucker (1992).

[22] An admittedly less than statistically significant indication of this trend was evident when I began reviewing the libraries for past work on the Kondratieff cycle. Of the 23 holdings which existed, only one was published prior to 1984! (That one was the 1972 work of Shuman and Roseneau.)

Innovation is clearly a diverse field. Review of the literature has revealed many, often conflicting, definitions of innovation. Among some theorists, there is a "natural" melding between innovation and entrepreneurship. Among others, innovation is a very particular activity which occurs on a seemingly random basis. Some have felt that innovation can be a learned activity; others feel that it cannot be taught, is rather innate, but can be assisted through increased awareness of the environment of the market, industry, or firm; still others feel that it is an inevitable activity which will occur despite the efforts of policy makers or nurturers. Whom is right or wrong may never be conclusively decided; we may find that there is some underlying blend among each of these views, or particular circumstances where one position holds up better than another.

Perhaps the most important point to carry from this review, however, is this: ***Innovation research is still an embryonic science, still dominated more by opinions than facts.*** Until we learn much more about the true nature of innovation (as opposed to experiential assertions of facts), many of our attempts to increase our innovative capabilities may prove more fruitless than rewarding. Innovation research is as much or more a science about people as it is about techniques. In this vein, we discuss some findings from the cognitive sciences later in this chapter. For now, let us review some past work on product development.

2.2.2. Product Development Research

While reviewing research on product development, an over-riding theme was quickly apparent: unlike the somewhat abstract discussions on innovation, authors who discuss product development are (almost inevitably) trying to sell something! I have seen perhaps 500 different "articles" over the past five years which were little more than formalized advertisements for a product or service which will make one's engineering-life easier. Hardware, software, proprietary methodologies, and a swarm of consultation services (both external and internal to academic institutions) are constantly peddled as the latest and greatest ways to enhance product development. This theme, perhaps, is a result of an attitude that the problems of faster, higher-quality, cheaper, (fill in any other adjective you like!) product development have been, in theory, solved. A prevailing attitude seems to be, ***"Now that the technical solution is in hand, all we have to do is implement it."***

The need for better implementation of some solution has been well established over the years. Though estimates of the success rate of new products are quite diverse, and depend on factors such as which industry, what kind of customer base (e.g., industrial or consumer), and the definition of "new" products (as opposed to slight twists of existing product), all studies seem to indicate that the success rate is low.

In a study of over 13,000 new product releases from over 700 companies, Rockwell and Particelli (1982) estimated that about two thirds of commercialized products (i.e., those which actually make it to the market) were deemed successful. If this seems high to the reader, don't feel alone. Consider the makeup of these "new" products:

- 10% were "new to the world"
- 19% were new product lines
- 26% were additions to existing product lines
- 26% were revisions or improvements to existing products
- 7% were existing products that underwent cost reductions
- 11% were merely market repositions of existing product

Thus, only about 29% of these "new" products were really new to the organization! To put this in perspective, suppose that all of the remaining 70% were market successes; this would mean that a little less than a third of the products (29%) accounted for the 33% overall failure rate. If these numbers are accurate, this translates into an 88% (0.29/0.33) failure rate. Further, 60% of the products in the sample were classified as industrial products, which often are developed closely with the customer, and thus tend to enjoy higher success rates. When these considerations are taken into consideration, the projected normal failure rate in this study falls well in line with other studies, which tend to hover at around 90%[23].

Various accounts have been made about new products that are stillborn (i.e., never proceed out of the development organization or are rejected by marketing management before even being seen by a customer). For reference, consider Gruenwald (1985), Davies (1982), White (1976), or Twiss (1974), .Such "pre-release" failures can be attributed to capitalization problems, incorrect market assessments, inadequate cost/benefit or break-

---

[23] Even in the "modern" computer industry, failure rates have been observed observed to be from 72% to 91%. See Infoworld, Feb 6, 1995, p. 62.

even estimates, managerial predisposition, government regulation changes, or even failed technical capability. However, the most prevalent, observed internal problems with new product development involve slow organizational responsiveness (even though individual responses are adequate) and system-wide misunderstanding of product objectives. Though there can be many reasons for the high failure rate of new product developments, overall system sluggishness (which results in missed opportunities) and inappropriateness of developed product (which also results in missed opportunities) rank high.

As with the innovation literature, research on new product development takes on many forms. Unfortunately, some "new findings" are merely the result of rediscovery of old knowledge. Some writings are insightful and descriptive case reviews of existing practices. Some such writings seem to be transferable from site to site. Others seem so site-specific that little useful knowledge can be carried from them. There are many books on the subject, in which authors attempt to bridge isolated findings of past experiences.

A good review of some ***practices*** of new product development was forwarded by Steven Wheelwright and Kim Clark (Wheelwright and Clark, 1992). While not a literature review per se, and not a discussion of new research findings, this work considered the variety of different product development techniques which are employed by various firms, large and small. Of particular interest was the characterization of a "developmental funnel" outlining the process by which new ideas and technologies are screened in as features of new products or as processes in the manufacture of new products. As an overview of the product development process from a traditionally linear viewpoint, this work is useful reading.

Another excellent resource for product development managers and researchers has been forwarded by Preston Smith and Don Reinertsen (Smith and Reinertsen, 1991). In their work, the authors discuss the value of reduced development time and offer several suggestions for doing so. They seem to have incorporated many of the concepts forwarded by Tom Peters and Robert Waterman (Peters and Waterman, 1982), which include restricting the number of objectives within an organization to a reasonable number, considering full life-cycle profitability (instead of just local engineering function costs), building a holistic approach (i.e., overall process orientation instead of local function optimization), and only selecting appropriate tools for process improvement (instead of assuming existing popular tools will work "because Company X uses them"). Of the many discussions on product development which have been reviewed for our current study, Smith and Reinertsen's work seems to be among the most pragmatic; yet, it may also be considered the most controversial when compared to the traditional manufacturing-oriented paradigms of product development. In this regard, this work is similar to (though much more detailed in its orientation around product development) the paradigm-bashing, never-stop-asking-why, management writing of Robert Townsend (Townsend, 1970, 1984).

While we are still in the controversial mood, it may be good to consider a widely-known study. By now, many progressive manufacturing and product development managers are wholly familiar with the International Motor Vehicle Program (IMVP) research studies[24]

---

[24] For those not in the know, the IMVP was an outgrowth of the MIT-based International Automobile Program at the Center of Transportation Studies. The earlier program produced a number of papers, as well as a collaborative book The Future of the Automobile (Altshuler, Anderson, et al, 1984), which examined the problems facing the automobile industry.

conducted at Massachusetts Institute of Technology (MIT). The landmark discussion of the study can be found in a book, *The Machine that Changed the World* (Womak, Jones, et al, 1990). MIT research affiliates also prepared 116 monographs in association with this study. This study is intriguing for three reasons:

- The study ***has become well-known*** among engineering managers world-wide in a relatively short time period.
- This study seems to be ***endorsed*** by a large number of professors and high-level managers in many product development organizations.
- Although insightful in a number of areas, the study's ***conclusions are ill-founded***, in many regards, with respect to the observed and prescribed nature of new product development.

The last point may seem intriguing, even controversial, to many readers. However, the first two points are much more interesting, particularly in light of the third point. This study is classical in its thorough assessment of the state of the world-wide automobile industry. The production efficiency, costs, market share, even supplier-chain characteristics which are described carry great weight. In each of these areas, the MIT study concurs with observations which were made in the current study. That such a thorough dissemination of the industry was conducted, is in itself, a significant achievement; this is particularly true when one considers how turbulent the automobile industry has been over the past decade. Given the void of existing objective information

prior to the study, it should not be surprising that it became virtually instantly recognized by managers world-wide[25].

We have discovered that many "professionals" including university professors, executives, and management consultants have embraced the study's contents at face value, without checking with the *real* experts in product development--design engineers. The study seems to rely on an assumption that "what works for production also works for development." In our current study, development engineers repeatedly revealed that their processes are markedly different in character than those of manufacturing, and cannot be "automatically analyzed" in the same fashion. The concept of "Lean Development," which is presented in the study as the integration of leadership, teamwork, communication, and simultaneous operation is so generic that a first-year management student who professed such a plan would be summarily dismissed as being too naive to study the case with any real insight; nearly everybody involved in development knows such characteristics are desirable already! The suggestions offered to help meet such goals read like a JIT production manager's handbook. The reasons for certain development success stories are asserted to be similar to the reasons that lean production works (under certain circumstances). First hand experience, coupled with the experiences of several hundred design engineers, tell a different story. Unfortunately, much of the "professional" readership of the MIT study seems to be enamored with the source of the study, not its content. While strong on credibility and marketability, the study's logic falls

---

[25] During one of my first visits with an automobile industry executive in Germany in mid-1991, he asked of my opinion of the book. Not having read it yet, I told him as much. He remarked that it was the consensus of his engineers that the commission, on its visit to his organization, was not really looking for objective information about how his organization operated, but rather were searching for verification of their already established conclusions. Such pre-conceived conclusions, naturally, were what was published.

between the cracks, at least as far as system-wide product development is concerned. It is disappointing to observe that many intelligent and influential people could so easily fall into the trap of blindly following the study's conclusions.

After reading the remainder of the current thesis, the reader may wish to verify or argue with the third point.

John Ettlie and Henry Stoll (Ettlie and Stoll, 1990) presented an insightful discourse, describing some of the issues which real managers face, as opposed to merely exhorting what should be done. In conjunction with the experience of development managers from "the trenches", they forwarded the concept of ***disciplined anticipation*** as a managing mode. Foremost in the successful, experienced managers' repertoire is the ability to sense the appropriate "degrees of freedom" (autonomy) which are necessary within development teams, and to understand when and why the level of such freedom should wax or wane. Coupled with this dynamic variable is the notion of coordination between functions in the development process. In the cases presented in this work, it was refreshing to see their recognition that many of the "new" techniques (e.g., simultaneous engineering, DFx, GT, QFD, SPC, FMA, Value Engineering, etc.) which others extol have actually been around for ages, albeit with different terminology.

There has been considerable discussion on the topic of ***design*** as an important ingredient in the success of a product over its life cycle. Adding some confusion to the topic, design carries different definitions among different authors. To some, it is synonymous with the complete innovation cycle. To others, it can be merely the packaging, or aesthetics, of a product. A full spectrum of definitions between these two extremes exist, as well.

Christopher Lorenz (Lorenz, 1986) suggested that design be considered the "all-around role of coordination and integration" of products, from concept creation onward. Although not always agreed upon by the captains of industry[26], it has been forwarded by Lorenz that individuals responsible for product design (under this more encompassing definition) must carry many similar, holistic, considerations as anyone responsible for overall corporate strategy (see also NRC, 1991, Stanford, 1989; Gorb, 1986).

In support of yet another "appropriate" focus of design, consider the "engineering technique" category of design literature. There are many references of this flavor, often written by former engineers who feel the need to download their cleverness as practicing engineers. Focused on development of physical product designs, they contemplate various techniques and axioms for engineers to follow, based upon practical experiences. The theme of such works seems to stem from a paradigm of "cerebral emulation" (i.e., "think like I think" and you'll be a good engineer). Unfortunately, we have seen no evidence for the efficacy of such works to enhance design performance in development organizations. Fortunately, such types of works do provide a fair degree of holism, due to the number of different, diverse examples given. Thus, there is some inkling of hope for cross-fertilization to arise from this style of writing.

One such writing has been presented by MIT professor Nam Suh (Suh, 1990). For Suh, the orientation of design is decidedly on the development of accurate functional

---

[26] For an insightful look at this issue, consider the reaction of the former Ford CEO, Don Peterson, and others to the question of whether or not a designer typically possesses qualities essential for acting as an effective CEO, see Stanford Design Forum (1989), p. 39.

requirements (FR's), to be followed up with design parameters (DP's). In this way, design is defined as the creation of synthesized solutions that satisfy perceived needs through a 1-to-n mapping from the FR's in a functional domain to the DP's in the physical domain. To facilitate this process, two general axioms are developed, the ***Independence*** Axiom and the ***Information*** Axiom[27]:

- Independence Axiom: Maintain independence among functional requirements.
- Information Axiom: Minimize the information content of the design.

Based upon our field study, such a design philosophy parallels that which is used in defense-system developments and many other large-scale development projects today[28]. The first axiom essentially states that designs should be robust: ***deviation of one parameter should not affect the suitability of other existing design parameters***. The second axiom can be boiled down to three words: ***keep it simple***.

While Suh offers some interesting mathematical reasoning (much of which we have, coincidentally, used in the current thesis) for why these axioms can provide best design system performance, it is not at all clear that such axioms are universally correct prescriptions for best development performance. They do offer good starting points or guidelines, however, for managers who are not able to otherwise control, much less understand, their development processes. This, of course, does not bode well for anyone trying to manage highly integrated products from a non-reductionist perspective. If one is

---

[27] Suh (1990), p. 9. and p. 27.

[28] This is not completely surprising, considering that, at the time of his writing, Suh was wrapping up his two-year tenure as assistant director for engineering at the National Science Foundation.

to follow Suh's axiomatic approach to the letter[29], products should have no functional interfacing (i.e., each feature of the product has one and only one function to perform) and is very simple, information-wise. Clearly, the most sophisticated development minds (especially in the military) have been breaking these rules! Keeping this theme in mind will be useful when reading the remainder of this thesis.

Michael French (French, 1985) also presented an "engineering technique" treatise, which is less focused on axiomatic lessons, and more focused on specific types of engineering problems. In this regard, his work reads more like a ten chapter brainteaser than a treatise on the general problems of new product development. Nonetheless, some significant engineer-oriented (as opposed to strictly managerial) lessons are forwarded. These include: combinative idea generation; methodological snags and simple, practical approaches to "optimization"; step-by-step paths for insight; looking for affinity between problems; reconsideration of kinematic/elastic design; appropriate life-cycle costing; and basics on the appropriate use of engineering tools.

Both the Suh and French works do offer recaps of modern approaches and technologies for development, outlining advantages and warnings for their application. Perhaps the most important lesson to be gained from either of these works is the same one to be gained from the current thesis: neither managers nor developers can afford to be reductionist in their approach to developing new products, particularly if their end results

---

[29] To be fair to Suh, he developed seven corollaries to help clarify the basic axioms to suit his needs. On the other hand, he states, on p. 51, that the Information Axiom does not necessarily guarantee the ultimate design (which seems to violate the definition of "Axiom"). Further, according to his definition for functional requirement, all FR'a are independent... If not, then they are not FR's! Sounds like circular reasoning...

(for managers, a more effective ***process***; for developers, more effective ***products***) depend upon interfacing with each other.

In another prescriptive work dedicated to the need for faster new product development, Roseneau (1990) considered some tools and techniques. Though oriented around the currently popular "phased approach" to product development[30], this work was interesting for our study because of its attempt to delineate some useful product development management tools and techniques. These included semi-automated project management tools for scheduling (e.g., PERT or CPM oriented software), resource requirement determination (e.g., resource histograms) and costing (e.g., capitalized and expensed charges). As concurs with first-hand experience in using such tools, there are several problems with such tools in practice. The "top ten list", in increasing order of importance, is as follows (numbers 2-10 courtesy of Roseneau, pp. 154-156):

10. Schedules must be kept current to be useful.
9. Credible time estimates for each activity are hard to obtain.
8. The output formats are not flexible enough for one's own needs.
7. Training may be needed for the not-so-software intense members of the project.

---

[30] Phased development (which seems to carry a different pseudonyms in each company it is adopted) is a simple, linear view of the development process. At the completion of each phase (usually, there are 4-5 phases), the prototype is "one step closer" to market. It is a convenient representation for use by executives, looking for some idea of overall development progress. Unfortunately, it is actually an imagined process, which ignores the continual reworking of prototypes throughout the development process. For more information on this process refer to the works of Roseneau (1990), Cooper and Kleinschmidt (1986), Cooper (1983), Butrell (1984), Feldman and Page (1984), Crawford (1983), Buggie (1981), Douglass et al (1978), Merrifield (1977), or McGuire (1973).

6. Most software is limited to single project analysis, rendering company-wide resource requirement estimates a manual exercise.
5. Some tools are used to satisfy procedural requirements, rather than because they are the right tools to use.
4. Projects do not automatically follow pretty software-generated schedules; managers must still manage!
3. Software tools tend to eat the product managers' time
2. Automated tools tend to be a waste of time for small projects.
1. PROJECT MANAGEMENT TOOLS ASSUME UNI-DIRECTIONAL PROCESS PATHS. Actual product development projects are not so simple. (Our Observation)

Other types of tools offered include the now mundane CAD/CAM/CAE/CASE tools, electronic publishing (Roseneau stated that roughly 30% of a product design costs are dedicated to documentation), and a whole host of "Design for Life Cycle" methodologies[31]. Roseneau also presented simulation modeling as a potentially useful tool for developing products. Unfortunately, such modeling has been primarily limited to modeling the ***product***, not the very ***development process*** itself. In this study, we have attempted to make some inroads into this area.

---

[31] The so-called Design for Life Cycle Manufacturing or Design for Manufacturing (DFM) methodologies include a host of policies, practices, and attitudes with the goal of optimizing manufacturing cost, product quality, reliability and service capability. Some specific subsets of the DFM strategies include: axiomatic design; functional elimination, simplication and standardization; reduction of information complexity; "process-driven" design; design for quality; design for change; design for flexible manufacturing, design for analysis; design for assembly; DFx tools (specifically designing around tool capability); producibility measurement; and the statndard DFM toolkit, which comprises roughly a dozen pre-packaged methodologies, most of which are currently avaialble from vendors. For a review of these methods, refer to Ettle and Stoll (1990).

Though it is tempting for a reviewer of the field to be pleased with the clarity and insight of authors' contributions[32], it is apparent that authors carry many similar beliefs, yet are frustrated by the slow progress of firms to follow their suggestions. Upon further review, it becomes apparent that there is ***little depth*** of research in new product development. This is especially evident when one considers the propensity for researchers to write reflective, philosophical books on the topic, rather than truly scientific journal articles. Based upon the paradigm of science formation forwarded by Kuhn (Kuhn, 1962), the study of new product development may be considered only a "pre-science". Eventually, we hope, this field will carry more characteristics of full sciences. This, however, cannot be expected until much more is learned about the basics of new product development.

It is with this in mind that we commissioned the study in these pages. This does not, by any means, imply that studies to date have been a waste. Rather, we have tried to look beyond the hoopla of current day-buzzwords (which often carry the same meaning as different buzzwords of several generations past), to see some of the underlying dynamics of how products ***are actually developed***. Only after obtaining a better understanding of ***how*** the processes really work, and ***why*** they operate as they do (i.e., gain better in-firm diagnoses), may we better prescribe appropriate remedies.

---

[32] Of all the disciplines considered in this study, the product development literature seemed to be the easiest to read. This may be attributed to the fact that many works in this field have been written by management school professors for corporate executives. As a result, the content of the articles tend to be simplistic, written in easy to understand language, with little backgrounding necessary on the reader's part.

2.3. The Cognitive Sciences (Limits on our Perceptions)

There has been a "minor hitch" to the many technical solutions to the problems of new product development; it is a perennial obstacle which is often considered to be only temporary in nature, merely to be overcome through the passage of time (or indoctrination of the new techniques); as a result, it is a problem which many researchers and consultants have either cleverly sidestepped, blatantly ignored, or turned into yet another profit center. Nonetheless, it never seems to abate. The barrier? ***Product development is a PEOPLE process, not a technical process.***

There can be little dispute that the capabilities of human beings to grasp the intent and inticracies of each other's opinions has major bearing on their performance in development organizations. The exact nature of these capabilities, however, seems to be an exceptionally difficult problem to conquer. Further, the capability of managers to improve upon the status quo seems to rest upon some understanding of the prevailing conditions, both before and after the "improvement". Whether we like it or not, there is a recurring problem of communication breakdowns in new product development, as well as in other business processes. This phenomena is acute in organizations which have adopted sophisticated information systems, as well. With this observation in mind, it seems appropriate to consider some observations from a set of fields seemingly independent of new product development. These fields are known individually as ***Artificial Intelligence, Anthropology, Linguistics, Neuroscience, Philosophy, and Psychology***. Collectively they are known as ***Cognitive Science***.

Our foray into cognitive science has two purposes. First, there are some *specific findings* in these disciplines which we shall find applicable to new product development. Second, it is apparent that new product development is increasingly taking on many of the characteristics of cognitive science *as a field.*

The birth of cognitive science has been identified (see, for instance, Newell and Simon (1972), Miller (1979), Mandler (1981), Bruner (1983), Gardner (1987)) as occurring on September 11, 1956. This was the second day of the three day Symposium of Information Theory at MIT. The reason for this particular date was that four of the speakers of that day went on to become leading figures in this newly developing science. These speakers were Allen Newell, Herbert Simon, Noam Chomsky, and George Miller. These speakers presented new, significant changes in direction for their respective fields. Their contributions of the day are summarized as follows:

- Newell and Simon outlined their "Logic Theory Machine": a device that could, for the first time, prove a mathematical theorem on its own. They had expanded upon the modern framework for Artificial Intelligence, previously conjectured by Turing (1936).

- Chomsky, in his "Three Models of Language" presentation, demonstrated his own approach to grammar, based upon linguistic transformations. This mathematical approach to language construction was a revolution in the world of linguistics.

- Miller, from the world of psychology, delivered his claim that human short-term memory had a limited capacity of approximately 7 items. His paper, "The Magical Number Seven, Plus or Minus Two..." (1956) went on to become one of the most influential works in the cognitive psychology literature.

Thus, at this single forum, three (***Artificial Intelligence, Linguistics,*** and ***Psychology***) of the six major disciplines of cognitive science began to "cross-fertilize" their findings. Soon, ***Neuroscience, Philosophy,*** and ***Anthropology*** would enter the scene. Their cause: to help define the nature, components, sources, development, and deployment of knowledge as influenced by the human brain. This was to be the genesis of "cognitive science."[33]

According to Gardner (1987), there are five key features present in the majority of studies considered within the domain of cognitive science: ***representations***, ***computers***, ***de-emphasis on immediate practicality***, belief in the rewards of ***interdisciplinary studies***, and a deep-rooted ***emphasis on philosophical issues***. Aside from the non-trivial feature of practicality, these attributes parallel much of the new product development literature and new product development practices. As we see further in Chapter IV, even the need for practicality is sometimes unheeded in both analysis and implementation of new product development. In the fields of cognitive science, these five features manifest themselves as follows:

- ***Representations*** are those abstract mechanisms by which human cognition is described. They may take the form of symbols, schemas, images, ideas, etc. Cognitive scientists generally accept that mental processes are represented in the nervous system. Notably, neuroscientists have the least enthusiasm for these "representational accounts." Psychologist, linguists, and computer scientists (particularly those associated with Artificial Intelligence) have shown the most enthusiasm for these representational accounts.

---

[33] Gardner (1987), p. 6.

- Because Artificial Intelligence is thought by some to be the central discipline of cognitive science, ***computers*** (AI's dominant medium of expression) often play a central role. In much of cognitive science, computers serve as *the* model for human thought. Computer involvement is a reliable gauge of a discipline's involvement in "cognitive science" as a useful discipline. Skepticism for the use of computers often leads to skepticism about cognitive science. Anthropologists and neuroscientists don't think of computers as viable cognitive models. Linguists and psychologists have been quietly underwhelmed by their usefulness as cognitive models.

- Cognitive science has been characterized as having ***few concerns over practicality***. In the sense that it is being thought of as a science, concerns of implementation should be the duty of another discipline. In contrast with this view, it seems more reasonable to contend that such concerns are essential to a more complete description of human experience, and thus shouldn't be ignored. As we shall see later (in chapter 5), there is an unfortunately strong parallel of the former view (isolation from reality) in some management of new product development.

- With many cognitive scientists lies a faith that insight and results from their studies will not be limited to the domain of their particular discipline, but rather from ***interactions between disciplines***. This aspect of cognitive science maps well to the observations that development personnel must often interact with specialists from outside their field.

- Finally, cognitive science perpetually addresses ***issues which have deep-rooted (and classical) philosophical histories***. Revealing here is the key role of Philosophy in cognitive science: to supply fundamental issues for analysis and to judge how well such issues have been analyzed. In the field of new product development, this role is carried by some management consultants, for the scientific contribution to charting new directions has been minimal. Given the auspicious role of consultants, however, is it any wonder that the field of new product development has been so directionally flighty?

Each of the fields within cognitive science has some basis in assisting our understanding of the nature and processes of the human brain. Another common theme of each of these disciplines is a propensity for controversy. Such controversy provides a useful medium for assessing the state of each field, particularly as such controversy has shifted the orientation, or paradigm, of these fields.

Neuroscience

Karl Lashley (1929) set the stage for holism and, later, plasticity. His "holistic" approach to brain behavior was in direct conflict with the predominant "localizer" views of the time. Such predominant views had been developed by European researchers such as Fritsch and Hitzig (1870). Support for Lashley's views was shown by Huglings Jackson (1932) and much later by Pribram (1971), Hooper (1982), and Harth (1982). The discipline has swung from one point of view to the other several times, with various "intermediate" explanations of brain functioning, based on experimentation by Weiss (1952), Sperry and Miner (1955) and Pribram (1971) and non-empirical views such as Hebb (1949). More recently, the pendulum has swung back towards the localistic points of view, albeit with some corollary holistic explanations. With the recent explosion of organic chemistry onto the neurological analysis scene, there has been more understanding that brain behavior is a highly integrative process of locally responsive chemical reactions. Yet, there still remains a major problem with actual integration of such knowledge, due to very tight disciplinary focus among researchers. This may be

related to the fact that neuroscience is still in the process of discovering and cataloging knowledge, not explaining or resolving pre-ordained paradigms[34].

Anthropology

Before the turn of the century, Lucien Lévi-Bruhl started with a unique distinction and description of the "primitive mind", reversed his views (claiming that all human thought processes are the same), only to have the field re-address the distinctive thought processes of individual cultures (Cazeneuve (1972)). Edward Tylor (1871) had attempted to bring unprecedented methodology to the newly developing science, forwarding definitions of ***culture***, ***schemes of survival***, and ***adhesion***, as well as introducing statistics (in the form of correlation) to the field. It may be argued that this had the effect of transforming anthropology from a traditional mode of speculation and generalization to a modern mode of empirically-based understanding of specific cultures.

Throughout, anthropology has had the dubious distinction as a fickle field, not sure how to characterize ***culture***, despite ever increasing levels of scientific sophistication. To some, the definition of culture has been an unstable linchpin of anthropology, which has stifled its growth. Again, we have seen two classes of studies:

- those which tackle broad questions with low levels of scientific rigor (e.g., Lévi-Strauss, 1963, 1964, 1969);
- those which use precise analytical methods in restrictive domains (e.g., the computational analysts, such as Wallace and Atkins, 1960).

---

[34] Per discussions with Dr. Jeff Mulchahey, Emory University's Neurobiological Laboratory, 1994-1995.

The latter class, often summarized as a sub-field within anthropology--ethnoscience--, had risen with great promise in the 1950's, only to fall from grace as inconsistent results obviated useful, practical generalizations. Though there have been glimmers of hope in anthropology through the ages, there still remains some need for scientific crystallization. The similarities between this condition and the status of new product development as a field is remarkable.

Artificial Intelligence

AI has been filled with controversy. Newell (1983) found over 30 different issues which have divided the field. Four issues are discussed here. Within AI, a quickly formed and long lasting distinction arose between "weak" AI (where programs are developed as a means of testing theories of human thought processes) and "strong" AI (where the developed programs are considered minds in their own right). This controversy has been coupled with the polarity of the "generalist" and the "expert" camps. The "generalists" have looked to develop programs or sets of programs that can cover a broad range of disciplines. This group was anchored by Newell and Simon, with their General Problem Solver (1972) (and earlier Logic Theorist) program. The "experts" have developed programs which use considerably detailed knowledge, but are narrow in their application. This camp has been led by Feigenbaum et al (1971) and their DENDRAL program. Winograd's pivotal doctoral work (Winograd, 1972), describing the SHRDLU program, seems to have set the AI pendulum swinging in the direction of the "experts."

The battle between those who distinguish declarative and procedural types of representation was one which has subdued in the past decade. Declarative representation is the coding mechanism whereby knowledge is stored as a set of facts, or declarations.

This representation was generally adopted by LISP type programmers, such as McCarthy et al (1962) and Foster (1967) (actually, they were developers of LISP), because of its ease of understanding and economical storage. Procedural representation is a coding mechanism whereby knowledge is stored or imbedded within sets of procedures. This type of representation mimics natural human actions better than declarative representation and allows interactions across domains. Its proponents included Boden (1977), Cohen (1977), Newell (1983), and Winograd (1975). More recently, researchers have accepted *both* types of representation, as evidenced by knowledge representational languages, which use both (see Bobrow and Winograd, 1977).

Another major controversy in Artificial Intelligence has been the increasing use of "top-down" examinations of cognition and language. Some AI researchers have proposed script structures of language (for instance, Schank (1972), Schank and Abelson (1977)). Minski (1975) proposed the use of frames to allow learning, and forwarded his "society of minds" view (1979, 1982), in which every mental activity consists of multiple specialists, or agents. Such top-down approaches have called out for different methods of computing: away from the digitally serial von Neumann computer and towards parallel processing. Feigenbaum and McCorduck (1983) anticipated that the next major breakthroughs in AI would come from such newer style devices. This has yet to manifest itself, though recent forays into frame representation in the emerging sub-field of virtual reality must surely consider this.

As we discovered in our process documentation efforts in this study, there can be a stark contrast in both methodology and analysis results when analyzing new product development from either top-down or bottom-up approaches.

Linguistics

Of the cognitive sciences, Linguistics has possibly undergone the biggest change in the past 40 years. During the "Chomsky Revolution," the field was pulled away from the structural linguists' view and towards transformational grammar. It was much the work of Noam Chomsky, who's *Syntactic Structures* (1957) (no publisher would print his controversial 1955 dissertation, "The Logical Structures of Linguistic Theory" until 1975) called attention to and built upon many of the issues raised earlier by Sapir (1921) and Harris (1952). In short time, he proved the fallacy of (then theoretically plausible) finite-state grammars and demonstrated the inherent difficulties with phase-structure grammars. The new transformational grammar he initially proposed focused on syntax, and excluded many semantic and phonologic factors. In 1958, at the Third Texas Conference on Linguistic Analysis in English, he took on the structural linguists and proved to them why their theories were wrong. Later, Chomsky (1965) proposed his "standard theory," in which he reformulated his views of sentence formation into a two-layer structure.

Nevertheless, there has been a continually running debate over the mechanisms of thought as it relates to language. In brief, the conflicting questions are as follows:

- Does thought (or cognition) determine our linguistic content?

or

- Does our linguistic capability determine our thought process?

Though there are stalwarts on either side of this fence, there seems to be an emerging concession that both of these suggested phenomena occur in practice. Linguistics is an

excellent example of a field where abstract modeling has, at times, over-ridden the realities of empirical findings. Reconciliation of empiricism with theory seems to be forthcoming, however, as related by Gardner (1987).

The abstract-pragmatic controversy has yet to really begin in new product development. As we discussed earlier, the field is dominated with practical, experiential theory. Coupled with excess localism, such experienced-based theory can be expected to produce sub-optimization "traps." Through abstract representations, or modeling, we may enter a new era of thinking off the page of existing experience, and discover new developmental methods of ideas. Ultimately, however, any successful modeling approach will have to have basis in reality (even if not currently a "popular" representation). Hence, our extensive field research prior to the development of the CPP modeling structure presented in Chapter V.

Psychology

Though the beginnings of psychology as a science is normally placed to the late 19th century experiments of Wilhelm Wundt and his students, there have been many "scientific" inquiries about the nature of human thought mechanisms over the past two millennia. In his comprehensive compendium of the field, Morton Hunt (Hunt, 1993) relays perhaps the first recorded psychological experiment, conducted by Psamtik I, King of Egypt in the late 7th century B.C. The experiment was actually a linguistic endeavor to prove that Egyptian was the fundamental language of mankind. To identify the "inborn" language of the species, he isolated two infants from society until they "spoke" on their own. By current criteria, it was a failure on two counts: its results (that Phrygian, not Egyptian, was the fundamental language) do not match the results of modern studies

(which indicate that there exists no innate language) and its hypothesis was incorrectly dependent upon the assumption that a baby's first sounds are necessarily a language. Nevertheless, it was an attempt at learning about the workings of the mind through some scientific methods.

Since such ancient days, psychology has undergone many transformations. Up to the days of Wundt, the field had been driven by the conjecturism (idealism & realism) of the ancient Greeks, the conservative deliberatism of scholars through the middle ages, protopsychology (consisting of rationalism, empiricism, and nativism) throughout the 17th and 18th centuries, and the pseudo-scientific physicalists (e.g., magician healers, skull readers, mechanists, specific energy advocates, etc.) of the 18th and 19th century. From the latter group, despite their emphasis on currently questionable healing capabilities, came some important realizations. Ernst Heinrich Weber (1795-1878) work on "just noticeable differences," and the sensory law which carries his name, were among the first quantitative contributions towards understanding the relationship between the physical world and the mental world. Johannes Müller (1801-1858), a proponent of specific nerve energy, deduced that our perceptions are not replicas of, but analogues or isomorphs of, the objects around us. Hermann von Helmholtz (1821-1894) built a strong bridge between the fields of physics and neurological functions; his work on nerve transmission, color perception, hearing, and spatial visualization have had a profound effect on psychology as an empirical research area throughout the twentieth century. One of Weber's students, Gustav Fechner (1801-1887) expanded the concept of just noticeable differences to account for the relationship between stimulation intensity and sensory intensity. In attempting to confirm that relationship, he developed a new experimental method (the method of adjustment) and refined two existing methods (the method of

constant stimuli and the method of limits). Though his findings have since been discounted, his refined methodologies remain fundamental to sensory measurement[35].

Upon the formalization of modern psychology as a science, marked by Wundt and his students, the field emerged as a discipline of its own, not mere intellectual appendages of philosophy, physics, or physiology. Wundt's methods of introspection were a throwback to the pre-mechanistic era of analyzing conscious mental processes. This contrasted with the predominantly experiment-oriented physical analyses of neural response. Yet, Wundt seemed to believe that introspection could be utilized as an experimental tool to better understand mental processes.

As a contemporary of Wundt, it was remarkable how different (and, yet still similar in many ways) William James' (1842-1910) approach to psychology could be. He abhorred the notion that psychology would be considered a science. Yet his *Principles of Psychology* (James, 1890) had a lasting effect on the field as a science over the next sixty years[36]. His "functionalist" view sought to investigate the totality of mental activities under real-life conditions, rather than draw conclusions from isolated introspections. In this manner, James could be considered more holistic than Wundt. Yet, James was an experimental psychologist, and is given credit for introducing and developing the field in America. His concept of free-will, as a conscious process that directs one's actions, is an interesting example of how an original concept could flourish for some years (particularly while the originator was still alive), become submerged during the reign of new attitude

---

[35] Hunt (1993), pp.109-126.

[36] Hunt (1992), p. 153.

(in this case, behaviorism), then re-emerge as an original thought, albeit under a different name (currently this concept may be known under a number of existing terms--"self-control", "intentionality", "purposive behavior", etc... but "will" is mysteriously absent from current psychology vocabulary). Though James' influence on the field was significant, it did not have the solid following during his life that Wundt experienced.[37]

The rapid emergence of behaviorism in the 1920's might be credited to its intensive orientation around experimentation, especially with regard to stimulus-response (SR). The prolific[38] researcher Edward Thorndike (1874-1947) seems to be the turning point of this era. Most are well aware of the Pavlov (1849-1936) experiments, which identified various aspects of conditioned responses with dogs. John B. Watson (1878-1958) might be considered behaviorism's biggest salesman, and carried his knowledge of the field into the business arena, as a resident psychologist for the J. Walter Thompson ad agency. Like other behaviorists[39], he believed that almost all human behavior was a result of SR conditioning.

This prevailing thought was not to last, however, as the cognitive sciences began to coalesce. Interestingly enough, the development of cognitive science has not reduced the impact of psychology (even, to some extent, behaviorism), but rather fostered the creation of autonomous specialty fields *within* psychology. According to Hunt (1993), there are

---

[37]Hunt (1993), p. 161.

[38] During his long career as a psychologist at Columbia, he wrote fifty books and 450 articles. Despite this, he is probably most remembered for his graduate research on the instinctive behavior of chickens. See Hunt (1993), p. 246.

[39] Other behaviorists of note were B.F. Skinner (1904-1990), Clark Hull (1884-1952).

now 58 sub-fields of psychology. For an individual looking to apply findings from the "field" of psychology to their area, there are clearly many places one must look.

The 1950's witnessed a rise in cognitive psychology, with Cherry (1953), Broadbent (1954), Bruner (1956), and George Miller (1956) leading the way in the studies of mental processing and short-term memory capabilities. I have found John Anderson's work on the architecture of cognition (Anderson, 1983) to be an interesting attempt to unify the works of cognitive psychology. Anderson has pointed out a very important consideration, which directly impacts decision-making processes in development organizations (in fact, any organization): humans, as part of the memory/thinking process, incorporate the context of their knowledge or information, in varying degrees.[40]

Dan Russell and Warren Jones (Russell and Jones, 1980) describe what amounts to a conformity bias: we tend to notice and store experiences which support our strongly-held beliefs, and tend to ignore or discount those which do not.

The currently fashionable model for the human thinking process is ***connectionism***. This theory, derived from physical brain research, contends that knowledge is not stored via the states of neurons, but rather according to connections among neurons. Thus, it is the structure of the brain, as a processor, not merely the existence of its components, that determine the course of mental processing. Interleaved with this view is the concept of Parallel Distributed Processing (PDP), which states that the brain is not a serial processor, cannot perform any particular activity very quickly, but can perform many activities

---

[40] Hunt (1993), p. 522.

simultaneously. As David Rumelhart and Jay McClelland (Rumelhart & McClelland, 1986) have illustrated, this processing structure may be a complex (non-linear) array of connections, each of which may be exciting (amplifying) or inhibiting (insulating) connections. By changing the structure of such connections, vast changes in mental output may occur. This can be contrasted with the simple model of a single serial- process for each output, a model for which computer representations of "thinking" have been largely based on. Some of the most revered researchers in the field of brain structure and behavior endorse this approach[41].

For our purposes, such work in this discipline have direct relevance to the study of new product development (NPD). As we shall discuss in our review of field findings, inaccurate cognitive appraisals are a way of life in development organizations. Further, we see that the human-intensive structure of development carries many analogues to the dynamic PDP structure. We are excited about further developments in this area, for if we can better understand the dynamic structure of the brain's processes, we may learn even more about how to manage the dynamic structure of NPD processes. With further analysis of NPD processes, perhaps we can also provide some insight to the researchers of brain behavior.

Philosophy

Jerry Fodor (1971, 1981, 1983) seems to have led a contingent of researchers away from their focus on empiricism, as developed by Hume and others over the past three hundred

---

[41] See, for instance, Newell (1990) and Crick & Koch (1990).

years. In particular, Fodor has called for a re-examination of the views of Descartes, particularly as the Cartesian views acknowledged mental states and the causality effects of mental "events." His intent seems to be the reincorporation of those older views rather than abandonment of the empiricist notion. As perhaps the oldest, most inexact science, philosophy seems to carry deeper lessons about our existence than any single field we may wish to undertake. In this regard, philosophy carries the dubious honor of loosely self-evaluating and directing our forays into other fields[42].

Regardless of their primary disciplines, it seems certain that scientists engage in philosophical thinking to do exactly that. As we discuss further at the end of this chapter, some highly regarded scientists seem to have been willing to address (or re-address) issues which are considered among the more humble to be "untouchable."

---

[42] Gardner (1987), pp. 86-88.

2.4. Mathematics (reality from the abstract?)

It should not be a surprise to see a certain degree of mathematics in this form of study. Although the organizational theory literature is not calculus intensive, the production, systems analysis and operations research disciplines have become quite adept at creating abstract mathematical representations to help solve problems of interest. Because of the integrative nature of this research, however, it has proven to be difficult to develop a traditional, calculus-based mathematical model for how a development organization operates. There are so many different aspects of development to consider that a mathematical treatise, though contemplated briefly (and partially described in Chapter 6), would quickly become highly complicated and extremely difficult to solve. Based upon the observations of just the first two field sites, it was apparent that any mathematical model would require simultaneously discrete and continuous functions. Though certain regimes of these types of problems are solvable, the majority of such problem formulations can easily eclipse current solving capability.

This does not remove mathematics from this analysis, however. Although specific equations have not been developed or solved, there is still some value in considering simple-case scenarios to gain some insight into this intricate web of innovation and new product development. There are upper and lower limits (to human cognition, for instance) which can help us bound our problems, if not solve them directly. By observing the behavior of physical objects, we can see how it is possible for us, as locally "rational" human beings, to engage in behavior which, in retrospect, seems highly irrational.

Perhaps the most appealing (and most entertaining!) branch of mathematics research that has relevance to our topic is ***game theory***. This field was developed by John von Neumann and Oskar Morgenstern (von Neumann and Morgenstern, 1944) as a formal examination of conflict situations in economic development. Since this time, concepts such as zero-sum games, constant-sum games, prisoner's dilemma, duopoly analysis, and saddle points have become common language among economists, political and military strategists, public policy-makers, psychologists and other decision-scientists. Yet, the engineering fields (and engineering managers, in particular), have been slow to adopt the most elementary game theoretic concepts developed over 50 years ago.

This is particularly surprising when one considers that Von Neumann is considered by many to be the "father" of the modern (stored-memory) computer[43], a device which many engineering managers readily accept as a useful, often necessary, tool of their trade. It seems reasonable to suspect that the development of game theory and the development of the computer were non-coincidental events which could provide key links to the understanding of human behavior. (Recall that the initial focus was on economic behavior, analysis of which has utilized game theory and computerization in a big way.) Since innovation and product development are predominantly composed of cognitively intense decision-making activities, any method which can assist in understanding this cognition (specifically, understanding of one's own decision alternatives) should provide great assistance in improving the cost, quality, and timelines of product development

---

[43] As is unfortunately typical in developments of just about any new and significant idea or device, great controversy arose over the intellectual property value of the design of the modern computer. Other scientists that laid claim were Vincent Atanasoff, Wallace Eckert, and John Mauchly, the latter two being the originators of ENIAC. For a very interesting discussion about the history of the pre-PC computer, refer to Shurkin (1984).

activities. The currently untapped potential in the field of game theory (especially the formulation, deciphering, and execution of "strategies") may provide us with better analysis and integration capabilities.

Although game theory has been presented repeatedly in the economics literature as a useful descriptor of behavior, most such studies have been oriented around decision-sciences, policy development, and market strategy. As we alluded to earlier, Baumol (1990) contemplated that innovation is destined to proceed in any society; whether such innovation is productive to the society, however, depends upon the "rules of the game" which are established for the innovator to work within. Unfortunately, formalization of an "innovation rules" framework has not been developed.

In as much as innovation and innovative product development are driven by competition for technological advantage (this study contemplates that this is not necessarily the case), Kamien and Schwartz (1982) contemplated the interaction between market competitors as a driving force for each player's R&D strategy. Specifically, they suggested that the R&D expenditure (or publicity) of one player could, under certain conditions, affect the R&D budgets of competing firms. Game theoretic examinations of how to better proceed within one's own development organization, however, have not been apparent.

For a good review of some specific game theory concepts, refer to Jones (1980). An interesting history of the development of this field has recently been presented by Weintraub (1993).

Despite the success and further promise of game theory to our topic, however, a more recent contribution to science from the mathematics literature may prove to be the most insightful and, simultaneously, the most confusion-ridden. For all the advances made in the various disciplines presented thus far, perhaps none has transfixed a wider range of researchers in the past decade than the mathematics of ***deterministic chaos***. In its most elementary form, deterministic chaos can be defined as system behavior that depends so sensitively on the precise initial conditions that it cannot be distinguished from that of a randomly behaving system[44]. For our purposes, we consider chaotic behavior as any behavior that we cannot adequately predict, even though we know it is not technically random nor stochastic.

The concept of chaos is not, by any reasonable time measure, a new one. In ancient Greece, chaos was considered as a description of the primeval "empty" state which existed prior to the development of the universe as we now see it. As a nondescript, formless mass, chaos has often been associated with evil; contrast this with order, which has been considered a symbol of goodness. It may be contemplated that such good-evil connotations arose from the uneasiness and frustrations associated with attempting to understand such less-well-ordered phenomena. Paradigms, or theories, have been developed through the ages in a semi-continuous effort to describe observed phenomena. When "well-established" theories were subsequently violated in nature, it was often asserted that such indescribable acts were the work of demons. The evidence that chaos has long been associated with any highly misunderstood phenomena may be seen in the

---

[44] At the prestigious international conference on chaos, held by the Royal Society of London in 1986, a dictionary-ready definition of chaos was, in a surprisingly awkward fashion (considering the participants), forwarded as: "Stochastic behavior occurring in a deterministic system." From Stewart (1991), p. 17.

effort of the Dutch chemist Jean-Baptiste Van Helmont, in 1632, to describe that state when a material is neither solid nor liquid: His invention of the term "gas" carries deliberate similarity to the ancient old word chaos. One who understands the physics of gases, wherein lies the meeting of determinism and randomness, immediately understands the appropriateness of this term.[45]

As characterized by Baker and Gollub (1991), one can gather the character of chaotic dynamics by imagining that a system can be started twice, each time with slightly different initial conditions. If one considers the difference in initial conditions to be exceptionally small, as might result from error in measurement, one would expect the two systems to be incrementally out of phase with each other, but still behave in approximately the same fashion. Such is the nature of a ***non-chaotic*** system; the "error" in predictability grows linear with time. For a ***chaotic*** system, on the other hand, the error grows exponentially in time, rendering predictability all but impossible in very little time. This special situation, which can only occur when the governing drivers (or equations) for the system are non-linear, is known as ***sensitivity to initial conditions***. As a formal mathematical concept, this phenomenon was discovered by the mathematician/ astronomer Henri Poincare in 1913.

---

[45] Based upon Stewart (1991), p. 52-53. It is interesting to note that roughly 100 years earlier, the "miracle physician" Paracelsus (1493-1541) coined the term chaos to the odorous "essence" or "spirits" which diffused from the human body after death or during surgery. Prior to this time, gaseous-type substances were referred to as "sulfurs". Paracelsus, of course, is probably best known for his early contributions to medicine-making, the earliest forms of chemotherapy. To some historians, he is considered the first modern physician. (Schuman, 1951, p. 122-123)

The science of chaos, as a distinct mathematical concept, has become increasingly popular in the past ten years; today we see theories of chaos being used in a diverse variety of fields. Just a brief sampling of applicable fields and some of their predominant authors include the following:

- ***acoustics*** (Lautenborn and Cramer, 1981)
- ***astronomy*** (Buchler Perdang, and Spiegel, 1985)
- ***biology*** (Colding-Jorgensen, 1983)
- ***chemistry*** (see particle physics, below)
- ***finance*** (van der Ploeg, 1985; Brock, 1986, 1990; Brock and Sayers, 1988))
- ***fluid dynamics*** (Gollub and Bensen, 1980; Swinney and Gollub, 1986; Heslot, Castaing, and Libchaber, 1987)
- ***international trade*** (Lorenz, 1987),
- ***inventory control*** (Mosekilde et al, 1990)
- ***literature*** (Hayles, 1990, 1991)
- ***macroeconomics*** (Day, 1982; Chiarella, 1986; Routh, 1989; Rosser, 1990)
- ***microeconomics*** (Rasmussen & Mosekilde, 1988)
- ***medicine*** (Olsen and Degn, 1977; Jensen et al., 1986)
- ***particle physics*** (too many to list!)
- ***physiology*** (Goldberger & Rigney, 1988)
- ***sociology*** (Young, 1991)

The recent explosion of popularity of chaos theory may be asserted to be the result of the easy-reading compendium forwarded by James Gleick (Gleick, 1987). In his review of the development of the science of chaos, Gleick introduced a number of practical examples in which chaos was clearly evident. As relayed by Gleick, chaos is no longer a remote, abstract oddity of computer-intensive mathematics, nor does it suffice as a generic description for randomness or misunderstanding. Since this time, literally

thousands of studies have been published which consider chaotic affects in their experiments. Contrast this with the estimate that the *cumulation* of chaos-related studies before 1985 numbered perhaps a hundred.

This may seem to be a mere coincidence, and may very well be. However, consider that the modern day (re)discovery of chaos was made by Ed Lorenz over 30 years ago (Lorenz, 1963). As a meteorologist with a mathematics background, he modeled a simple dynamic weather system. Using three simple differential equations, which were based upon the past works of Henri Benard, Lord Rayleigh, and B. Saltzman, and his simple computer, he simulated the longer-term behavior of such a system. What he found perplexed him: small changes in equation coefficients produced startling large differences in the behavior of the system. At first, he contemplated that such behaviors were due to round-off error. Yet, subsequent reduction of such error only "corrected" the behavior by a small degree. Regardless of the number of significant digits used in the calculations, the system remained highly divergent. Regardless of the significance of his finding, his published paper languished in obscurity for a decade. Meteorologists did not seem to understand, nor care much for, his findings; mathematicians didn't read his work, for it was published in the *Journal of Atmospheric Sciences*--not common reading for mathematicians who understood such phenomena. Gleick's work popularized Lorenz' work (and that of several others) and, it seems, accelerated the slow pace of interdisciplinary communications on the topic of non-linear dynamics.

Certainly, there have been many mathematicians who have been closely involved with the concepts of chaos, and its mathematical antecedent, non-linear dynamics. These seem to have been closed societies, however, with little cross-fertilization among disciplines.

For instance, Gleick reported that there was a definite rift between mathematicians and physicists (who use many similar mathematical principles) in the period from the 1930's to the late 1960's. This was, presumably, because of different philosophies about the "appropriate practice", and the true source (math or physics) of mathematical topology. If such fields as mathematics and physics could not see eye to eye, how could one expect an obscure meteorology-related article to be heeded?

The very nature of chaos theory as a study of the dynamics of certain non-linear systems cannot ignore the impact of the computer as an analytical tool over the past 30 years. Because most systems of non-linear differential equations are either extremely difficult or impossible to solve analytically, it was only until the advent of (and necessarily, the cultural adoption of) the computer that one could reasonably use numerical methods (i.e., "plug and chug") to understand their nature. Thus, the existence of the computer enabled a growth in the study of non-linear dynamics and, not coincidentally, a growth in awareness and understanding of chaotic dynamics as a subset of non-linear dynamics.

Not all dynamic systems are chaotic. As alluded to above, a chaotic system must first be non-linear. This means that there must be an intercoupling relationship between at least two variables, such that the dynamic scaling of the two variables does not remain constant. Yet, just being non-linear does not suffice. The system must be composed of at least three variables. This three-plus dimensional requirement was actually a bit of a surprise when discovered, for it had been felt that very strange behavior was limited to problems of only many more variables. The three variable-plus requirement is a reflection of the chaotic "need" for dimensional space to permit divergence of trajectories,

confinement of motion to a finite region of phase space, and unique (non-overlapping) trajectories.

Principles from this field which seem to have some potential bearing on new product development include ***cyclicity of process, noise, dynamic structures,*** and ***fractals.*** While all of these may ultimately be considered important from a managerial point of view, we, from a sceintific understanding perspective, shall focus on the first principle, cyclicity.

As we have found in the field, new product development can be, in an ex post facto fashion, illustrated as a circular process. When such activities cease to iterate, one may say that the innovation process within an organization has died. If one is examining a closed-end process, then such "passing away" of certain functions is desirable. For an organization which wishes to effectively continue its activities, however, such death is intolerable. For innovation, this is the very basis of the Schumpeterian business cycle. In the current thesis, we attempt to ascertain the nature of the cyclicity which may occur within a development process (i.e., cycles within cycles). Depending upon the specific nature of organizations in this regard, this smacks of the need to consider organizational dynamics as a set of difference (in the discrete case) and/or differential (continuous case) equations. Even if an organization can only be represented in a qualitative form, such differential perspective may provide new insight into the true dynamic behavior of development organizations. If organizations behave in chaotic fashions, then we may gain even better insight into the underlying drivers (i.e., NL equations) of the development process.

Engineers have been increasingly aware of the ramifications of chaotic (or "chaotic-looking") functions in their technical analyses of physical structures, circuits, acoustics, and so forth. Yet, engineering managers have not considered such effects as having any resemblance or relevance to their ***organizational*** behavior. This study is the first and only study known to consider new product development processes as non-linear, leaving the door open for follow-up chaos studies on the topic.

2.5. The Management of Science

This research is an integrative examination of the processes of product development, specifically ***new*** (and sometimes innovative) ***product development.*** This integration effort has turned over several "stones", some of which certain scientists will characterize as sacrilegious to their specific causes. This is a direct, and expected, result of conducting non-reductionist research in a field indoctrinated with reductionist paradigms. It has been interesting, and admittedly a little discouraging at first, to witness this phenomenon already. It has been even more interesting, however, to see this has been a hallmark of significant new research for thousands of years.

Consider the fate of the edict of Marquis de Laplace, the famous nineteenth century astronomer. In effect, he asserted that the universe was completely deterministic, if only we understood all of nature's physical principles and knew the complete state of the universe at any one time. This concept was strongly resisted by many in the scientific and devout community. Yet, it had seemed in full accordance with recently developed scientific theories of Newton, Galileo, and others. Eventually, and subsequently throughout the remainder of the nineteenth century, scientific determinism became an accepted, standard assumption in many, diverse areas of scientific research. A complete turnaround. Then, with the advent of quantum mechanics, and Heisenberg's observations and resulting theory regarding uncertainty, the deterministic bent of the scientific world declined. Determinism became the mark of the "old way" of looking at phenomena.

Through the twentieth century, it has become more *in vogue* to consider that variance as a way of life, that the universe is a giant random mixing bowl, only governed by our

theories as far as they remained statistically valid. If repeated large-scale iterations of equations produced non-uniform results, then it was because of round-off error. If certain genetic mutations persisted, it was because they were just lucky to have survived the high odds against them. If a man has avoided a car accident his entire life, then it is only a matter of time... To a great extent, we seem to have turned Heisenberg's findings around: from an *admittance* of our incapabilities to see to a *justification* of our assertions that we wish to see, regardless of the evidence. Witness, for instance, how long it took for the results of Lorenz's weather model to be seen as more than just computational round-off error.

In a humorous (yet quite serious) account of our environmentally developed analysis blinders (Ackoff, 1978), it quickly becomes obvious that paradigm-locking affects all of us, to a greater or lesser extent. Unfortunately, it is highly uncomfortable for most of us to challenge such paradigms in our daily lives. We are surrounded by others who carry their own mental version of the way the world works. If any one of us, regardless of who we are, significantly threatens the delicate balance between comfortable tradition and our individual views of the world, we become "off-the wall heretics." As recounted in the works of Kuhn (1962) and Bauer (1992), such problems have affected even the greatest thinkers in history. It has happened so often in the history of scientific development, that it is difficult to ignore: ***truly significant changes in science may take a lifetime or more to prevail.***

For instance, even Albert Einstein refused to believe in the realities of quantum mechanics, despite the fact that he played a significant role in its development[46]. As he wrote to Max Born, *"You believe in a God who plays dice, and I in complete law and order."*[47] He was demonstrating his orientation around a paradigm of classical mechanics which had no provision for quantum indeterminacy.

Some authors seem to have resigned that there will be unavoidable limits which constrain us from ever really knowing the truth (Casti, 1990; Ayer, 1956). Further, we may have great difficulty as communicators to completely transfer our thoughts with precision[48].

---

[46] Hawking (1988), pages 56 and 155.

[47] Stewart (1991), p.1.

[48] Ayer (1956), p. 205.

CHAPTER III: FIELD RESEARCH

"Ye shall know the truth, and the truth shall make you mad."

-- Aldous Huxley

"There is a great difference between knowing and understanding: you can know a lot about something and really not understand it."

--Charles Kettering

A fundamental premise of this study has been that understanding new product development is much more important and involved than just developing armchair theories on the subject. Two requirements were necessary before we could apply this premise to our study: objectivity and relevance.

1. ***Objectivity***: We could not afford to become biased by pre-conceived notions of the processes of new product development. This meant ignoring certain existing, well-established paradigms and, at times, performing triage of previous studies which did not satisfy this requirement.[49]

2. ***Relevance***: Regardless of the temptation, we had to remain focused on the subject matter at hand. Contrary to the oft held belief in some research communities that this directly implies focus, focus, focus... to the point of elemental, reductionist studies, we found that problems exist with *both* over-generalization and over-specialization. Thus, we had to judge which "intermediate" focal area would be appropriate.

---

[49] It is not our policy to brush-off past research work. In fact, we have learned a great deal about product development and its associated disciplines from other research. However, there have been several examples of fraudulent research exposed over the past few years, of which we have been acutely aware. In our field research, we have spoken to executives who asserted that some of the research community's most highly regarded, most prolific, product development researchers have strayed from this objectivity requirement.

This study satisfies both of these criteria. The objectivity requirement has been satisfied through rigorous field study observations at a diverse variety of development sites. By refining and utilizing IDEF0 functional modeling techniques for documenting product development, and by getting developers involved in helping to document their own processes, we forced ourselves to document ***what we saw***, not ***what we wanted to see***. In doing so, we came to some startling, sometimes uncomfortable, findings. More importantly, however, development ***participants*** have played a huge part in unveiling the findings which are presented here.

The requirement for relevance has also been satisfied. We have tried to develop better understanding of the ***overall*** process of new product development, rather than creating optimization methods for individual ***parts*** of the process. One must recognize, however, that this analysis approach can produce results which, at times, conflict with one another, as the dynamic system under study changes.

For purposes of this study, our focus has been on reduction of ***total product development time***. We have also investigated cost and product quality issues, but reserve these discussions for the more detailed appendices[50].

It became evident very early that this subject could not be accomplished satisfactorily in a laboratory. Even an abstraction of the realities of product development could not be con-

---

[50] It is interesting to note that time (measured in man-hours) and cost are nearly equivalent measures in new product development. This is particularly true in development, as opposed to manufacturing, because development is still composed of preponderantly labor intensive activities, with comparatively small raw materials cost.

sidered without substantial first-hand data collection, if this study was to reflect reality. Thus, it was determined that a mix of records analysis, direct observations, interviews, and questionnaires would ***all*** be necessary. Further, development of a systematic modeling technique for analyzing processes in a meaningful manner was considered to be a prerequisite for this type of study At first, I utilized a simple graphically-based process charting technique which I had devised. Subsequently, the strict IDEF0 functional modeling methodology proved very useful[51].

3.1. Summary of Sites & Methods

We visited 42 organizations in the U.S. and Germany. First-hand interviews were conducted with approximately 450 individuals. Questionnaires were administered to over 700 personnel and 14 organizations. 425 of these surveys were returned for analysis. In all, over 1000 people and 100 organizations were contacted. Developmental organizations (departments performing the tasks under review) ranged in size from 3 to 8800 personnel[52]. At many of these sites, comprehensive functional models were generated, with the support of engineers and managers of the companies under study. These studies were performed over a five year period, from late 1988 to mid-1993. A list of field sites, as well as some information about our site visits can be found in Appendix B.

---

[51] An overview of the IDEF0 modeling methodologies employed in this study can be found in Appendix C.

[52] If one considers the integrative nature of some of the sites, we have actually modeled a nationwide "development organization" composed of approximately 100,000 people.

3.2. Summary of Field Study Findings

Our field observations fell into three major areas: ***Recognition, Technical Performance,*** and ***Management Paradigms***. Within each of these categories there are several classification areas. Specific findings from each of these categories can be found in Appendix C. A few major conclusions from the field research are summarized here:

- Product development processes, when viewed with functional modeling tools, exhibit ***high degrees of feedback***, resulting in recursive activity. Such feedback ranges from the chronologically extensive loops of the product life-cycle to many local loops *within* the development "process". When viewed as a *dynamic* system, these circular characteristics may translate into a variety of ***complex (non-linear) behaviors***. Important aspects of this complex behavior include unanticipated periods of pause by some functions (while other functions work furiously), bursts of productivity among engineering functions (interspersed with long periods of lower productivity), amplification and attenuation of communications, moving waves of activity (which surround prototypes as they progress "through" the process), and extraordinary levels of reworking of designs. These behaviors stem from both internal and external influences.

- Commonly, there are ***mismatches in product requirements*** throughout the course of development. Customer wants or needs are often inadequately understood by developers. As development processes proceed, more gaps may be generated than rectified, resulting in less desirable products in the market. Further, rectification

activities can lengthen development time even more, resulting in less competitive opportunity.

- ***Changes in product requirements during the process*** can seriously affect the performance of development organizations. Requirements changes are both ***internally and externally driven***. Though either source may result in circular processes, externally sourced changes occurring late in the process are the most damaging. Acute sources of such externally-driven[53] changes are:

  (1) changes or clarifications in *executive/managerial objectives* or preferences,
  (2) *regulatory* changes, and
  (3) competitive *environment* changes.

  In addition, changes in supplier relations, production labor conditions, capital costs, and so forth could impact requirements. Regardless of source, we found that system-wide dissemination *and understanding* of requirement changes could take significant time, seemingly regardless of technology. This often resulted in further delay and coordination problems among other development participants.

- Many engineers and their management are deluged with ***non-engineering responsibilities***. The "non-value-added" activities associated with fulfillment of these responsibilities can detract from development performance. Though we

---

[53] External, for our purposes, is external to the "development organization." Thus, changes which arise from the same company, but from another division, would be considered ***external***. Conversely, changes which come from within the development process, but from another company (i.e., during joint development projects), would be considered ***internal***.

recognize that some of these activities can be necessary to help coordinate individual efforts during development, many of these activities are merely time sinks in the development process. Furthermore, we have found that such activities ***can occupy significant proportions of development time***. At one site, approximately 85% of engineers' time was dedicated to non-engineering activities. The source of such non-engineering responsibilities was equally remarkable; such activities were often derived from their own prior activity. Thus, much non-value added work which participants complain about is self-induced.

- Though products are released at regular intervals in some industries, we found that most processes have ***naturally irregular product release times***. If regularity is forced on the process (i.e., the new product must be released earlier than the "natural" date), managers and engineers have indicated, then one or more anticipated features of the released product are compromised. Unfortunately, if "natural" releases are allowed, excessive cost and poor market adoption are common.

- When asked about the specific structure and behavior of development processes within their own organizations, development managers and engineers offer inconsistent and conflicting responses. This is not meant to imply that these participants don't *generally* understand their organizations or procedures, but that there are ***variations in specific understanding*** of their dynamic, changing processes. In addition, there seems to be inconsistent understanding of the relative importance of certain process details *of related processes*. Coupled with the observation that processes can be dynamically complex, it can become extremely

difficult for managers to effectively control the overall product development process.

- As development processes approach completion, we have observed an acceleration in the rate of progress (i.e., system effectiveness). This appears to stem from a reduction in the proportion of non-engineering time of engineers, through re-priority of local objectives. Local objectives which do not support "getting the product out the door" are temporarily ignored. This is coupled with increased local activity (e.g., overtime, multiple-shifts, increased employment), which may be either less or more *efficient*. Essentially, players in the system behave in an *effective*, emergency-like ("war-time") mode, rather than a more casual, pleasant ("peace-time") mode. We have branded the resulting system performance as ***development crystallization***.

- Though there are many management methods employed in the field, the ***managerial tools utilized are largely inadequate*** for analysis and control of their development organizations. Many tools in place have roots in manufacturing methodologies. Though such tools may provide some interesting insights, they offer only limited capabilities for better *understanding* of dynamic, non-linear development processes. This is particularly concerning when we consider that successful development projects of the future will have to permanently operate in performance regimes that currently resemble development crystallization.

- Despite the push for more integration among organizational activities, we still observe ***low commonalty of focus*** among participants in development projects.

> This results in local objectives, standards, and reward systems, which further result in reduced emphasis on improving the overall development organization. Thus, ***reductionist paradigms*** are utilized in this ***non-reductionist system***.

Utilizing such conclusions, and the more complete descriptions found in Appendix C, we have developed a model to better illustrate and understand the behavior of product development processes. The structure and performance of this model is presented in the next chapter.

CHAPTER IV: A DETERMINISTIC, SIMPLE MODEL OF NEW PRODUCT DEVELOPMENT

"As far as the laws of mathematics refer to reality, they are not certain;

and as far as they are certain, they do not refer to reality."

--Albert Einstein

4.1. The Need for a New, Dynamic Perspective

In our field studies, we witnessed that new product development can be a highly convoluted path of semi-parallel tasks, being conducted by participants who each carry unique visualizations about "the process" and their impact on the process. Coupled with these basic anomalies is the observation that the processes of information transfer and product (prototype) transfer often loop back onto themselves. This we characterize as ***non-linear*** behavior. Due to these characteristics, we conclude that development participants are involved in a highly complicated, highly complex system. Unfortunately, a disproportionate amount of development time is spent on administrative tasks. The effects of such time expenditures is the focus of the remainder of this report.

We determined that it would be helpful for both managers and the research community to see a more representative view of some characteristics inherent in actual development systems. To enable this, we created a simple, yet insightful model of a development process. For more detailed information on the development and background of this model, refer to Appendix H.

The fact that product development processes in real organizations are composed of many[54] tasks classifies it as a ***complicated*** system. Such complication can obscure certain underlying system tendencies. It would be better if we could relate certain product development behaviors in a simple way. Further, since product development is a system continuously in flux, we would also like to demonstrate its ***dynamic*** nature. The ***non-linearity*** (feedback) nature of new product development should also be incorporated into any demonstration device. As we alluded to earlier, distinctions need to be made between ***information*** processing/transfer and ***prototype*** processing/transfer. Our simplified visualization should accommodate this distinction. In addition, we have seen demonstrated needs for incorporating ***priority*** structures, ***resource allocations***, ***time-based*** measures, ***quality*** measures, ***cost*** measures, ***raw materials*** accounting, ***scrap materials*** accounting, changing task ***efficiencies***, varying task ***dependencies***, ***prototype buffers***, ***information stores***, information ***systems***, managerial ***controls***, ***multiple projects***, and ***simultaneous*** processing. All of these characteristics and capabilities can be visualized in our newly developed model.

In our efforts to create a representative model, we found that there were many attributes of existing modeling structures which could demonstrate parts of the system of development. Using ***no*** single existing modeling structure, however, could we convey **all** the structural and performance attributes discussed above. Some existing structures which we reviewed and utilized included a variety of Markovian processes, calculus and algebraic-based analytical ordering structures, deterministic and stochastic (PERT)

---

[54] For some systems, we have documented over 1000 major distinct activities. Each of these activities are, themselves, composed of many more detailed activities.

networks, state-augmented decision trees, cellular automata, IDEF0 functional modeling structures, IDEF1(x) data modeling structures, and several process-flow techniques.

Thus, we conceived a new analysis methodology. It is called the ***Complex Process Path (CPP)*** methodology. It provides for concurrency, feedback, varying dependencies, and changing service times. It accommodates both time-based and frequency based internal generators, and thus is self-induced, as real human systems seem to be. It is also possible to permit direct impulsive (i.e., managerial) control. Unlike PERT, it does not require a particular beginning or end to each function within the process, except for a more global indicator of when all functions have been performed "adequately." Thus, the process may start at any node or set of nodes. Likewise, it may finish at any node or set of nodes, depending on the self-induced behavior of the system, although it is most likely that the system will always be waiting for some particular node to "finish" the process.

The heart of the CPP methodology is the CPP modeling structure. It is composed of several components, which will be immediately familiar to those familiar with Markov processes, functional structures, and PERT-type networks. Yet, it is not an outgrowth of a Markov process, nor an evolutionary PERT network. It is a different conceptualization technique, which requires different analytical methods. There is still some question of how even simple CPP structures could be analyzed mathematically. Thus far, it appears that computational analysis, simulation, may be the only realistic method for analyzing them. A few of the major reasons for this conclusion include ***abstractness, complexity/ complication*** problems, non-smooth ***variation*** of local service times, human ***contingency***, and a need for ***system-wide insight***. Let us review these briefly, so that the CPP structure may be put into perspective.

4.1.1. Abstractness

While mathematically-based abstract models can be very revealing, their conclusions can greatly depend on the appropriateness of their premises. Yet, limitations of our current calculus abilities (and some spectacular computer-assisted systems to assist such abilities) only permit relatively "simple" problems to work with. We have observed that the era of problem solving via mathematics has been restricted to solving problems which can be structured and solved in "convenient" mathematical forms. Thus, there exist literally thousands of articles with Poisson arrival rates, exponential service times, non-degenerative matrix parameters, and/or well-understood MTBFs, and have thousands of answers to highly abstract, convenient, albeit too-often unlikely problems.

Conversely, attempts at mathematically structuring some realistic problems have often resulted in an algebraic or calculus stalemate, in which "solvability" ***techniques*** become the focal points, rather than the ***answers*** they are to provide. The mathematics can become so complicated so as to render realistic problems un-addressable, particularly for the engineering/development manager.

Rather than get caught in the nuances of the mathematics, which can be a major study in its own right, we decided to look at the performance of a system which more closely resembles observed phenomena. Mathematics has played a major role, but not to the same degree warranted by an operations research purist.

4.1.2. Complexity/Complication

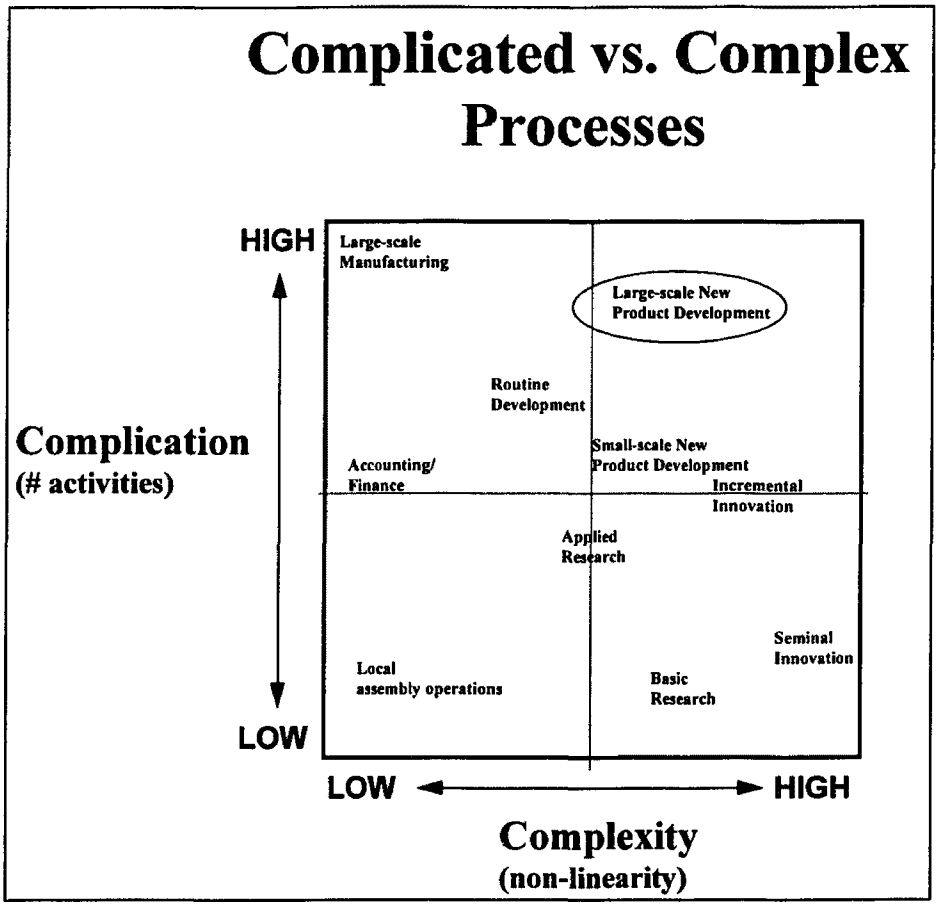
Related to the issue of abstraction is the degree of ***complexity*** and ***complication*** of a development as a system. Mathematics, along with branches of disciplines such as physics, electronics, and mechanical engineering has recently done an honorable job of investigating complex (non-linear) phenomena. When systems get complicated (high number of interconnected nodes), mathematics has proven less useful. Computational analysis has more strength in this area. Now, we visualize a system of organizational behavior which has elements of ***both*** complexity and complication: the highly iterative developmental process, with as many as 500 individual engineers and up to 1/4 of a million communication channels[55] on a variety of communication media.

To help put new product development in perspective, consider Exhibit IV.1. Here we contrast a variety of common processes, according to their relative complexity and complication levels.

---

[55] This estimate is just the number of possible channels among 500 individuals: n(n-1) = 499 x 500 = 249,500. Of course, observed processes only use a fraction of these possible channels. A priori, however, it is not known ***which*** channels will be used. Thus, a realistic model needs to consider that any of these channels are conceivably open.

EXHIBIT IV.1.



Notice that new product development reaches far into the HIGH complexity, HIGH complication zone. Contrast this with a large manufacturing system, which may be very high on the complication scale, but tends to avoid complexity. A large manufacturing system can be seen as a multitude of smaller, far less complicated systems. Processes of invention, on the other hand, may have few activities, but incorporate very high levels of feedback, making them highly complex.

If one can identify the specific nature and relation of any of these types of activities, they may be modeled into a CPP structure. Thus, the CPP

methodology can reduce in both complexity and complication to suit the particular system under analysis.

4.1.3. Variation

Overlaid upon this complicated, complex nature of new product development is another consideration: variation. Different parts of the system operate with changing efficiency and effectiveness at different times. Attempting to mathematically model such variation for even simple systems is extremely difficult. Stochastic modeling, whereby each part of a system operates in a statistically distributed fashion, is an attempt at examining some effects of variation. Even stochastic modeling, however, tends to focus on convenient statistical distributions, with little work on continuously varying distributions. That is, distributions which, in and of themselves, vary over time within the system. By creating an *a priori* dynamic functional efficiency generator, we can study such variation with the CPP methodology.

4.1.4. Contingency

One of the observations of this study has been that participants in the development process do not vary their work rate purely by chance, nor by pre-programmed algorithm. There are usually some justifiable reason(s) for their variation. Much of the time, this appears to be a function of the more immediate situation which an individual faces. In this study, situation dependent performance has been observed to be the prevailing *rule*, not the exception. Attempting to model such behavior mathematically would be painstaking, if even possible. Utilizing dynamic, real-time Boolean comparison algorithms, the CPP methodology can accommodate this behavior quite well.

4.1.5. System-wide Insight

One of the reasons for conducting research on the new product development process is to glean more useful information than has been observed to date. This has meant looking beyond local concerns of individual engineers and their managers, and realizing that these people are key elements to a more global, constantly changing system. Thus, our model of development cannot just focus on the overall output of the system; they cannot be centered around just the quality of system outputs; they should not be solely on finding the best methods for determining the real costs of running the system; they cannot dwell on localized "fire-fighting" which design engineers seem to endlessly partake; they cannot just examine the amount of time which engineers spend doing engineering tasks; nor can they hone in on redundancy of efforts. Rather, we need to keep in mind that these are some of the issues which developers and their management address over and over in each particular development project. Each of these issues are closely intertwined ***and can not be separated*** for accurate analysis. Accurate models demonstrate that development systems are not just the sum of their parts, but sometimes less and sometimes more. They demonstrate a continuously changing system which, according to our field observations, never performs exactly the same way twice, yet performs with similar characteristics time and time again.

We are interested in harnessing the full potential of the system, with objectives of minimal cost, minimal time, maximum "quality" of output, minimum aggravation of participants, and a host of other localized manager-specific objectives. Without adequate understanding of the entire system, however, it is extremely unlikely that *any* of these objectives will be met in a satisfactory manner.

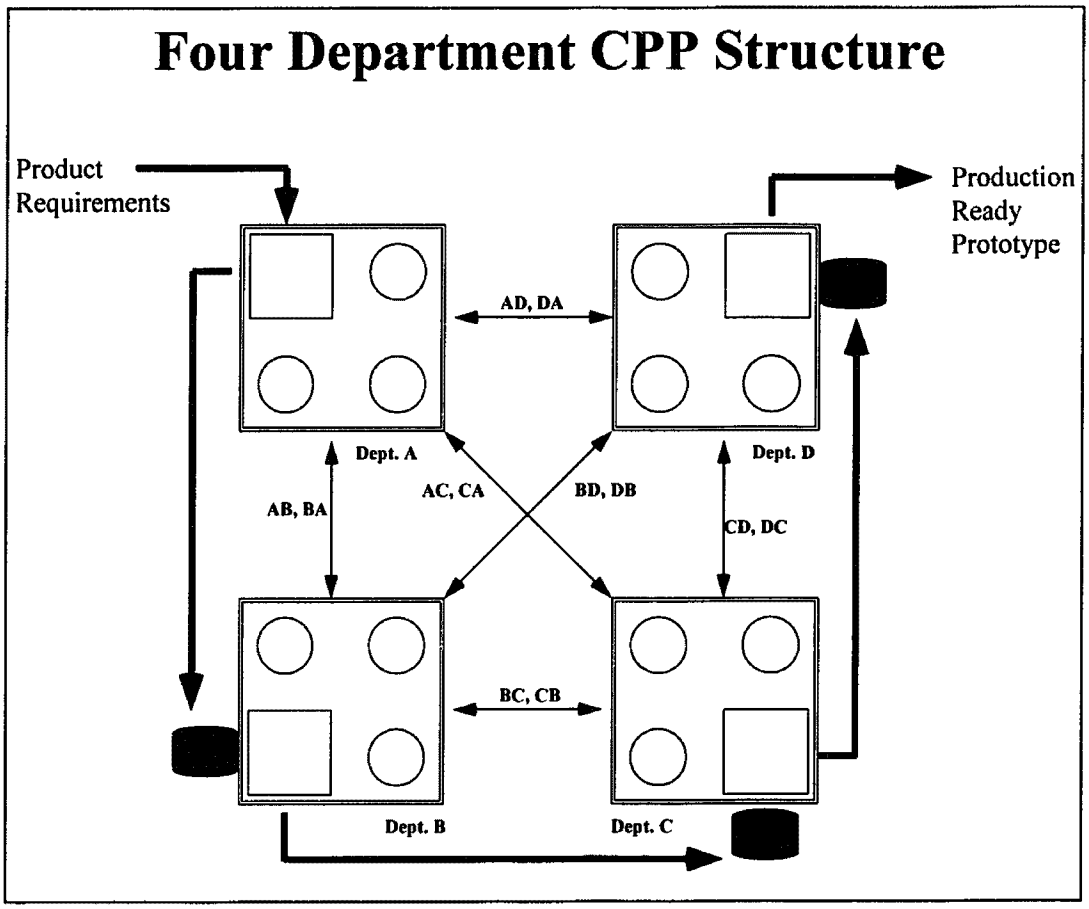
4.2. A Complex Process Path Model Structure

The basic structure of a CPP system is illustrated in Exhibit IV.2. Though simple in form[56], this particular structure contains several attributes which have been observed in a multitude of development organizations. Some of these attributes include:

- Sequential, Unidirectional ***Prototype*** Movement
- Bi-directional ***Information*** Transfer
- Limited ***Human Resources***
- Variable ***Local Processing Efficiencies***
- ***Prototype Storage*** Facilities
- ***Information System*** Limitations
- ***Contingent Behaviors***
- ***Information Priority*** Schemes
- ***Behavior Priority*** Schemes
- ***Simultaneous*** Processing
- ***Multiple Projects***

[56] The terminology in this section is especially important for the reader to understand, because it will be used repeatedly in the CPP results section of this report.

EXHIBIT IV.2.



The structure of this system offers an infinite variety of scenarios to be contemplated. By modifying the values of specific parameters, this structure can range from a simple, linear process to a complicated, non-linear (complex) process. The following pages outline the basic structure of the model. This structure is categorized into model ***elements*** and model ***attributes***. Elements are the "physical" building blocks of the system. Attributes are the specific characteristic which describe how these building blocks "behave."

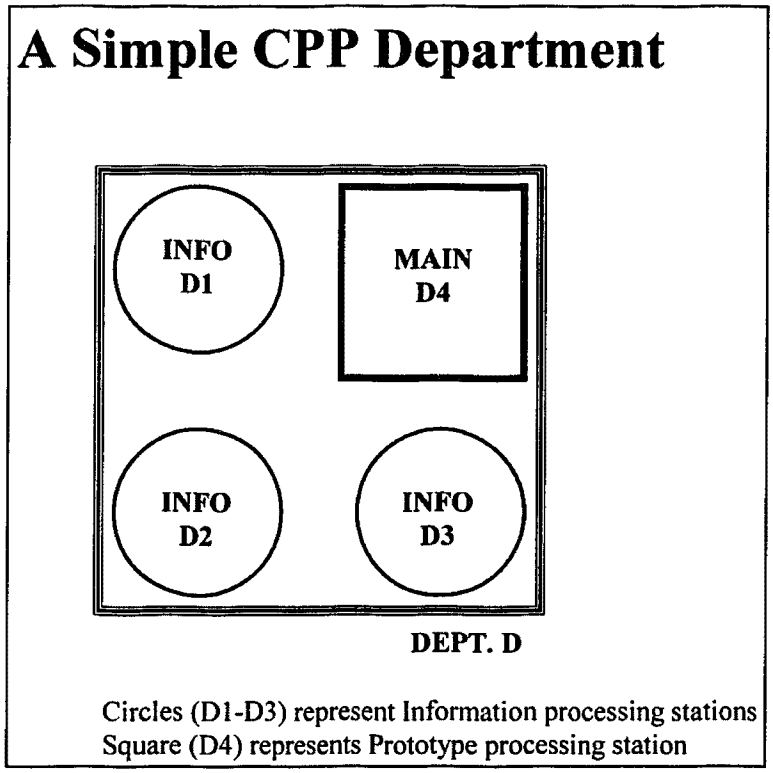
4.2.1. Model Elements

The basic elements of the structure include ***stations, information, prototypes, buffers,*** and ***human resources*** (people).

Stations are grouped into sets, which collectively represent "departments". Engineers work within a department, specifically at station(s) within that department. Thus, each department may be considered a "shell organization", composed of stations. Departments are denoted with letters; in this particular CPP model, there are four departments: A, B, C, and D. Stations within a department are denoted with the department letter, followed by its own number. For example, the first station within the first department is denoted "A1"; the third station at the fourth department is denoted "D3". Each particular station has its own operating rate, which might vary with specific conditions, and requires its own number of human resources. Refer to Exhibit IV.3.

A station is representative of some *place* (desk, drafting table, CAD workstation, workbench, boardroom, etc.,...) within the department. At each station, there are particular tasks, or functions, to be performed. Given various parameters (or rules) in this model, a station may repeatedly start and stop performing a function according to the conditions which it encounters at any point in time. Pursuant to a functional hierarchy, stations in this model perform the most elemental functions.

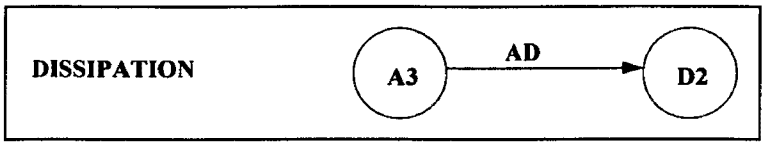
EXHIBIT IV.3.



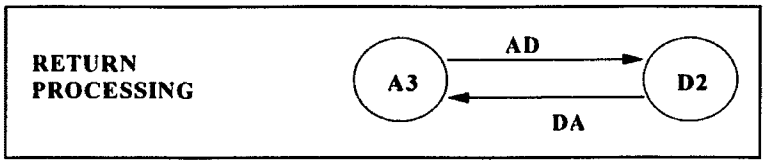
A CPP model may have as many departments and stations as the analysts may wish. For simplification, we have limited the number of departments in this model to four. For further simplification, each department is composed of only four stations, for a total of 16 stations. Compare this to a real development organization, where there may be several hundred stations. As if this simplification is not enough, we have also limited each station's capability to that of a single function. For reference, we have documented real development systems with over 1000 functions. Given these simplifying circumstances, the terms ***station*** and ***function*** may be used synonymously.

Information is passed from department to department, on an "as-required" basis. Channels for such information are indicated with the fine, two-way arrows between departments. When a department receives information from other departments, the information is typically sorted by priority. Then, each of the appropriate stations (within that department) process the received information. Information processing may have three results: *dissipation*, *return*, and *re-direction*.

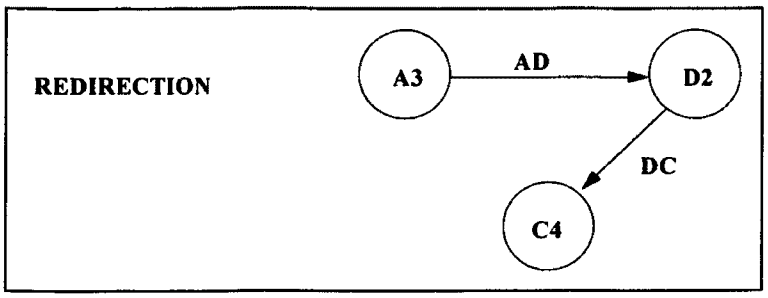
- ***Dissipation*** is when information is processed, but not forwarded to any other station. Hence, when a station is dissipating information, it produces no further information for other departments. Network analysts would refer to stations operating in this manner as an information ***sink***.



- ***Return processing*** is when a station receives information, processes it, and then sends information back to the original source. For instance, if station D2 is performing return processing of information that came from Department A, it would send some information back to Department A.



- ***Re-direction*** is when a station sends information to a department other than its previous "origin." Hence, if station D2 is performing redirection processing on information that came from Department A, the next destination would be another, such as Department C.



Information flow is often considered a positive indicator of departmental preparedness, and sometimes considered a precursor to quality. Paradoxically, information flow can also have negative consequences, in design time, throughput per unit time, cost, and quality. In our discussion of simulation results we address such issues in more detail.

Prototypes, like information, are also passed from department to department. However, this particular structure assumes that prototypes (or, more accurately, prototype sub-assemblies) proceed in a sequential fashion: Department A --> Department B --> Department C --> Department D. In the general case CPP structure, this unidirectional sequence is not required.

By using this sequence, however, our characterization of "Departments" can also be described as design "phases". It may be convenient to think of "Department A" as the "Conceptual Design" phase; "Department B" as the "Detailed Design"

phase; "Department C" as the "Prototype Build" phase; and "Department D" as the "Production Preparation" phase. Such terminology is common in development organizations today.

However one classifies such phases, it is clear that all phases have some interaction with all other phases throughout the course of development. The degree to which this interaction takes place or should take place is the subject of considerable interest in the current design community. In this model, this interaction is demonstrated in both the prototype flow, as well as the information flow described above.

This brings us to an important consideration: In the development process, how does one separate ***information*** from ***prototype***? Particularly at the earliest stages of development, there are few tangible items which can be considered "prototypes" or even "prototype assemblies". Specifications of various detail, conceptual drawings, memoranda, customer requirements, internal organizational requirements, governmental regulations, previous production samples, alternative "concept" assemblies, and so forth are passed about in a variety of formats. Gradually, developers convert such "non-prototypes" into a physical core which at some, seemingly arbitrary, point is considered a preliminary "prototype." For some developers who perform detailed design near the end of the development chronology, such preliminary "prototypes" and their documentation are not stable enough to work with. For them, such a fluid, though tangible, item may be considered mere information, perhaps even clutter which interferes with their daily activity.

At times, non-physical information is considered to be as good as the "real thing." In the current period of increasing CAD/CAE usage, physical prototypes or mock-ups are less

plentiful, relative to their computer-created design "models." It may be conceivable that "prototype-less" design will be conducted on a much wider scale than at present[57]. Nonetheless, such a computer model is considered a "prototype" for our purposes.

The judgment of what to classify as prototype and what to classify as information rests with the individual or team of individuals receiving such information. If the communication being reviewed is usable or necessary for completion of one's development function (i.e., a function which physically modifies or creates new aspects of the design), then it may be considered a "prototype," even if it doesn't "look" like a traditional, physical "prototype." In this regard, a prototype may be realized as a form of ***input*** (in IDEF0 parlance) to the specific function.

Information, on the other hand, may be considered a ***control*** to a development function. It does not contribute directly to development, but may carry some data about how to perform the function or some contingency/coordination considerations about the process, relative to other functions.

By changing the communication structure from station to station (and thus from department to department), management has the capability to "play God" with the system. For example, by removing or enabling a communication channel and screening-in or screening-out classes of information content, the structure and dynamic behavior of the

---

[57] For many hand-held consumer electronics products today, such non-prototyping custom "design" is conducted today. In this scenario, the relatively simple requirements are converted directly to tooling specifications, which are fed to the CAM machinery to stamp and assemble the final product. Depending upon the level of pre-structuring of requirements, this is also known as FMS production with lot size of one.

system may change. This is true of both information and prototype channels. Existing static models do not permit such insight.

To help isolate and track information processing from "mainstream" design processing, departments contain two types of functions: "INFO" functions and "MAIN" functions. INFO functions process information from each of the other departments. MAIN functions process and produce prototypes. In our simple CPP structure illustrated here, each department contains 3 INFO functions and 1 MAIN function. For example, Department A has four functions: INFO B, INFO C, INFO D, and MAIN A. Departments and their specific stations and functions are denoted in Table IV.1.

Information is denoted in the CPP Structure with a two letter code, *XY*, where *X* denotes the immediate source of the information and *Y* denotes the immediate destination of the information. Thus, information described as DB is in transit from department "D" to department "B". The various sources and destinations of information in our simple CPP Structure are demonstrated in Table IV.1.

Prototype materiel is denoted as "Prototype *X*", where *X* denotes the source of the prototype materiel. In our model, there are only four such elements, each sourced from their respective departments. They are denoted in Table IV.1.

TABLE IV.1.

CPP Model
Information and Prototype
Channel Structure

Dept.	Station	Function	Inputs	Outputs
A	A1	INFO B	BA	AB,AC,AD
	A2	INFO D	DA	AB,AC,AD
	A3	INFO C	CA	AB,AC,AD
	A4	MAIN A	--	Prototype A, AB,AC,AD
B	B1	INFO A	AB	BA,BC,BD
	B2	INFO C	CB	BA,BC,BD
	B3	INFO D	DB	BA,BC,BD
	B4	MAIN B	Prototype A	Prototype B, BA,BC,BD
C	C1	INFO A	AC	CA,CB,CD
	C2	INFO B	BC	CA,CB,CD
	C3	INFO D	DC	CA,CB,CD
	C4	MAIN C	Prototype B	Prototype C, CA,CB,CD
D	D1	INFO B	BD	DA,DB,DC
	D2	INFO A	AD	DA,DB,DC
	D3	INFO C	CD	DA,DB,DC
	D4	MAIN D	Prototype C	Prototype D, DA,DB,DC

Lest this might seem complicated, consider the following:

- INFO function (stations) *only* process information;
- MAIN functions *never* process information;
- both INFO and MAIN functions *can produce* information;
- only MAIN functions can produce prototype. INFO functions cannot.

Buffers are implicit in any information or prototype transfer process. These buffers are usually inherent to the transfer mechanism; whether a simple physical conveyor or a complicated high-speed information system, there is some time period over which the transfer process takes place. The location or locus of locations which facilitate the resulting temporary storage of information or material may be considered as buffers, whether physically evident or not.

The importance of such a buffer is twofold. First, it reveals potential ***time delay*** between the sending of information or prototype at one end and the receipt of information or prototype at the other. Second, its size can affect the instantaneous flow ***capacity*** of the transfer process. If the buffer capacity is reached, then a blockage is formed upstream of the transfer process. Depending on the nature of the process flows, this blockage can have either debilitating or negligible effects on the system.

There are two classes of buffers: ***information buffers*** and ***prototype buffers***.

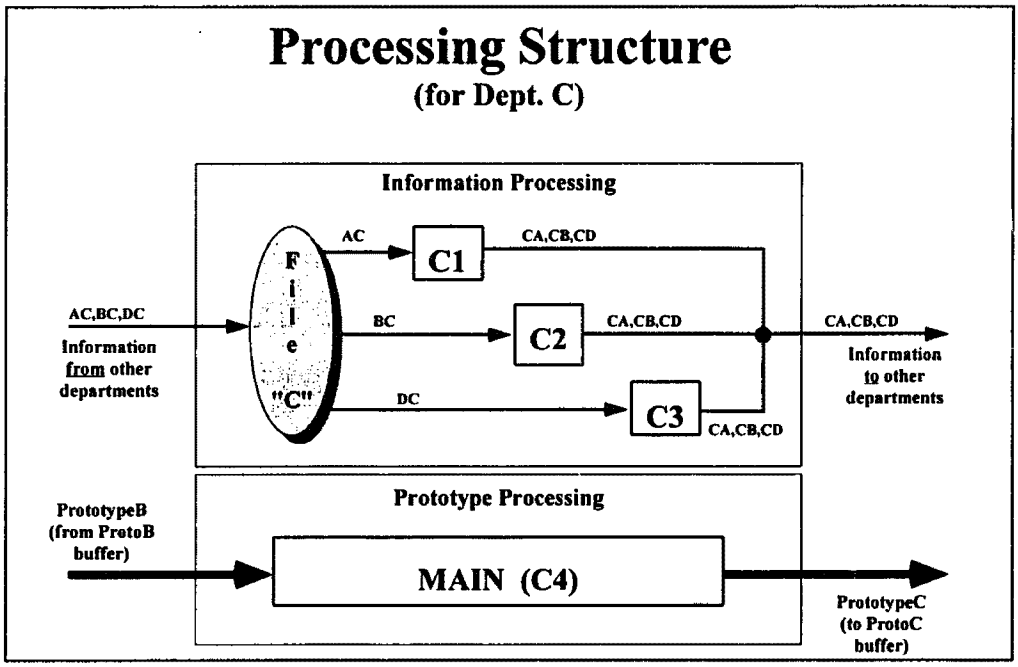
Information buffers limit the amount of information which can reside at the "in-box" at each department. They are useful for tracking the amount of

information which must be processed prior to working on a MAIN function. Thus, they make a useful and interesting indicator of a department's immediate work backlog. This will be discussed further in the results section.

Such buffers are widely apparent in real product development organizations, though they are rarely characterized as buffers, per se. Obvious tangible examples include physical and electronic mailboxes, desktop in-boxes, "to-do" files, and task lists. Not so obvious examples include limited schedules and mental triage schemes.

In this model, there are four information buffers: one for each department. They are denoted as **FileA, FileB, FileC,** and **FileD**. Refer to Exhibit IV.4., which illustrates the use of **FileC** for Department **C**.

EXHIBIT IV.4.



Prototype buffers limit the number of prototype assemblies which can be held in queue at each department, or phase. These buffers have been established between the output of one department and the input to the "next" department. The prototype buffer is especially apparent when one considers development systems that are responsible for more than one design (i.e., multi-product organizations). Such buffers are physically apparent, often as parts bins[58], storage rooms, desk drawers, or, in more modern facilities, computer file storage (i.e., computer hard-drives, storage tapes, optical disk, etc.).

For this model, there are three prototype buffers of interest: those which follow MAIN A, MAIN B, and MAIN C. They are denoted as **ProtoA Buffer , ProtoB Buffer ,** and **ProtoC Buffer**. Technically, there is also a buffer which follows MAIN D (denoted as **ProtoD Buffer**). However, since the prototype assembly leaving MAIN D is considered the "Final Design", this buffer has been set large enough in our analysis to have no restrictive effect on the MAIN D function.

Specifics on the sizes of such buffers are presented as attribute considerations.

---

[58] At one site, such bins, which occupied the better part of a large room and were used as pre-assembly storgae areas, were nicknamed "bannana boxes", due to their appearance as large produce shipping crates. Lest one think that such bins are trivial to consider, contemplate this: the contents of such bins could easily sit for 6 months, waiting for a single item to arrive, or for the downstream activities to be ready use the componentry. The significance of storage costs of this accumulated WIP is an on-going battle between engineers and accounting personnel.

Human Resources are the critical component of all design processes. They too are represented in the CPP modeling structure. Naturally, the appropriate utilization of these resources is a perennial concern among engineering management. Questions such as "How many engineers should we have?", "How many engineers should be focused on the mainstream functions", and "What percent of our engineers' time is spent doing engineering functions?" are asked by executive engineering managers regularly. The answers to these questions are not clear cut, however, for numerous trade-offs often exist.

4.2.2. Model Attributes

In addition to these visually evident and tangible elements, there are underlying components to the model which affect the performance of the system. These are called ***attributes*** because they are specific descriptors of the main elements of the "system." In fact, they are not "seen" in the outward physical structure of the model, but are key underpinnings to facilitate the structure. They include ***station processing rate, information transfer probabilities, priorities,*** and ***buffer sizes.***

Station Processing Rate

In any time-sensitive process, one must consider the speed at which a function is performed. In this model, such a rate is expressed in units of time per function. Thus, a rate of 10 means that it takes 10 units of time (days, in our example) to complete a function. This can be represented as a constant rate or as some distribution of rates which vary in time, status of another part of the system, or random variation.

Overlaid upon this rate is an ***efficiency*** figure. This is merely an indicator of how well the station is operating, relative to the stated baseline rate. When operating at 200% efficiency, for example, a station is capable of processing twice as fast as its baseline rate-- 5 days per function in this example.

In the current CPP model, the baseline station processing rates were set as follows:

- All MAIN functions: 10 days/function (with no variance)
- All INFO functions: 1 day/function (with no variance)

From run to run, the efficiencies of these stations were simultaneously varied from 50% efficiency to 200% efficiency, in 50% increments. Thus, the ratio of MAIN processing rate to INFO processing rate varied from 2.5 (MAINRATE = 5, INFORATE = 2) to 40 (MAINRATE = 20, INFORATE = 0.5).

Information Transfer Probabilities

When creating or processing information, an individual typically gives some indication of ***where*** the resulting output information should be sent to. Further, one should expect some kind of indication of how well such attempted information transfer took place. This is the reason behind information transfer probabilities.

Information transfer probabilities have been established to direct the flow of information from one department to another. More specifically, these parameters are instructions for each station, revealing the frequency of a particular output, or sets of outputs, given some

input(s). Thus, information transfer probabilities define the situation analysis for a given station, once that station has been "told" to start working.

This very powerful feature raises a relevant and equally powerful question, which can be used to judge just about any research on human decision-making. It was directly observed in the field studies that individuals are continually performing their own situation analyses. The specific rules by which an individual actually makes such localized decisions seem to be highly obscured by a number of subtle factors, many of which may not even be explicit to the decision maker. Thus, how can we be so presumptuous as to define the situation analysis logic of engineers?

Clearly, we cannot define an individual's situation and cognitive processing in a satisfactory manner. The best we have been able to do thus far is attempt to decipher and classify the "most important" factors in one's situation, and observe that individual's response. Even then, we cannot be sure that the responses we observe are accurate reflections of the individual's thought processes or capabilities (Evans, 1991).

Thus, it is probably unrealistic to expect that comprehensive situation analyses can be simulated. Does this mean that practical aggregated situation analyses are unrealizable? Our research suggests that some construction of aggregated situation analysis methods may be useful for management decision-making. One must observe this caveat, however: Good judgment of how much aggregation is excessive requires, in itself, a complicated situation analysis.

In this analysis, information transfer probabilities offer but one approach to seeking some intermediate level of situation analysis aggregation. By merely specifying simple conditional probabilities of occurrence (based upon the origin of the information being processed), we can offer a flavor for some overall, longer-range tendencies, without strict rules or policies for human behavior. Thus, we can see some ***overall tendencies***, without really knowing ***exactly*** what will happen next; a situation observed readily and repeatedly in the engineering organizations from the field studies.

We hope that much more research will be conducted on situation analysis during product development activities. In the future, perhaps we can reduce our reliance on mere probabilities and discover underlying ***drivers***.

Information transfer probabilities were carefully developed during the course of model formulation. Using some judgment from participants, as well as observed aggregate qualitative results (e.g., "there is little interaction about conceptual development among the throngs of production engineering personnel" translated to "Department A talks less to Department D than it does to Dept. B or Dept. C."), probabilities were generated. Subsequently, these probabilities were slightly modified, to check their sensitivity. The results of these perturbations is a topic which is addressed at more length in the results section of this report.

For the bulk of the dynamic development runs, information transfer probabilities were established with the following guidelines:

- Information is "generated" by MAIN functions only. Thus, any and all "infoglut" among INFO functions is merely *n*th order ripple effects of information sent from MAIN functions.
- MAIN functions, upon completion of a "prototype", always send some "bundle" of information to each of the other departments (this information is processed by INFO functions).
- MAIN functions send more information to "closer" departments than to those which are "further away."
- INFO functions engage in return processing (i.e., send information back to its "source" department) 50% of the time.
- INFO functions send information to "closer" departments more often than to those which are "further away."
- INFO functions send no more than one "unit" of information to other departments (i.e., they may send one unit or nothing, after some processing). Thus, the INFO functions are not generators of more information than they "consume."
- INFO functions only send information to other departments (assume that intra-departmental communication is part of the INFO processing function).

Exhibit IV.5. is an information transition probability matrix used for a number of runs.

EXHIBIT IV.5.

Information Transfer Probabilities
(4 departments)

Run #: H (complex8 setup)

		TO											
		<-- Dept.A		-->	<-- Dept. B		-->	<-- Dept. C		-->	<-- Dept. D		-->
		BA	DA	CA	AB	CB	DB	AC	BC	DC	BD	AD	CD
	BA	xxx	xxx	xxx	50%	xxx	xxx	10%	xxx	xxx	xxx	5%	xxx
	DA	xxx	xxx	xxx	20%	xxx	xxx	10%	xxx	xxx	xxx	50%	xxx
	CA	xxx	xxx	xxx	20%	xxx	xxx	50%	xxx	xxx	xxx	5%	xxx
	AB	50%	xxx	xxx	xxx	xxx	xxx	xxx	20%	xxx	10%	xxx	xxx
	CB	20%	xxx	xxx	xxx	xxx	xxx	xxx	50%	xxx	10%	xxx	xxx
F	DB	20%	xxx	xxx	xxx	xxx	xxx	xxx	20%	xxx	50%	xxx	xxx
R	AC	xxx	xxx	50%	xxx	20%	xxx	xxx	xxx	xxx	xxx	xxx	20%
O	BC	xxx	xxx	10%	xxx	50%	xxx	xxx	xxx	xxx	xxx	xxx	20%
M	DC	xxx	xxx	10%	xxx	10%	xxx	xxx	xxx	xxx	xxx	xxx	50%
	BD	xxx	5%	xxx	xxx	xxx	50%	xxx	xxx	20%	xxx	xxx	xxx
	AD	xxx	50%	xxx	xxx	xxx	10%	xxx	xxx	20%	xxx	xxx	xxx
	CD	xxx	5%	xxx	xxx	xxx	10%	xxx	xxx	50%	xxx	xxx	xxx
	MAIN A	xxx	xxx	xxx	20*	xxx	xxx	10*	xxx	xxx	xxx	5*	xxx
	MAIN B	20*	xxx	xxx	xxx	xxx	xxx	xxx	20*	xxx	10*	xxx	xxx
	MAIN C	xxx	xxx	10*	xxx	20*	xxx	xxx	xxx	xxx	xxx	xxx	20*
	MAIN D	xxx	5*	xxx	xxx	xxx	10*	xxx	xxx	20*	xxx	xxx	xxx

Notes:

* Each of the "MAIN" functions always produces the listed number of information units, rather than probabilistic singular output (as with the INFOx functions). Thus, MAIN B always "produces" 50 information units: 20 of these units are always sent as "BA" information (to be processed by Station A1); 20 of these units are always sent as BC (to Station C2); and 10 of these units are alwayS sent as BD (to Station D1).

Priorities

In addition to information[59] transfer probabilities and station processing rates, it is useful to consider ***communication priorities*** and ***processing priorities***. This allows one to consider the very real situation whereby a particular message is extremely urgent, and needs to be processed by a department before other information. Clearly, a FIFO buffer sorting technique is inappropriate in this case. Yet, a LIFO protocol is inappropriate in a variety of other cases.

By prioritizing information as it is sent, and sorting by priority within the recipient buffer, it is possible to "sneak" some messages ahead of others, and let some sit in the "in-box" until slack times. This is a very real situation which can, at times, cause hardship for those waiting for their "low priority" information to be processed by another department.

Communication Priorities

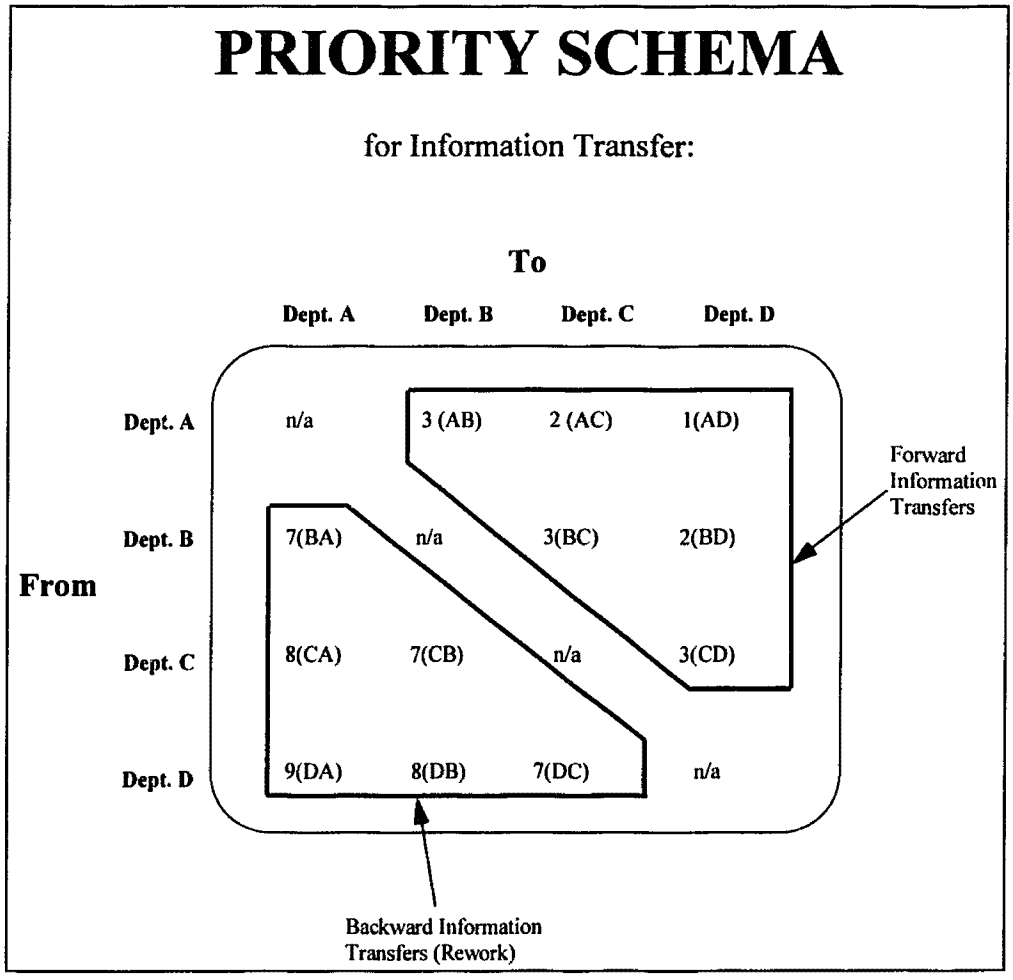
For information transfer, priorities have been established according to estimated criticality of a source department's information signals. Based upon experiences of engineers and managers interviewed in this study, it was evident that information from ***functions "closest" to production get the most attention***. Secondly, it was apparent that developers and managers ***listened to their downstream comrades more than their upstream counterparts***. At this point, we have assumed that such experiential judgments

---

[59] We have incorporated information transfer probabilities into this CPP model. Yet, we did not discuss such probabilities for prototype transfer. The reason for this curious omission is that, in actual practice, prototypes tend to be sent from department to department with a bit of fanfare. It is assumed that the prototype message arrives at the next department every time that it is sent--giving a prototype transfer probability equal to 1.0. With the CPP structure, we can provide for less ideal transfer by merely lowering the probability.

about "who to listen to" are correct. Given this, we have established information transfer priorities in this specific CPP model according to the matrix in Exhibit IV.6.

EXHIBIT IV.6.



In this matrix, information priority is revealed on a 0-10 scale, with 10 being considered the highest priority. Thus, we can see that information sent from Dept. D to Dept. A (also known as "DA") has a very high priority rating of 9. Reflective of our upstream/ down-

stream criteria, we see that Dept. A's information signal to Dept. D is considered to be of very little value to a Dept. D developer: its priority rating is only 1. (The letter code after each number in the matrix represents a quick-reference abbreviation for the information channel of interest. Note that priorities are not mutual. For instance, the BA channel (a backward channel) has a priority rating of 7, whereas the AB channel (a forward channel) only has a priority rating of 3.

For prototype transfer, the priority rating scheme has been kept simple: FIFO. This may be changed in the CPP structure, although there seems to be little value in doing so at this time. A contributing reason for this is that each prototype transferred from one department to the next is considered "generic." Couple this with our sequential prototype flow and it does not matter, for our current purposes, ***which*** prototype is being worked on. Rather, we focus on ***whether*** a prototype is being worked on. Thus, any priority scheme can be used with the same results.

Processing Priorities

This brings us to the critical consideration of station processing priorities. Given that we have limited human resources and that such resources are flexible enough to perform more than one type of function within a department, we need some prioritization of activities. This extends beyond mere allocation of human resources to a department and information transfer priorities and into a judgment of choosing between the entire class of information processing functions and prototype processing. Specifically, we are interested in how much time is spent on information (INFO) processing and how much time is spent on prototype (MAIN) processing.

Relative station priorities among stations 1, 2, and 3 (the INFO functions) at each department are simple. They merely follow the priority schedule for information transfer. This comes about because of the 1:1 mapping of information input channels with each station (refer back to Table IV.1.).

When considering priority levels for MAIN functions (station "4" at each department), we have to make some practical judgment of how important prototype development is to the engineer, all things considered.

With no callousness towards engineers implied, our field studies revealed that ***development of new products is not the highest priority among development engineers.*** Higher allegiances are made to a variety of information processing activities which have little or nothing to do with developing the product. Answering memos, writing memos, attending conferences, attending committee meetings, filing resource reports, developing (and editing and editing...) program status reports, interviewing potential development personnel, planning and coordinating efforts of others, tracking and following up on ordered parts (affectionately called "chasing parts"),... even conversing with researchers about their engineering process--These (and many more) non-development activities take on ever increasing portions of engineers' time. Why is this so?

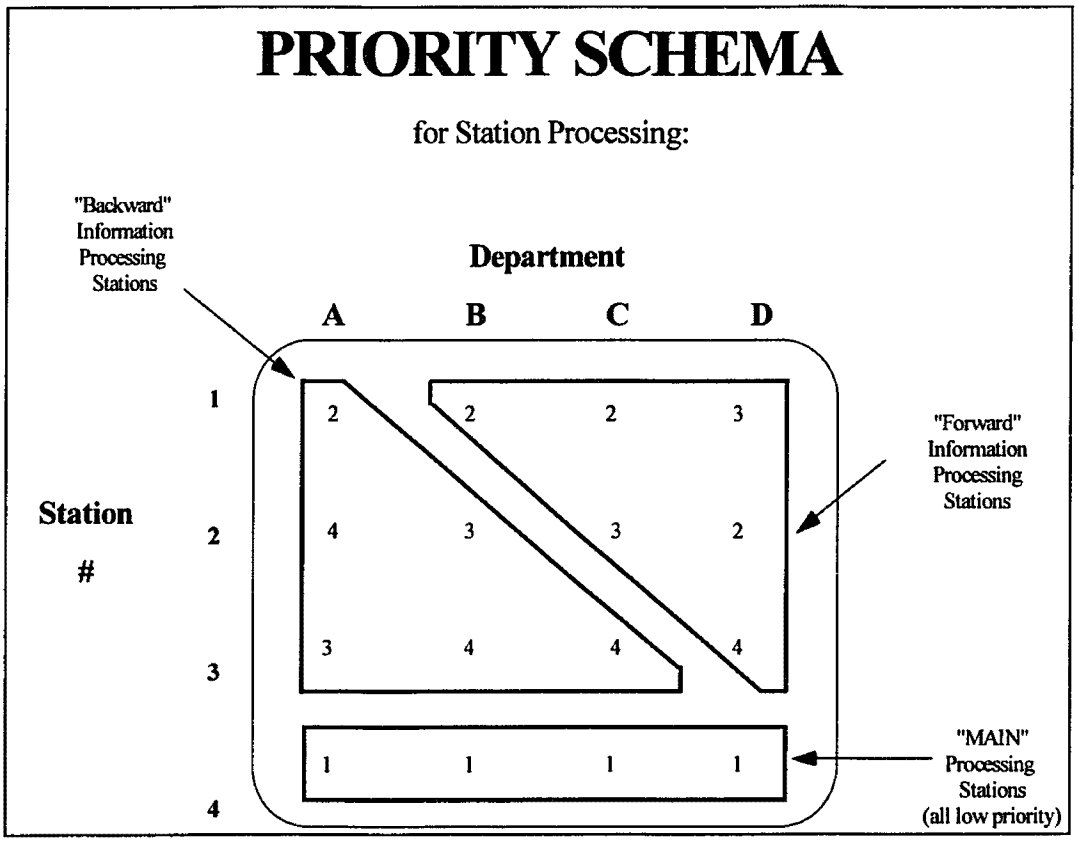
Some will argue that this typifies discipline problems of engineers. Some will offer this as a symptom of inappropriate scoping and scheduling of one's time. Others will argue that these are activities which *must* be done by engineers, because they are the only personnel qualified to perform them. Still others feel these activities are necessary, just to permit the engineer to get back to the development tasks. In some cases, it is also

apparent that such activities are performed as acts of conformity--so that one may fit socially into their environment (hey, "everyone else does it, why shouldn't you?"). In this regard, such activity has become somewhat of a routine for many engineers, particularly for developers who only work on small portions of an enormous development project (such as a defense system, commercial aircraft, or automobile design).

Regardless of the reasons and justifications[60] for this behavior among engineers, this model relates this behavior by establishing very low priority to the MAIN functions. In fact, MAIN functions have been assigned lower priority than *any* of the information processing functions. Refer to Exhibit IV.7.

[60] Despite these observations, we cannot suggest, nor even believe, that many engineers *enjoy* or seek out these "inverted" activity priorities. Rather, one gets the impression that engineers tend to resent such priority inversion and are somewhat *obligated* or thrusted into such activity because "that's the way the system works" and that the perceived cost of doing otherwise is personally too high.

EXHIBIT IV.7.



How does this priority schedule impact the behavior of each department? MAIN functions are only "eligible" to operate when two conditions are simultaneously satisfied:

1. A prototype assembly is available from the previous department, and is ready to be "pulled" from the prototype buffer (i.e., Proto Buffer(x-1) is not empty).

   AND

2. There is no more information in the in-box for that department to process (i.e., the FileX buffer is empty).

These conditions imply that development will only occur under certain windows of opportunity. At all other times, the "department" is busy churning information or waiting for upstream activities to finish their prototype processing.

These prioritizing capabilities are not merely "nice" features of this model; they have significant implications in system performance. For, an ordinal change in one operation can cause ordinal changes in subsequent operations. Slight changes in priorities, as with transfer probabilities, can have unforeseen, unpredictable effects on system performance. At times this effect is negligible; other times it is highly visible and substantial.

Buffer Sizes

In the CPP structure, we have the capability of modifying the size of both information and prototype buffers. Based upon experimenting and understanding their impacts, a variety of interesting sizes were determined for the majority of the formalized simulation runs. Recall that buffer sizes can reflect *time* delay and *capacity* of information and prototype transfer systems. In this vein, buffer sizes can be limited through time or physical restrictions. In the CPP structure, we could use either of these approaches. We have chosen, for simplicity, to use buffer size (specifically, count) as the defining measure for the upper limits of buffers. Though these upper limits were varied during our analysis, we never induced any lower limits on buffer size, other than zero. Though such limits are conceivable in the CPP structure, doing so would necessitate changes in the simple priority schemes we have forwarded.

Human Resource Allocation

Among managers in the development organizations we visited, the task of appropriately allocating personnel was considered very important. Because labor cost is such a significant portion of development expenses, it seems wise for frugal development managers to keep their eyes open for continuous efficiency improvement of personnel. Appropriate allocation of engineers is considered a major step in getting them to work together efficiently. The CPP structure provides excellent opportunities to evaluate alternative allocation strategies.

In the CPP structure, human resources (we call them "engineers" from here on[61]) are limited in their number. The demand for engineers is usually higher than their supply. This causes shortages, for which allocation judgments must be made. Formulation of such judgments are the very heart of the engineering management function and are extremely important. Yet, the complex cognitive dynamics of such judgments are beyond the realm of this research, and test the limits of the most progressive cognitive research to date. We have limited the judgment to a few simple policies and observed the system-wide dynamics. By observing such dynamics it is possible to more clearly see when and why alternative judgments can be useful, harmful, or immaterial.

---

[61] Naturally, development organizations have more than just engineers as employees. In fact, at one development organization employing approximately 8800, only 795 of them were considered design engineers. The other 8000? They were composed of engineering assistants, technicians, managers, and administrative staff. For purposes of the CPP structure, anybody who ***participates*** in the process is considered an engineer, regardless of official title. If applied to the above organization, this would raise the "engineer" count to the neighborhood of 2500. This method also reveals that certain administrative personnel have similar impact as strategists and other high-level management who are responsible for structuring and controlling the system. Much to the chagrin of many engineers, this has a certain thread of reality to it.

In the CPP structure, we have the capability to change the number of engineers employed in each department, the number of engineers required to perform MAIN functions within each department, and the number of engineers needed to perform any INFO function within each department. Appendix I outlines the resource allocation strategies employed in our analysis.

As we shall discuss in the next chapter, by modifying the number of available engineers in each department, the requirements of each station for departmental engineers, and the priority rules for specific engineer assignments, it is possible to establish an array of alternative engineer allocations. Suitable allocations can facilitate "switching" effects, whereby engineers switch from information station to information station, with MAIN processing done during "free time." Excess engineers can virtually eliminate this switching effect, causing less interesting, less realistic, and more expensive results.

4.2.3. CPP Structural Synopsis

The CPP Structure was developed in response to the inadequacies of existing managerial tools for monitoring and analyzing new product development. Based upon our field study work with the IDEF0 functional modeling methodology, we observed numerous peculiarities with the structure and stability of new product development processes. As insightful as IDEF-based process models can be, they lack dynamic analysis capabilities. The CPP structure picks up where IDEF-like techniques fall short. By combining simulation techniques with well documented IDEF-based process structures, CPP could be the genesis of a new ***dynamic*** analysis tool. A major step in developing a running prototype of CPP was to determine how to incorporate ***very high rates of rework*** into a

simulation program which was oriented around linear production paradigms. In the development of the CPP methodology, we have thus far provided for human resource allocation, recursive information flows, information transfer structures, buffers (stores) of information and prototype materiel, information processing, prototype processing, information and prototype priority methods, and simultaneous processing. A variety of performance measures are associated with these elements and attributes. We have also developed cost and quality methodologies in conjunction with CPP, but do not present them here.

The next section describes some specific results which arose during dynamic analysis of the above described CPP system. It is hoped that the discussion thus far has been clear enough for the reader to understand the terminology and structure whose dynamic properties are described in the upcoming pages.

4.3. Model Analysis/Results

In this section, we recap some significant results from our dynamic analysis of the CPP structure. It is important to keep in mind that the analysis performed here was based upon many judgements and observations obtained in the field studies. Thus, the CPP structure used in this analysis does not specifically describe a particular organization. Rather, it reflects characteristics of nearly every site visited, in some measure.

Because the CPP structure has been developed as a demonstration device, parameters have been "tuned" to reflect field observations. It should be understood that this parameter "tuning" does not employ the full spectrum of potential states in which the

system ***could*** operate. It is hoped and expected that future research will probe outlying parameter regions which are viewed as less acceptable to current management and research arenas. Further, each particular site has particular ***transient*** characteristics which affects its system performance. Thus, a given site, through its own ***changing parameter settings***, may "tune" itself in or out of the system domains observed here. Naturally, this is one of the advantages of analyzing the CPP structure: ***a manager can visualize impacts of changes in a manner never conceived before***. In this regard, this chapter may be forwarded as a template case for future analysis of CPP Structures.

The results of the CPP analysis are segmented into four main parts:

- **Run Overview**
- **Observed Effects**
- **Impact Parameters**
- **Model Validation**

The first segment is a brief review of the formal simulations which have been conducted. This segment introduces parameter terminology, which is useful for the remainder of this results discussion. The second segment relays information about observed CPP dynamics. The third segment discusses parameter changes which were made to this CPP system and review how they impact the system. The fourth segment offers some indication of how well the structure and performance of this model conform to real world observations.

4.3.1. Run Overview

52 formal variations of parameters described in the previous section were independently employed. These have been documented in four volumes of simulation results. In addition to such "finalized" variations, we experimented with countless "developmental" variations. These were used to help establish the CPP structure, to provide direct insight into the underlying dynamics, and to demonstrate exaggerated effects (which further helped to guide parameter values and better explain certain results).

Although every formal and informal development structure was used to gain insight, the final 37 runs ("L" through "ZV" in Table IV.2.) have been used in the statistical analyses and figures presented in this segment of our discussion. These runs used consistent information transfer structures; structures which were variable prior to run "L". This helped us isolate the effects of processing rates and prototype buffer sizes on the CPP system. Naturally, variations to the information transfer structure can be expected to affect the CPP system in major ways. This is a promising area for future research of CPP dynamics.

Table IV.2. is an overview of the chronological development of the "formal" variations of the CPP model.

TABLE IV.2.

CPP Run Variations

RUN ID	VOLUME #	MODEL FORM	DURATION (run time)	INFORATE *Time per*	MAINRATE *operation*	PROTO BUFF	RESOURCES (allocation)
1	1	COMPLEX1	2500	1	1	UNLIM.	N/A
2	1	COMPLEX1	2500	0.5	0.5	UNLIM.	N/A
3	1	COMPLEX1	2500	2	2	UNLIM.	N/A
A	1	COMPLEX3	500	1	1	UNLIM.	3/3/1
B1	1	COMPLEX5	50	1	1	UNLIM.	3/3/1
B2	1	COMPLEX5	2500	1	1	UNLIM.	3/3/1
C	1	COMPLEX6	2500	1	1	UNLIM.	3/3/2
D	1	COMPLEX6	2500	1	1	UNLIM.	4/3/2
E	1	COMPLEX6	2500	1	1	UNLIM.	5/3/2
F	1	COMPLEX6	2500	1	1	UNLIM.	4/3/1
G	1	COMPLEX6	2500	1	1	5	5/3/1
H	1	COMPLEX7	2500	1	1	5	3/3/1
I	1	COMPLEX7	2500	1	1	5	3/3/1
J	1	COMPLEX7	2500	1	10	5	3/3/1
K	1	COMPLEX8	2500	1	10	1	3/3/1
L	1	COMPLEX8	2500	1	10	1	3/3/1
M	1	COMPLEX8	2500	1	10	2	3/3/1
N	2	COMPLEX8	2500	1	10	2	3/3/1
O	2	COMPLEX8	2500	1	10	3	3/3/1
P	2	COMPLEX8	2500	1	10	10	3/3/1
Q	2	COMPLEX8	2500	1	10	5	3/3/1
R	2	COMPLEX8	2500	1	10	1	3/3/1
S	2	COMPLEX8	2500	1	10	7	3/3/1
T	2	COMPLEX8	2500	1	10	20	3/3/1
U	3	COMPLEX8	2500	1	5	3	3/3/1
V	3	COMPLEX8	2500	1	20	3	3/3/1
W	3	COMPLEX8	2500	0.5	10	3	3/3/1
X	3	COMPLEX8	2500	2	10	3	3/3/1
Y	3	COMPLEX8	2500	2	20	3	3/3/1
Z	3	COMPLEX8	2500	0.5	5	3	3/3/1
ZA	3	COMPLEX8	2500	0.5	20	3	3/3/1
ZB	3	COMPLEX8	2500	2	5	3	3/3/1
ZC	3	COMPLEX8	2500	2	6.67	3	3/3/1
ZD	3	COMPLEX8	2500	1	6.67	3	3/3/1
ZE	3	COMPLEX8	2500	0.5	6.67	3	3/3/1
ZF	3	COMPLEX8	2500	0.67	6.67	3	3/3/1
ZG	3	COMPLEX8	2500	0.67	5	3	3/3/1
ZH	3	COMPLEX8	2500	0.67	10	3	3/3/1
ZI	3	COMPLEX8	2500	0.67	20	3	3/3/1
ZJ	4	COMPLEX9	2500	1	10	3	3/2/1
ZK	4	COMPLEX9	2500	1	10	3	2/2/1
ZL	4	COMPLEX9	2500	1	10	3	2/1/1
ZM	4	COMPLEX9	2500	1	10	3	1/1/1
ZN	4	COMPLEX9	2500	1	10	3	3/1/1
ZO	4	COMPLEX9	2500	1	10	3	3/1/2
ZP	4	COMPLEX9	2500	1	10	3	2/1/2
ZQ	4	COMPLEX9	2500	1	10	3	2/2/2
ZR	4	COMPLEX9	2500	1	10	3	3/2/2
ZS	4	COMPLEX9	2500	1	10	3	3/3/2
ZT	4	COMPLEX9	2500	1	10	3	3/3/3
ZU	4	COMPLEX9	2500	1	10	3	3/2/3
ZV	4	COMPLEX9	2500	1	10	3	3/1/3

Rows in the table describe the attributes of a particular dynamic analysis, or "run."

Columns are described as follows:

RUN ID: This delineates the name of the run. Beginning with MODEL FORM = COMPLEX3, this name is designated with a letter.

VOLUME: This identifies in which volume the run results can be found. Within each volume, the run results are arranged in sequential (RUN ID) order.

MODEL FORM: This reveals specific structural information about the model used. Since part of this research involved modifications to the structure, it proved useful to record which structure was being used for a given set of parameters. Descriptions of these structures are listed in Appendix J--***CPP Model Structures***.

DURATION: The amount of time over which a run was conducted. For our purposes, it is convenient to consider each time unit as one working day. Thus, a value of 2500 corresponds to 10 years (assume 250 working days/year).

INFORATE: This figure shows the time required for each INFO station to perform a requested operation. Units are "days per operation." Thus, INFORATE = 0.5 corresponds to an information processing rate of .5 days per operation (conversely this can be interpreted as capable of performing 2 operations per day). The INFORATE indicated in the table was used for all 12 INFO functions in our CPP Structure (3 INFO functions per department times 4 departments). Thus, each department has the same processing capabilities.

MAINRATE: The time required for each MAIN station to perform an operation. It carries the same time units as INFORATE. As with INFORATE, the MAINRATE indicated is used for all 4 MAIN stations in the model (1 MAIN function for each of the 4 departments).

PROTOBUFF: The capacity of the buffers used to hold prototype assemblies. Units are "prototype assemblies." Thus, a PROTOBUFF value of 5 means that each of the prototype buffers has the capability of storing 5 assemblies. In this model, PROTOTYPE indicates the common buffer capacity of ProtoA, ProtoB, and ProtoC buffers. The ProtoD buffer was held very large (500 assy), to have no blockage effect on the system.

RESOURCES: The population and allocation of resources (engineers) at each department. This is composed of three numbers, separated by a '/'. The first number corresponds to the number of engineers employed at a department. The second number corresponds to the number of these engineers required to perform the MAIN function. The third number corresponds to the number of engineers required to perform any INFO function. Thus, 4/3/2 is interpreted as "there are 4 engineers in each department, any 3 of them must be available for the MAIN function to be performed, and any 2 are required for an INFO function to be performed." Engineers are considered generic: any engineer in a department can perform any function within that department. All departments are identical in their resource capability and allocation. For simplicity sake, no further matrixing of personnel (e.g., across departments) is incorporated. The CPP structure can accommodate variants to these simple possibilities.

4.3.2. Observed Effects

The results of these runs were voluminous. This report presents a small subset of the data and information obtained during our analysis. For this discussion, we shall focus on the system-wide effects of changes. This is in contrast with common practice, which focuses on local effects (i.e., from the engineer's or manager's point of view). Local effects are most immediately evident among most participants in the process. However, broad corporate policies seem to develop from "overall" results of the process, regardless of localized effects. Further, the performance measures of the development process which we are concerned with involve ***effectiveness***. Individual ***efficiencies*** by departments or local stations may be interesting, but cannot be considered valuable unless we can show their impact on the effectiveness of the development system.

In our analysis of CPP behavior, we considered a variety of measures. Predominant measures in the field may be placed into three basic categories: ***Responsiveness***, ***Quality***, and ***Cost***. For this report, we shall focus on the responsiveness aspect of system performance. Measurements of quality are inherently subjective, despite numerous quantitative assessments in the field. We have formulated some interesting quality assessment strategies for use with the CPP structure, however. For more information on these formulated strategies, refer to Appendix K.

The dynamic CPP analysis lends itself very nicely to analyzing a variety of costs. This is particularly true for labor intensive processes such as new product development. For such processes, cost can be well accounted for (ex post) by multiplying labor rates times the time (hours/days/etc.) spent on each function. Such functionally-based cost accounting principles are well established, although not always executed as well as one might desire.

Activity-Based Costing (ABC) or the more general Activity-Based Cost Accounting (ABCA) strategies can be easily incorporated into the CPP Structure[62]. As mere multipliers of activity time, however, such cost analyses are not interesting enough to us to warrant further discussion. In effect, costs indicators can be considered *dependent upon* the more fundamental time, information consumption, and quality results discussed here. In the field, such cost analyses can and should be conducted.

We consider any time-based measure during development to be a viable candidate as a responsiveness measures. As indicated earlier, however, we are more interested in effectiveness measures, not necessarily efficiency measures. A natural measure, then, may be ***chronological development time***. As we'll consider in a few pages, this is not such a straightforward measure.

Another responsiveness measure apparent in the field is ***engineering time***. Given that engineers are spending low proportions of their time on engineering tasks, this "obviously" should contribute to ineffective chronological development time. Per our analysis, we found this to be true...sometimes.

Per the field studies, we observed that information processing dominates development. One common conclusion often drawn from this is that improved information systems should help reduce the total time dedicated to such information processing. An interesting

---

[62] This does not, by any means, imply that cost and time measures or methodologies are simple to ***implement***. There are numerous (and depending on one's demeanor, humorous) detailed accounts about the problems of organizations and researchers attempting to adopt consistent, agreeable paradigms for assessing the costs and elapsed time of development. In contrast to objectively defining quality, however, these two classes of measure are much more clear.

measure, ***information consumption***, utilized in this analysis indicates whether this is really so. Though not a direct time measure, information consumption is an indicator of the amount of non-engineering effort being expended within the engineering organization.

Thus, we present three categories of effective responsiveness measures in this report: chronological development time, engineering time, and information consumption.

4.3.2.1 Chronological Development Time

We consider two chronological time measures: ***Design Throughput (DT)*** and ***Time to First Release (TTFR)***.

4.3.2.1.1. Design Throughput (DT)

DT is an indicator of how many final designs have been released from the system over a controlled time period. In this analysis, the unit for DT is "# of designs". In the preponderance of runs, the control time period is 2500 days. Assuming 250 working days per year, this is a rather long 10 years. Though a decade is more than many analyses warrant, this time period was selected for several reasons:

- *First*, the system undergoes a natural "warm-up" period, under which no output occurs. The control time periods should include and exceed this warm-up time, so that intervals of subsequent design release dates can be observed.
- *Second*, the effects of variation are difficult to monitor over short control periods. Since some of the effects appear to have pseudo-periodicity measurable on the order of years, a longer control time provides better observation of overall effects.

- *Third*, many managerial outlooks are oriented towards short-term, high-impact results, with little regard for longer-term 2nd and 3rd order impacts. Attempting to understand such higher-order effects is a major focal point of this research.

In the simulation runs, DT ranged from 0 to 45 designs, depending on the parameter settings. For those runs where more than one design was released in the controlled 2500 day period, the mean number of days between releases ranged from a best-case of 55.6 days to a worst-case of 500 days.

4.3.2.1.2. Time to First Release (TTFR)

TTFR is an indicator of when the *first* production-ready design is complete, given that we begin with an "empty" development system. The unit of measure for TTFR is time, days in our case. This measure is used to demonstrate the difference between development of the first design and all of its subsequent revisions. It is significant to realize that the TTFR value for all runs was always longer than the average release interval over the control time. In fact, the ratio of TTFR to average release interval was never less than 1.32 (max = 3.85). This means that the first release always took between 32% and 285% longer to develop than the average interval over the control time[63].

In production and distribution systems, it is natural to experience such delays in release of the first product. Often referred to as "pipeline filling", this delay phenomena is regarded

---

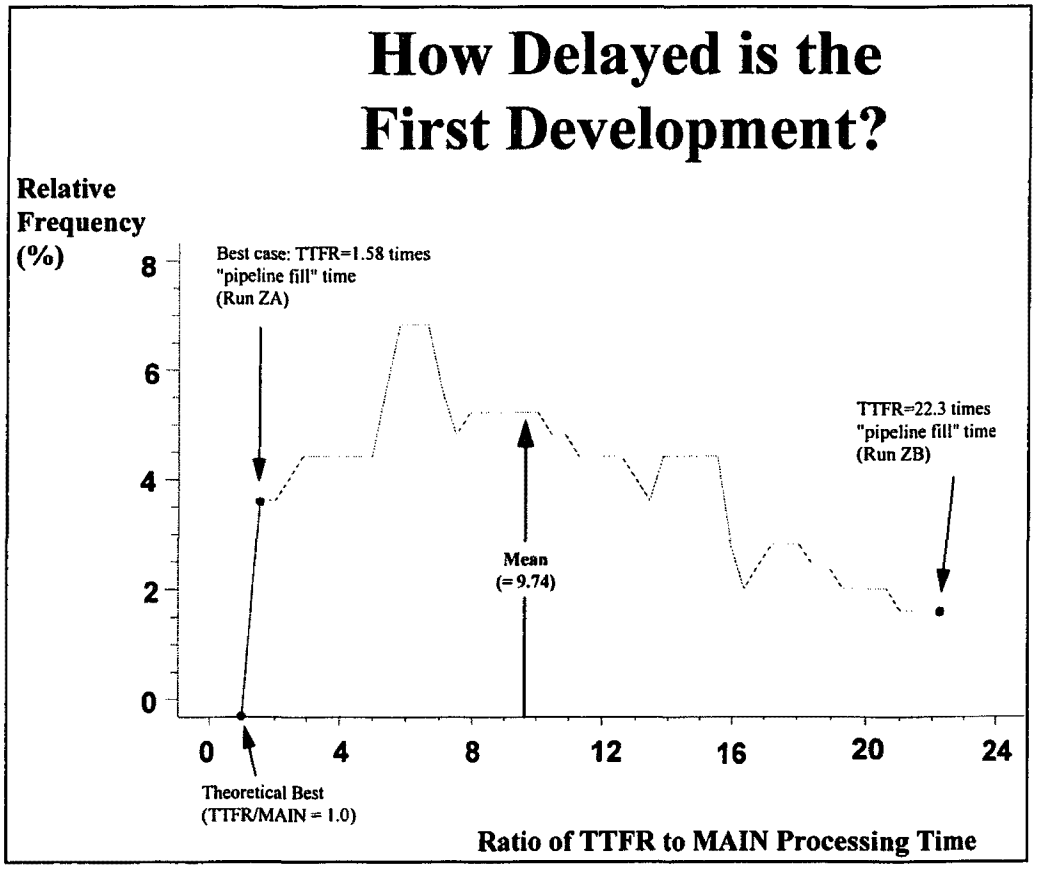
[63] It was not unusual, however, for any single release interval to be greater than the TTFR. This appears to be a function of the information transfer dynamics which occur throughout the development process. In simple terms, this means that the departments were sometimes so backlogged that "windows of opportunity" to perform MAIN functions were few and far between.

as the time necessary for the product to travel through the system. This is fully expected, even for this CPP structure.

Note, however, that TTFR is usually much longer than the arithmetic sum of the individual MAIN processing times. The boxcar density plot in Exhibit IV.8. shows that TTFR was ***as much as 22 times longer*** than total MAIN processing time. For the simulation runs in this study the TTFR was, on average, 9.74 times greater than one would expect through mere pipeline filling. Such filling has some responsibility for longer first release. This component is relatively minor, however, when compared to the total time spent performing "non-development" activities. Our next set of system effectiveness measures, ***Engineering Time*** and ***Information Consumption***, help us understand ***why*** this is so.

TTFR values for the 24 finalized COMPLEX8 runs ranged from 91.5 days to 875 days. Contrast this with theoretical best-case pipeline fill times, which ranged from 20 days (when INFORATE = 5) to 80 days (when INFORATE = 20).

EXHIBIT IV.8.



4.3.2.2. Engineering Time

When considering the TTFR and DT for a development system, it is important to consider how much time is spent performing value-added activities. In this analysis, the MAIN functions are considered value-added; the INFOx functions are considered accessory. Thus, though 2500 days elapsed during development, only a portion of that time was spent on the MAIN (development) functions. The amount of time spent on the MAIN functions is regarded as ***engineering time***. Depending upon the throughput of

designs, and the dynamics of information transfer regarding such design, the engineering time varied considerably.

In this analysis, engineering time is expressed as a ***percent*** of engineer's time. Using the 3/3/1 resource allocation (where all engineers within a department are needed to perform that department's MAIN function), this coincides with the ratio of average MAIN activity time to duration (t=2500 days). For other resource allocations, engineer's idle time lowers the time proportion of *both* MAIN and INFO functions.

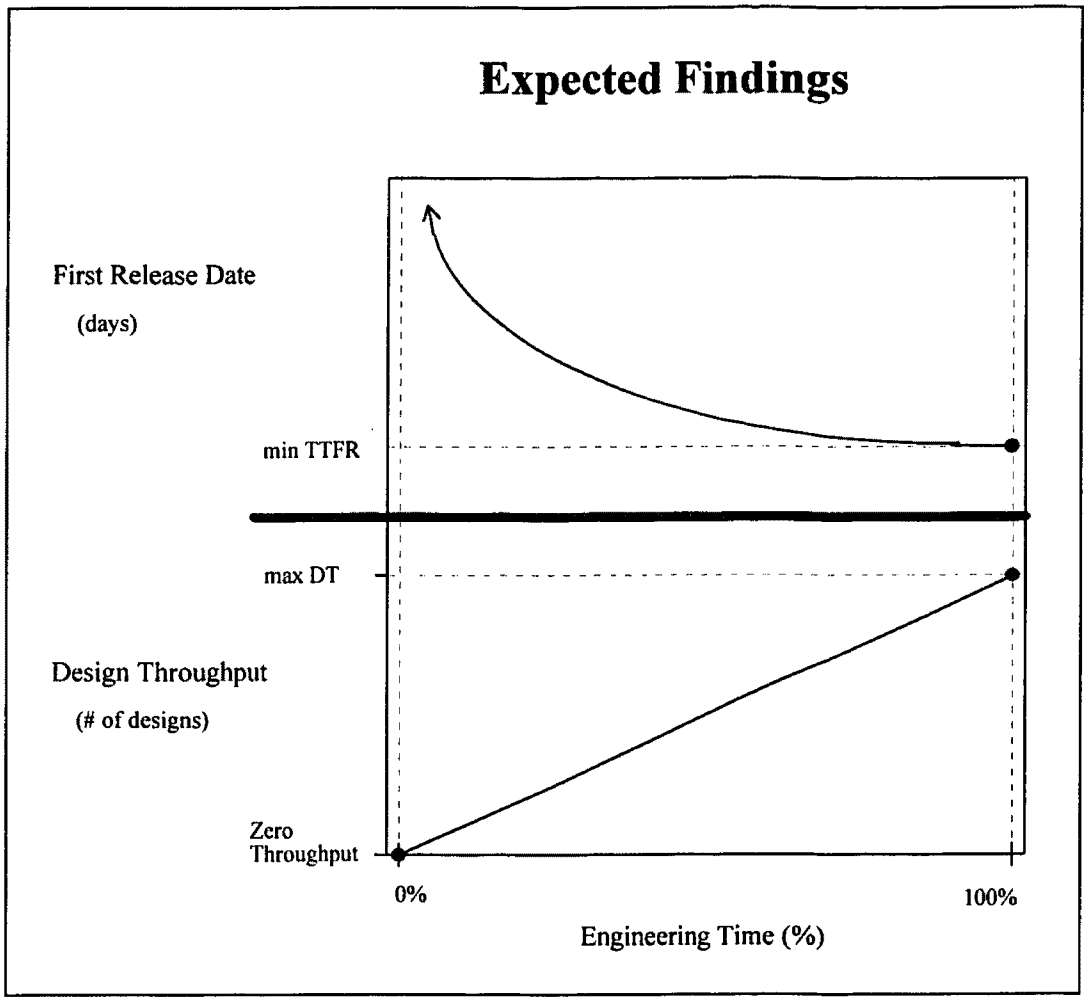
Engineering time in this dynamic analysis ranged from 2.1% to 29.4%. This compares well with the results of a comprehensive survey at one field site, in which developers spent approximately 15% of their time performing "value-added engineering" tasks.

The ***appropriate*** level of engineering time is a less straightforward issue than expected. It was expected that engineering time would be directly related to first release date and design throughput in the following manner:

- TTFR is inversely proportional to MAIN processing (i.e., TTFR $\propto$ 1/eng. time)
- Design throughput is proportional to MAIN processing (i.e., DT $\propto$ eng. time)

These expectations are graphically illustrated in Exhibit IV.9.

EXHIBIT IV.9.



With such an expectation, TTFR and DT would be optimized by maximization of engineering percent. Since we know that 100% engineering time is impossible (due to our "requirement" for some INFO processing), it is nonetheless expected that TTFR and DT would be maximized by increasing engineering percent as much as possible.

It was demonstrated, however, that DT and TTFR are not so linearly dependent upon engineering time. Refer to Exhibit IV.10. and IV.11. Increasing engineering time (as a

percentage of total time) does not necessarily decrease TTFR or increase DT. Likewise, decreasing engineering time does not always adversely affect these chronological development measures. Although there may be a vague accordance with expectations, TTFR and DT seem to be dependent on more than this simple indicator.

EXHIBIT IV.10.

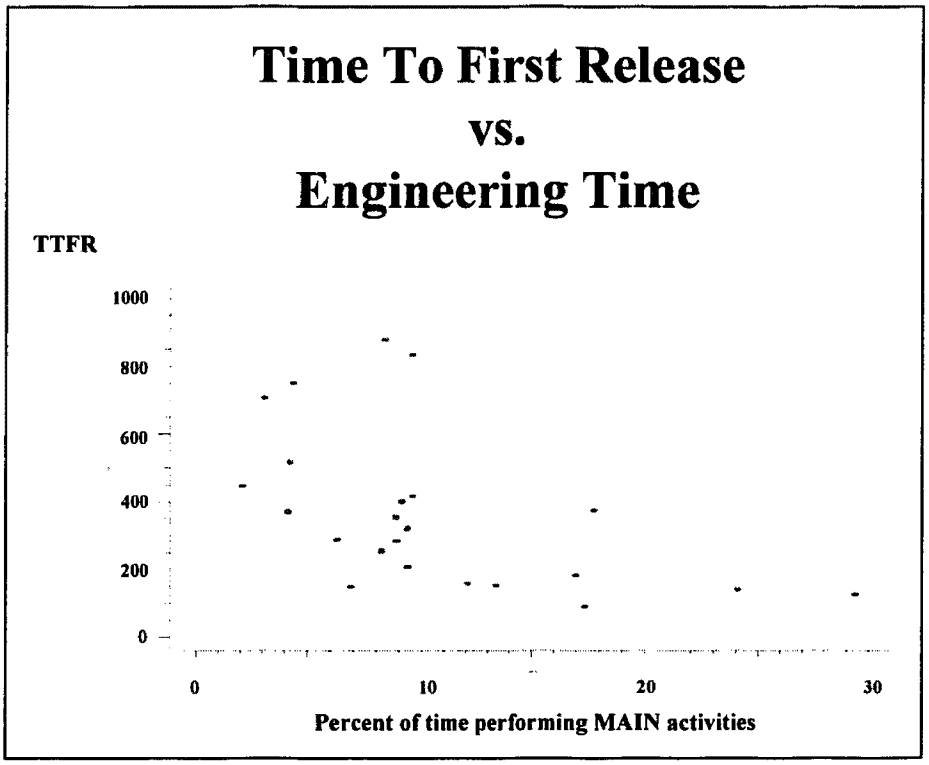
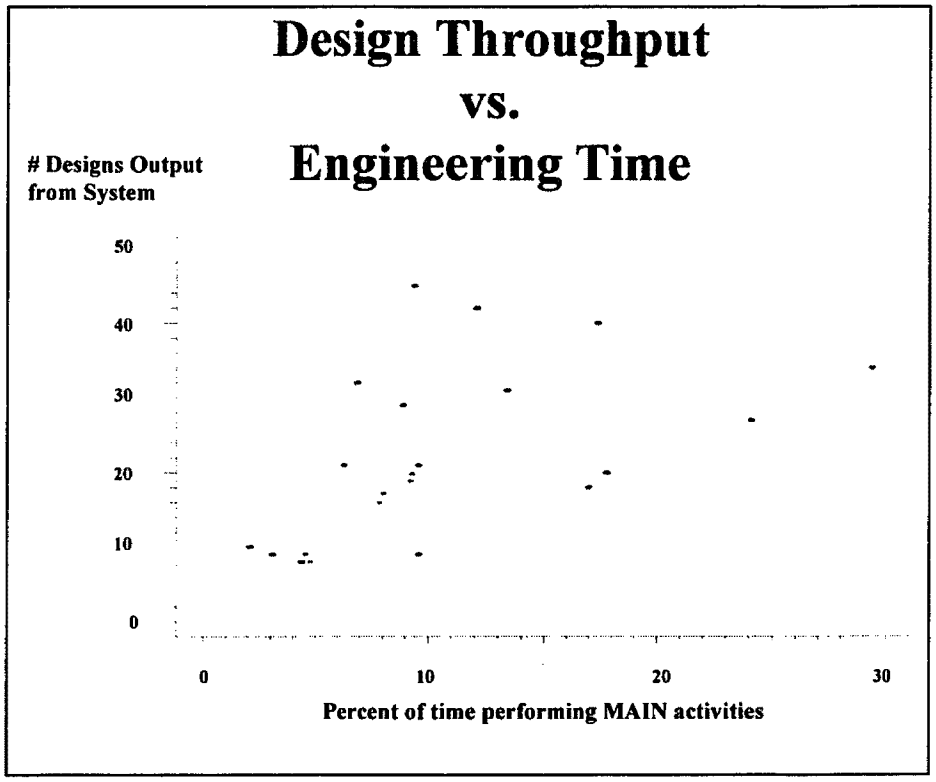


EXHIBIT IV.11.



Certainly, vast increases in engineering time, say from 2.1% to 24.1% do seem to generate better DT and TTFR results. Yet, the "path" to getting to such different regimes is quite "rocky." Imagine being a development manager trying to raise engineering percent to the 24% or 29% level. Given the great variation in system performance along the way, it is questionable whether one could stand the painful bouts of "bad" performance which will occur during this transition.

This strange relation between engineering time and our chronological time measures seems to have roots in two factors: ***communication bundle size*** and ***station processing rates***. Communication bundling sizes are the number of communications (information

and prototypes) which are sent at the conclusion of each station's processing. Per our information transfer protocols described earlier, however, we have held this parameter constant throughout our analysis. Station processing rates (including MAINRATE and INFORATE) are parameters which have been analyzed in detail and are presented as major issues in the ***Impact Parameters*** segment of this chapter.

4.3.2.3. Information Consumption

The output of each INFO function is more information. Thus, we can characterize information processing as a churning effect. When one INFO function sends information to another INFO function, the latter INFO function processes it. Depending upon whether the latter function is in the ***dissipation***, ***return***, or ***re-direction*** mode (refer to our CPP structural description for clarification on these terms), the information base is reduced, held constant, or increased[64]. Whenever the information base is reduced, some information is lost. [65] In our model, consumption is expressed in "units of information". This is similar to ***bits***, which is used in information theory. Since we have characterized all information in this model as generic, all dissipated information is equivalent. Thus, as long as the units for information consumption are consistent, we may use bits, bytes, or any other convenient measuring unit.

---

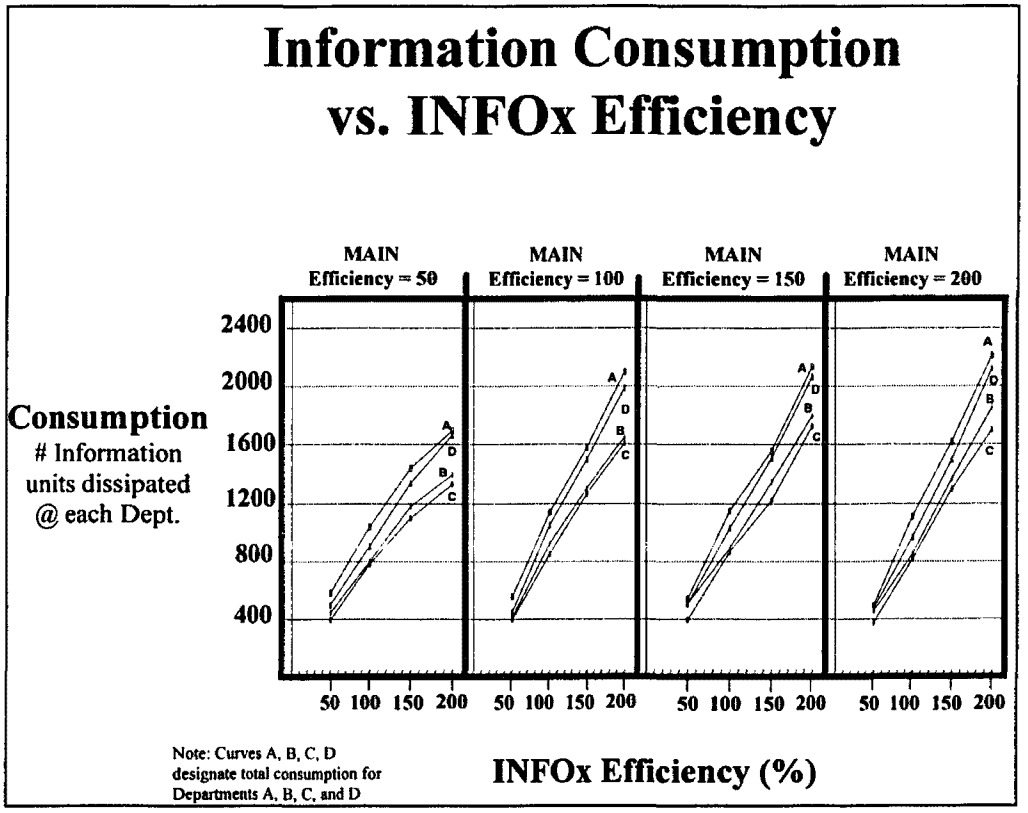
[64] Technically, we have limited the re-direction mode to production of no more than one information output, so the INFO functions do not partake in "creating" more information than they recieve. If not balanced appropriately, such information production can bring about a degenerative system, in which the system continues to produce and churn more and more information until one or more information buffers are full. At that point, or soon thereafter, the system ceases to operate.

[65] In Information Theory (see Shannon, 1949) the resulting loss of certainty in messages is known as *information entropy*. Since we are not currently making judgement over the merits of such loss, we merely it as *information consumption*.

The degree to which such consumption occurs was carefully recorded. With the model in a "mature" development mode (COMPLEX8), information dissipation was reviewed. Specifically, the variations of information consumption in response to information efficiency, MAIN efficiency, and buffer size changes were reviewed.

It is apparent that information processing rate and information consumption are positively correlated (i.e., faster processing correlates with higher degree of consumption). Refer to Exhibit IV.12. This was true for every control level of MAIN efficiency.

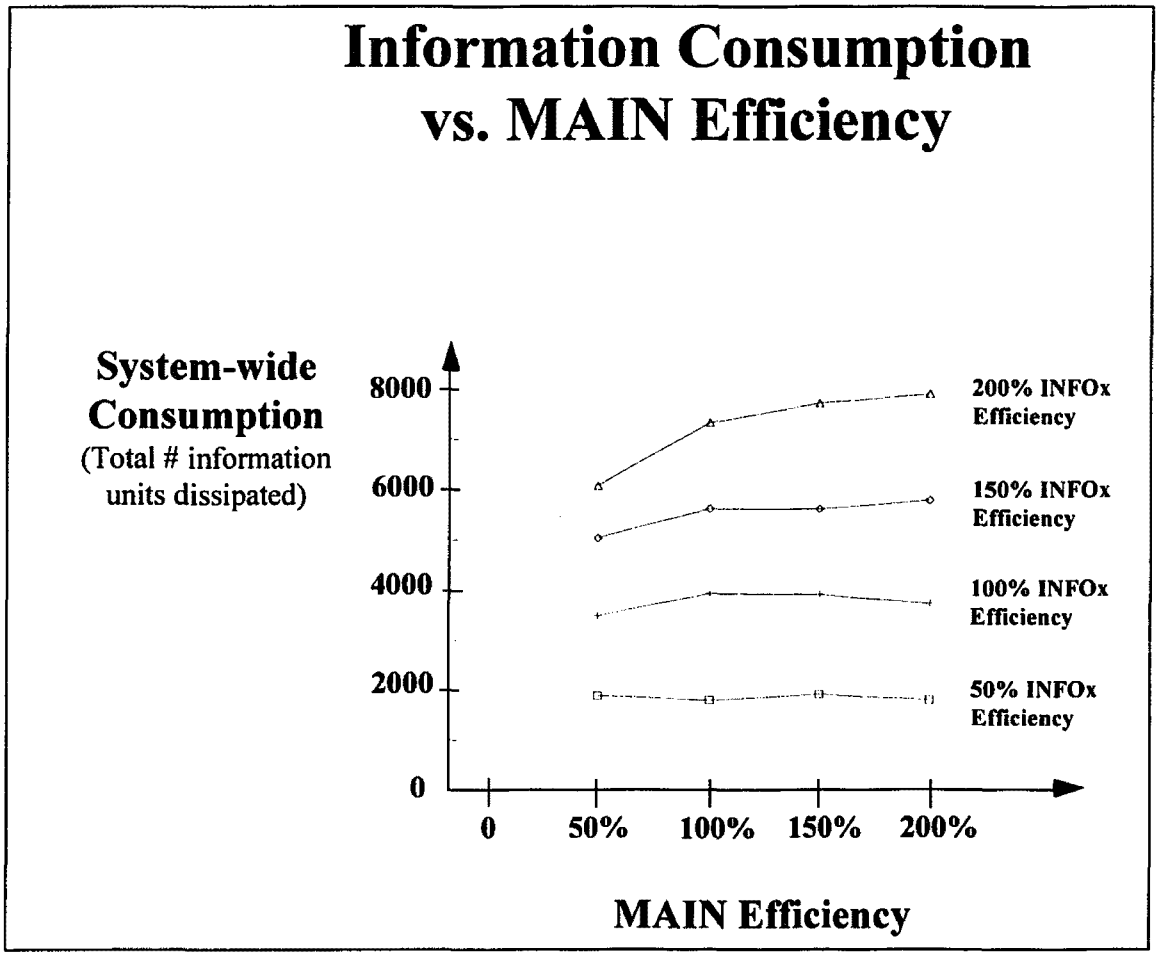
EXHIBIT IV.12.



Beyond mere correlative tendencies, the underlying cause for this was made evident during visual observation of the runs. For high levels of information processing efficiency (low INFORATE), the ability of INFO stations to process information over any time unit improves. Since information is only generated by the MAIN functions, this "improved" processing ability also applies to the rate at which information is dissipated. Thus, the dissipation rate, consumption, increases.

However, convexity differences are noticeable as one compares consumption curves of different MAIN processing rates. Refer to Exhibit IV.13. MAIN efficiency impacted variations in information consumption are virtually eliminated when information processing efficiency is low (refer to bottom curve). Conversely, high information efficiencies demonstrate increased variation in consumption. At 200% information processing efficiency (i.e., INFORATE = 0.5), the consumption level is an ordinal function of MAIN efficiency. (See the top curve in Exhibit IV.13.) This is a result of more "windows" of opportunity to produce information. Even so, this effect appears to be met with diminishing returns as MAIN processing efficiency continues to rise. Notice the ever shallower slope of the 200% curve, for each improvement in MAIN efficiency.

EXHIBIT IV.13.

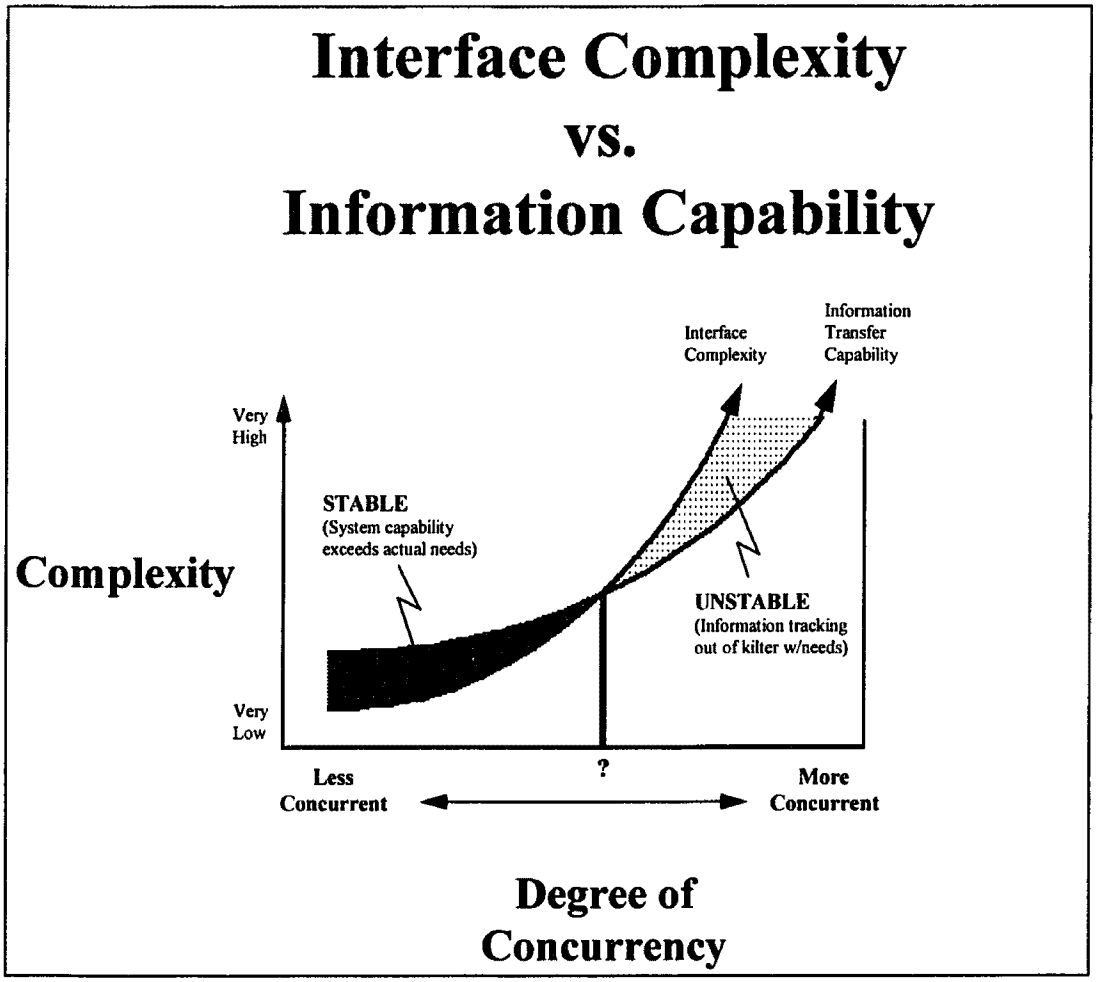


If this were a real development system, management would be in a dilemma. Increasing information processing efficiency permits MAIN functions more opportunity to "create" information. This, in turn, increases the requirement for more information processing. Thus, information backlogs can still be significant. Yet, decreasing information processing efficiency does not help the process move along. It merely increases the time necessary to process information. Thus, the information backlog is existent during low

information processing efficiencies regimes. In this simple feedback system, it is clear ***that information backlogs will always be existent, regardless of the information processing efficiency!***

We postulate that average information backlogs asymptotically approach some non-zero limit. If true, then future efforts to break this limit would prove fruitless, regardless of technology. In our discussion of information findings from the field, we contemplated the nature of ***interface complexity*** and ***information transfer capability***. By increasing information processing capability, it may be possible to iteratively and unconsciously fall into the unstable region illustrated in Exhibit IV.14. This could happen if interface complexity increases faster than such information processing improvement.

EXHIBIT IV.14.



In the model, such behavior occurs when MAIN processing increases unbridled, relative to INFO processing. Through the use of our "information first" priority structure and limited prototype buffer sizes, we have bounded this problem from degenerating the system. When removing the prototype buffer restriction, however, this problem was shown to quickly recur. In the model, this is manifested as follows:

> When the buffer size of protoA is unrestricted, department A is always "one step ahead" of the other departments. Dept. A is able to do this because of its

consistent production of information during every prototype release. Even though engineers at Dept. A do perform some information processing, they are not asked to do as much as "downstream" departments. Thus, they can spend more time producing ***more*** prototypes, an activity which produces even ***more*** information.

Meanwhile, the ProtoA buffer contents continually expand, as engineers at Department B have less and less opportunity to process any of the prototypes sitting in the ProtoA buffer. This problem cascades sequentially down the system in a manner such that ***NO*** prototypes ever progress to Department D. Under some scenarios, even Departments B and C fail to get the opportunity to process even a single prototype.

In the real world, this scenario runs counter to traditional thinking in the Management Information Systems (MIS) arena, which advocates that improved information processing capability will result in better system-wide performance. Yet, our simple model shows that ***information feedback can temper the gains of improved processing***. In other words, improvements in information processing may only offer temporary, local improvements in work flow. With feedback, a high-efficiency system may have the ability to naturally increase the information backlog.

Of course, this phenomenon is dependent on the specific information structure in place. For low feedback systems, this characteristic may vanish altogether[66]. For high-feedback systems with ill-described structures, such as innovation-oriented new product development, this characteristic is readily and repeatedly apparent[67].

---

[66] Another viewpoint here is that this characteristic merely gets camouflaged better in low feedback systems, with its impacts felt in a more global sense, such as lack of future sales.

[67] In fact, researchers at the Brookings Institute and at Harvard have discovered that automated information processing technologies have, at times, contributed to this computational/productivity paradox in a variety of industries. For more information on such studies, refer to Baily and Chakrabarti (1988).

4.3.3. Impact Parameters

During construction and execution of the dynamic CPP model, we experimented with several parameters. By holding all other parameters constant, we could observe the effects of a single parametric change. The following four classes of parameter variations are discussed here:

- Engineering Resource Allocation
- MAIN Processing Efficiency
- INFO Processing Efficiency
- Buffer Sizes

4.3.3.1. Engineering Resource Allocation

Nineteen different allocations of engineers within a department were developed and simulated. The first five allocation experiments were used to gather a general flavor for the effects of resource allocation. More for internal use in generating a realistic model, these five runs constitute the COMPLEX6 model runs. Because the information transfer rules for the COMPLEX6 model form were different from those of our more developed "control" model form--COMPLEX8, the results of these first five test runs are not presented in this discussion.

Using our "control system", beginning with run "L", fourteen resource allocation strategies were formally experimented with. This control system constituted the COMPLEX8 system, with INFORATE=1 and MAINRATE=10. For the resource allocation runs, this model form was renamed "COMPLEX9." (Technically, run "O" is also a resource allocation run, although its model form = COMPLEX8.) Although there

can exist infinite alternative allocations of engineers, we have limited our formalized strategies to all of those possible with 3 or fewer engineers per department.

Recall, our nomenclature for allocation in this CPP model takes the form **"T/M/I"**

where

T = The total number of engineers employed by each department;

M = The number of engineers which are required to perform the MAIN function within the department;

I = The number of engineers required to perform any INFO function within the department.

Also, recall that our CPP structure is composed of four "departments." Each department has four functions: 1 MAIN function and 3 INFO functions. For simplicity sake, we have further assumed that resource allocations are the same from department to department[68]. Thus, our T/M/I designation describes the allocation of all engineers at all departments.

---

[68] This constraint can be easily removed for any particular anlalysis that one might wish to consider. It merely has the effect of limiting the alternatives that we present in this analysis, to aid clearer communication of the resulting phenomenon.

Using our maximum employment of 3 engineers per department (12, system-wide), we have 14 different allocations. They are as follows:

Engineers per Department	Possible Allocation Strategies
Single Engineer:	1/1/1
Two Engineers:	2/1/1, 2/1/2, 2/2/1, 2/2/2
Three Engineers:	3/1/1, 3/1/2, 3/1/3, 3/2/1, 3/2/2, 3/2/3, 3/3/1, 3/3/2, 3/3/3

Appendix I, ***Engineering Resource Allocations,*** describes, in table form, the purpose and run variation details for each of these 14 allocation strategies. We consider two integral aspects of engineering allocation: ***concurrency*** and ***utilization***.

4.3.3.1.1. Concurrency Effects in Engineering Resource Allocation

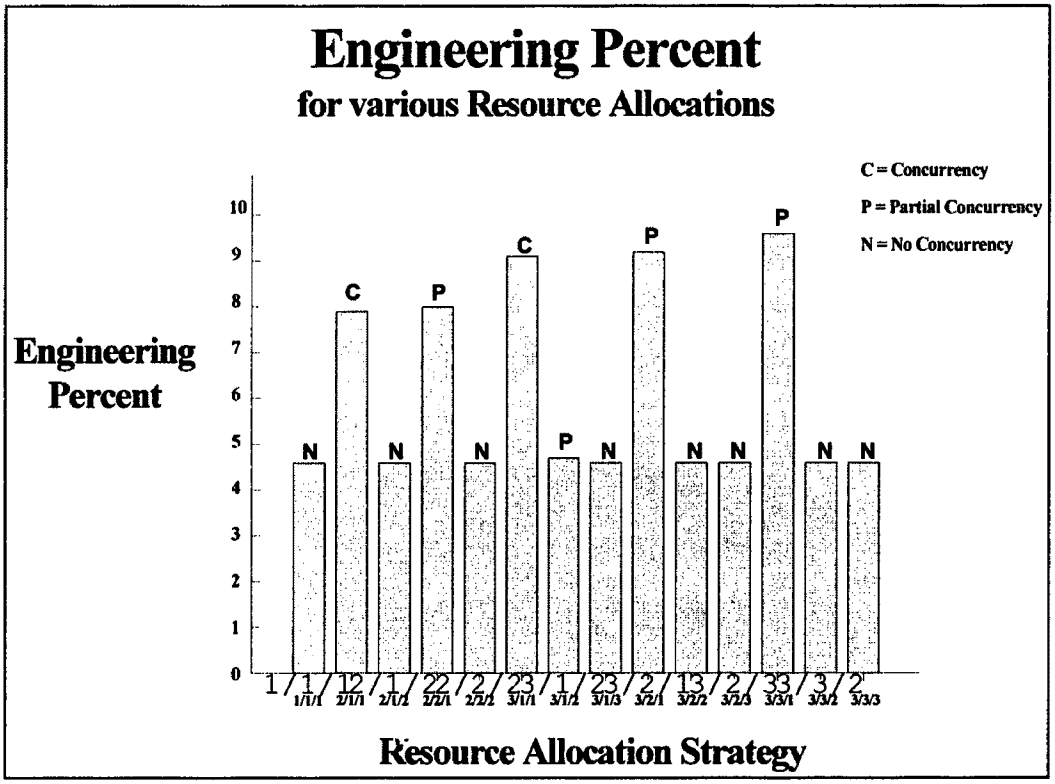
The CPP model always permits concurrency among departments. Even the 1/1/1 strategy allows up to four engineers (one from each department) to work simultaneously[69]. Concurrency *within* departments, however, is limited by the particular resource allocation strategy. With eight of the fourteen allocation strategies, intra-department concurrency is *not* permitted. This is true for the 1/1/1, 2/1/2, 2/2/2, 3/1/3, 3/2/2, 3/2/3, 3/3/2, and 3/3/3 strategies. Of the remaining six strategies, only two offer complete concurrency

[69] Such concurrency among MAIN functions, of course, is possible only if there exists work (prototypes) to be processed by all the departments simultaneously. Thus, one can easily see that the first prototype has little or no concurrency at early stages and increased concurrency as it moves downstream. For latter design releases, more concurrency is feasible.

(autonomous operation) among department functions: the 2/1/1 and 3/1/1 strategies. The resource allocation table provides conditional concurrency information about these fourteen strategies.

The effects of concurrency are immediately evident upon review of Exhibit IV.15.

EXHIBIT IV.15.



In this illustration, the percentage of engineering time is plotted for each of the 14 resource allocation strategies. For those allocations where concurrency was enabled, the engineering percentage was up to twice that of non-concurrent allocations. This is directly attributable to the fact that concurrent operations enable faster net information

dissipation per department. Direct comparison between the 2/1/2 and 2/2/1 allocations demonstrates this phenomena:

- With the 2/1/2 allocation, engineers cannot perform any two functions simultaneously. When the MAIN function is being performed by one engineer, the other engineer is forced to remain idle--s/he needs the other engineer to "help" process information. Thus, when the department is in MAIN processing mode, only 50% resource utilization is apparent. When in INFO processing mode, on the other hand, resource utilization rises to 100%--both engineers are working together on a single INFO function. Thus, information received from other departments[70] can only be processed one at a time.

- Contrast this with the 2/2/1 allocation. When the MAIN function is being processed, both engineers are working--both on the MAIN function. When information processing is needed, the MAIN function is abandoned; engineers can then "split" and perform two different INFO functions simultaneously. Thus, the net INFO processing rate (for the department) is up to double the rate observed during the 2/1/2 allocation. (It is only double under conditions where two or more different information types are available to be processed. Thus, the real gain in available engineering time is actually less--only about 74% better in this case.) Increased information processing (information dissipation, in this model[71]) helps provide more opportunity for the MAIN function to be performed, increasing the engineering time.

Recall that MAIN processing is the generator for "new information" to be added to the system. ***Thus, increases in MAIN processing efficiency and opportunities to perform MAIN processing also create more information to be processed by all INFO functions.***

---

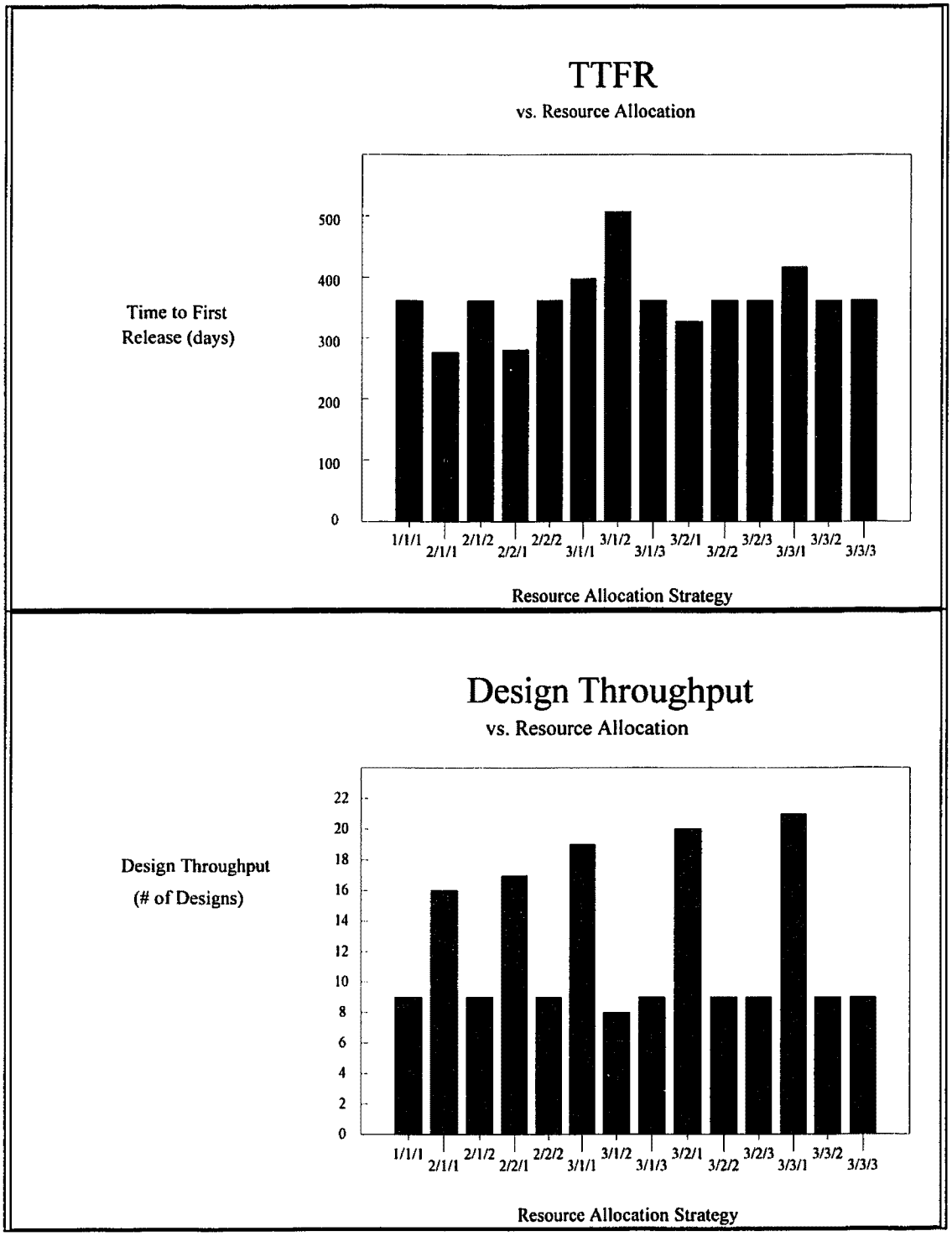
[70] Recall that our use of "department" may be also be construed as a design "phase." Because we have assumed that each department is responsible for a given phase, and because prototype flow (but not information flow) has been illustrated as sequential, we use "department" and "phase" interchangeably here. As we shall discuss later, this reflects the observation that sequential phases, particular with respect to information processing, are not representative of actual development operations.

[71] Once again we must stress that the INFO functions have been deliberately biased towards dissipating information, not creating information. If and when a function is in "creation mode", however temporary, it is creating more work for itself and other INFO functions to process, and (hopefully) dissipate.

This may temper the gains of departmental processing improvements from resource allocations. The apparent leveling effect of the concurrent allocations with increased resources is reflected in Exhibit IV.15. Interestingly, such leveling effects are reminiscent of Cobb-Douglas production functions and Hicks technology functions, for which coefficients have been historically developed from aggregate analyses. The self-leveling phenomena just described may be a useful *generator* for these more global observations of development and innovation.

Regardless of whether this is true, however, we are left with the convincing observation that resource allocations can affect engineering time in the CPP structure. Coupled with this observation is the finding that ***inappropriate use of additional resources can give the same system performance*** as having only one resource per department. Note that the 1/1/1, 2/1/2, 2/2/2, 3/1/3, 3/2/2, 3/2/3, 3/2/3, and 3/3/3 allocation provide identical system performance, as measured by engineering time, first release date, and design throughput. For allocation effects on TTFR and DT, refer to Exhibit IV.16. We shall return to TTFR and DT impacts from resource efficiencies later in this chapter.

EXHIBIT IV.16.



4.3.3.1.2. Resource Utilization Effects in Engineering Allocation

This discussion demonstrates an oft-observed condition where traditional efficiency measures can conflict with our effectiveness measures.

Among managers, particularly at the smaller sites visited, emphasis has been placed on maximizing the utilization of human resources. By definition, this means keeping resources "busy" as much as possible. This can also be interpreted as a strategy of "minimizing resource idle time." (In this sense, "idle" constitutes time spent ***waiting*** for upstream functions to catch up, time spent ***blocked*** by downstream functions, as well as time spent not working for non-systematic reasons such as lunch, coffee breaks, vacation days, sick days, etc.,.) There is something "nice" about the concept of maximizing resource utilization. It seems "right" that increases in resource utilization should result in increased system performance. After all, if all the parts of the system are working more, the system should be working more, right?

Manufacturing managers and industrial engineers have been formally oriented about this concept since the turn of the century days of Frederick Taylor[72]. Methodologies such as line balancing, PERT, JIT, CONWIP, SPC, and many others have been developed to try to maximize resource utilization. These have been developed as methodologies to aid throughput and minimize costs. For human and capital resources (in the form of equipment and non-tangible pecuniary investments), engineering economics methods

---

[72] An overview of the Taylor system of management, which advocates separation of planning functions from execution functions (via specialization of labor) to increase productivity, refer to Juran & Gryna (1988). A compilation of Taylor's writings can be found in Taylor (1947).

have been applied to extract as much return from each resource as possible. For linear processes (including *sequential* and *concurrent* processes), like many manufacturing systems, resource utilization focus has resulted in continuous improvements in manufacturing efficiency and effectiveness[73]. ***For systems that, in principle, resemble our CPP system, however, a resource utilization orientation can be very misleading.***

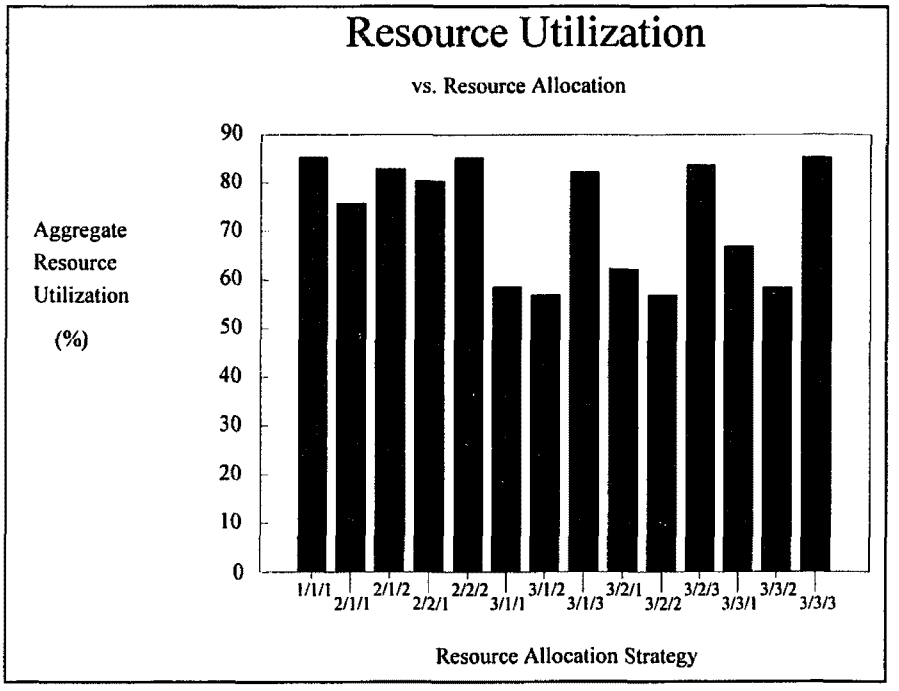
Refer to Exhibit IV.17., which demonstrates the resource utilization for each of the fourteen control allocations[74].

---

[73] Even in the manufacturing environment, the resource utilization argument is beginning to fall on deaf ears. Eli Goldratt presents excellent, easy-reading case studies of the false pretense of utilization efficiency as a method for obtaining throughput in even linear manufacturing systems. Two of his works, The Goal (Goldratt & Cox, 1986) and It's Not Luck (Goldratt, 1994) are must reads for anyone wishing to explore the ever looming problems of mixing efficiency with effectiveness.

[74] Resource utilization is defined as the percent of time that an engineer is "busy" working, whether it be on engineering (MAIN) or non-engineering (INFO) functions.

EXHIBIT IV.17.



Notice that there is no straightforward pattern which links allocation strategies to resource utilization in this CPP system. In fact, allocations which permit intra-department concurrency demonstrate somewhat ***lower*** resource utilization. Note the resource utilization levels for the 3/1/1, 3/2/1, and 3/3/1 allocations. Yet, we demonstrated that engineering time and throughput both increase with more concurrent allocations! How is it possible that ***more throughput is possible with lower overall resource utilization?***

The answer lies in the structure of the CPP system itself. Recall that each department has one MAIN (prototype processing) function and three INFO (information processing) functions. Thus, any functional concurrency which is observed within a development always involves at least one INFO function. Because of the processing priority rules in

this CPP structure, the MAIN function cannot be performed while any INFO processing is still being conducted. Thus, although an allocation structure may ***provide*** for concurrency between the MAIN function and at least one INFO function, such MAIN-INFO concurrency does not actually occur. Only INFO-INFO concurrency is permitted. Resource allocations which hope to provide MAIN-INFO concurrency merely result in reduced resource utilization. This is clearly apparent in the 3/1/1, 3/2/1, and 3/3/1 allocations. For our allocation strategies, maximum instantaneous resource utilization values are listed in the left hand side of Table IV.3.

TABLE IV.3.

RESOURCE UTILIZATION RATES FOR VARIOUS ALLOCATIONS

	MAXIMUM INSTANTANEOUS UTILIZATION OF ENGINEERS				PERFORMANCE MEASURES			
Allocation Strategy	The MAIN function	Any 1 INFO function	Any 2 INFO functions	All 3 INFO functions	Actual Utilization	Eng. Time	DT	TTFR
1/1/1	100%	100%	N/A	N/A	85.25%	4.6%	9	362
2/1/1	50%	50%	100%	N/A	75.80%	7.9%	16	276
2/1/2	50%	100%	N/A	N/A	82.95%	4.6%	9	362
2/2/1	100%	50%	100%	N/A	80.35%	8.0%	17	281
2/2/2	100%	100%	N/A	N/A	85.25%	4.6%	9	362
3/1/1	33%	33%	67%	100%	58.55%	9.1%	19	397
3/1/2	33%	67%	N/A	N/A	56.95%	4.7%	8	507
3/1/3	33%	100%	N/A	N/A	82.20%	4.6%	9	362
3/2/1	67%	33%	67%	100%	62.28%	9.2%	20	327
3/2/2	67%	33%	N/A	N/A	56.85%	4.6%	9	362
3/2/3	67%	100%	N/A	N/A	83.73%	4.6%	9	362
3/3/1	100%	33%	67%	100%	66.93%	9.6%	21	417
3/3/2	100%	67%	N/A	N/A	58.38%	4.6%	9	362
3/3/3	100%	100%	N/A	N/A	85.25%	4.6%	9	362

Upon reviewing this table, we can see how utilization rates can be deceptive indicators of performance. Consider a few examples drawn from the table above:

- The 1/1/1, 2/2/2, and 3/3/3 strategies, when executed, all returned the same results in throughput (DT=9 designs) and time to first release (TTFR=362 days). Yet, the 2/2/2 strategy costs twice as much and the 3/3/3 strategy costs three times as much. At 85.25% utilization rate, the engineers in each of these system scenarios could be considered quite busy.

- The 2/2/1 strategy returned better results in throughput (DT=17) and time to first release (TTFR=281 days). Yet, the resource utilization rate dropped to 80.35%. This corresponds to 89% better throughput and 22% shorter first release date when working an average of 25 minutes *less* per day.

- The 2/1/1 strategy returned similar improvements (DT=16, 78% improvement; TTFR=276, 24% improvement) by working at a measly 75.8% utilization rate. At 75.8%, each engineer could effectively take an additional month off from work every year, compared to the first scenario!

- The 3/3/1 strategy produces the best throughput of all the tests shown here, with 21 release designs (133% increase). Yet, the first release date suffers, with TTFR rising to 417 days (15% slower). Remarkably, resource utilization is only 66.93%!

- With the 3/1/1 strategy, the utilization rate drops even further, to 58.55%. Design throughput is still high (DT=19, 111% improvement over the first scenario), while the first design release date slips by 35 days (up to 397 from 362).

- A slight resource allocation change, from 3/1/1 to 3/1/2. throws any thought of an inverted utilization-productivity relationship away, however. With this change, the throughput drops to an all-time low (DT=8, 11% worse than our first scenario), while first release date skyrockets to 507 days (40% longer). The resource utilization rate: 56.95%. This is a reflection of the eliminated ability to conduct information-information processing concurrently.

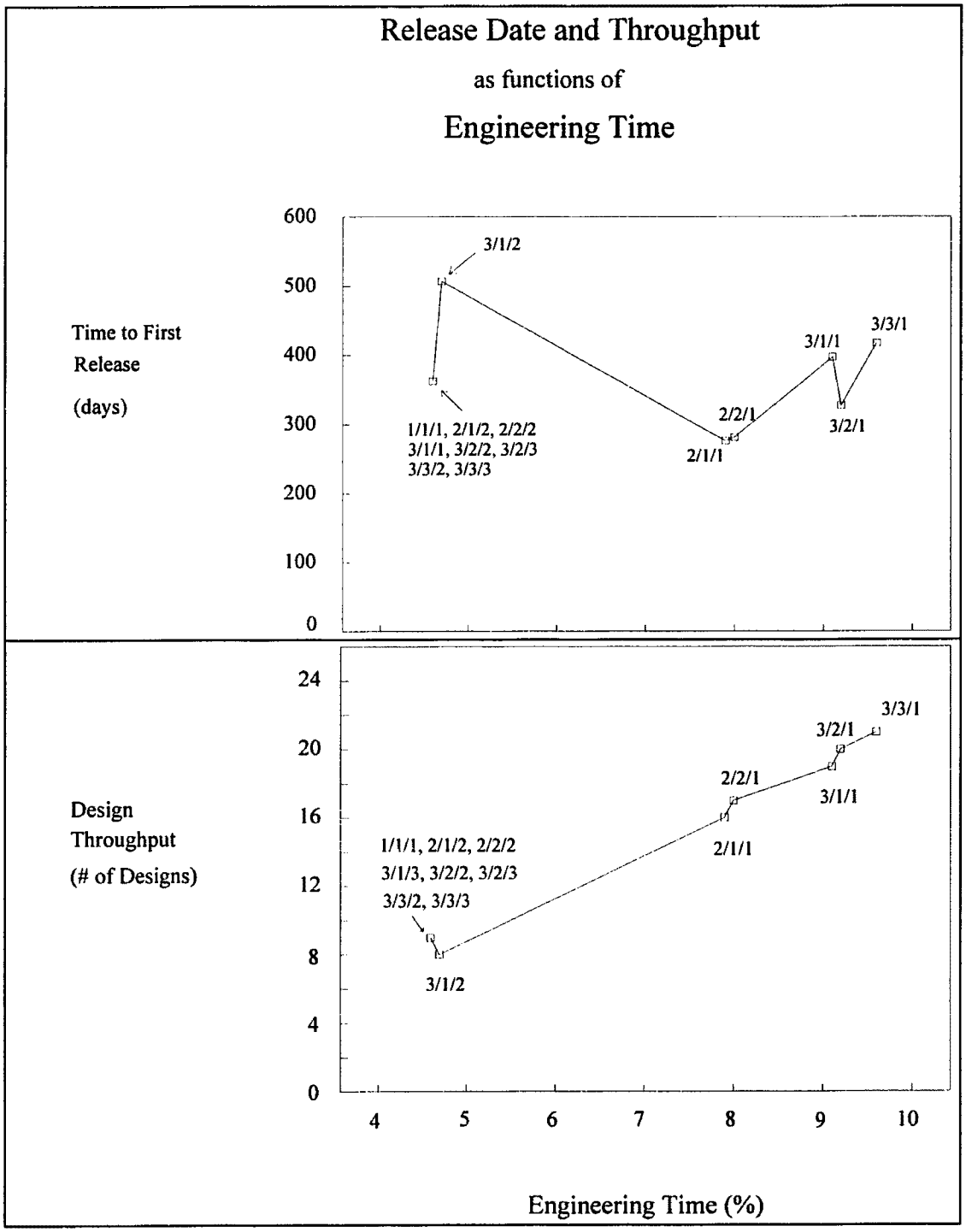
High utilization rates are merely indicative of a system that churns information more efficiently or uses all the people assigned, without regard for the value of their assignments. Our concerns lay with creating designs, not churning information. Thus,

utilization rates do not help us in the same way that some production-oriented process engineers would hope. This is the heart of the efficiency vs. effectiveness argument, perhaps best phrased as ***"Don't confuse activity with accomplishment."***

Since the CPP model of development considers two types of functions, ***prototype processing*** and ***information processing***, we expect to be better served by understanding the *components* of resource utilization, rather than trying to pass judgement from an aggregate utilization number. By segregating information processing from prototype processing, we should get a better feel for ***how*** resources are actually being utilized. One would be inclined to think that resource utilization biased towards MAIN (prototype) processing would provide better aggregate system results.

Refer to Exhibit IV.18., which compares our effectiveness measures, TTFR and DT, to engineering percent for the 14 allocation strategies we tested. All other parameters (MAIN processing efficiency, INFO processing efficiency, Prototype buffer sizes, and information transfer probabilities) were held constant during these fourteen allocation variations. Thus, there are fewer data points than illustrated back in Exhibit IV.10. and V.11. Further, note that there appear to be only seven data points in this illustration, even though fourteen have been plotted. This is a result of the fact that ***various resource allocations can return identical system performance***, as described earlier.

EXHIBIT IV.18.



By isolating the resource allocation-influenced engineering time from other methods of generating increased engineering time, we can observe two interesting effects:

- ***Increases in engineering time, due to resource allocation, appear to increase design throughput.*** A notable exception to this tendency is the 3/1/2 allocation, discussed earlier.

- ***Increases in engineering time, due to resource allocation, appear to have no immediately understandable effect on TTFR.***

Implicit in this result is a more fundamental discovery, which drives our analysis of other parameters:

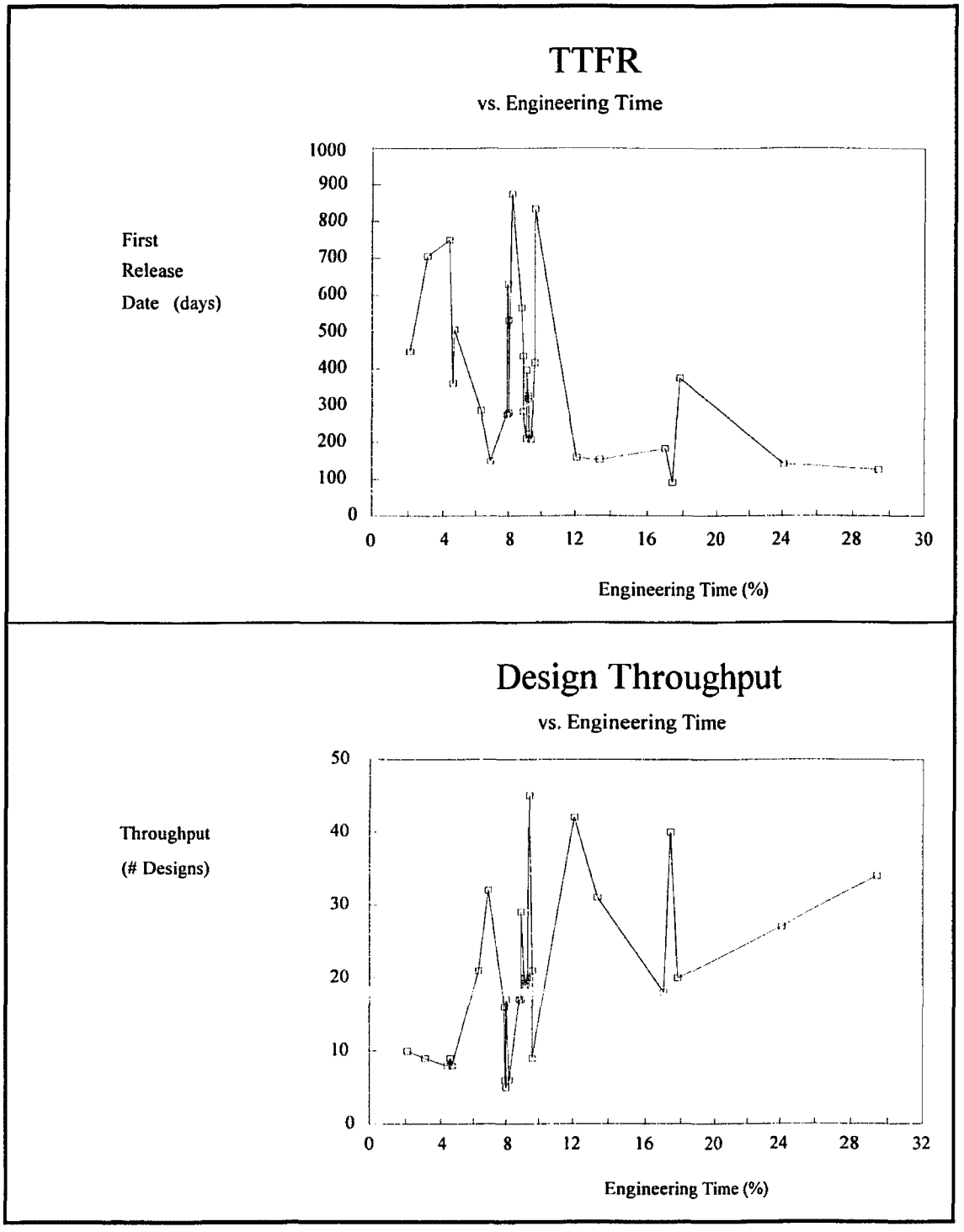
- With this CPP structure, there exist few ***absolute*** guidelines for predicting performance. Rather, ***parameters are dynamically interactive*** with each other, providing numerous ***conditional*** guidelines for design management.

This leaves us with a semi-intuitive, but perhaps controversial result:

- ***How one generates a higher engineering percentage may be more important than raising the engineering percentage in and of itself.***

Consider the differences between the TTFR and DT "maps" in Exhibit IV.15. and Exhibit IV.19. Exhibit IV.19. is similar to Exhibits V.10. and V.11. (from our engineering time discussion), but includes connected lines through the 37 data points.

EXHIBIT IV.19.



Though Exhibit IV.18. inspired confidence that DT increases with increased engineering percent, a more global view (Exhibit IV.19.) shows that throughput is a more complex function of engineering percentage. Examining the TTFR map in Exhibit IV.18., we were left with the implication that engineering time had little predictable impact on first release date. Yet, review of Exhibit IV.19. reveals a general shortening of first release time, particularly as engineering percent rises above 10%.

Though our isolation of resource allocation is intended to *simplify* our insight into the dynamics of this CPP system, we soon discover that there exists *more complex* behavior than originally suspected. It seems that ***the closer we look, the more intricately related our parameters become***. As we discuss in the next chapter, understanding the structure and behavior of these systems may require fractal-like visualizations.

In the next section, we turn to the effects of MAIN processing efficiency on the dynamic CPP Structure. Throughout the rest of our analysis, the 3/3/1 allocation strategy is employed. This strategy is selected for its decent design throughput level (DT = 21) and moderate TTFR value (TTFR = 417). Thus, the baseline resolution level for analyzing other parametric values is higher. Further, the 3/3/1 strategy provides a relatively simple scenario, in which information can be processes concurrently, but not concurrently with prototype processing. This enables a simple "toggling" structure between MAIN and INFO functions, so that engineering time and information processing time can be more easily tracked. In addition, the 3/3/1 strategy permits up to 100% resource utilization for both INFO and MAIN processing.

Throughout this discussion about engineering allocation, we have focused on the performance of the CPP model. Let us consider a few points about the differences between the model and real development systems, a propos the issue of allocation.

- Real development organizations exhibit ***more process complexity and complication*** than our simple model. There exist many more functions and information channels than we exhibit here. Usually, there exist many more departments and human resources, as well. As a result, the allocation effects which we discussed here may occur in local parts of the process, and for only transitory time intervals. This makes it even more difficult to manage from a high-level manager's point of view.

- Typically, there are many ***more functions per engineer*** in real organizations. This implies that engineers engage in even more task switching than demonstrated here. If coordination between engineers is required to accomplish certain functions, however, there exists increased likelihood that ***resource waiting time will be higher.***

- Compared to the real world, ***our research allocation strategies are rather stoic.*** Progressive managers realize the problems of instantaneous mis-allocation of resources and ***adapt in real-time*** to relieve these problems. Over the short-term, this is accomplished through juggling of work schedules, trips, vacation time, and through the use (or non-use) of contract labor. Over longer time frames, permanent employment levels are changed. Thus, managers have some "system correcting" schemes at their disposal.

- Despite goals for maximum resource utilization, there exist conditions whereby ***engineers are very busy, but not on engineering tasks***. In this regard, the CPP model offers good insight into a principal problem of development: non-development functions.

4.3.3.2. MAIN Processing Efficiency

In the field, there is considerable emphasis placed on improving design efficiency. The exploitation of computers by development organizations, in the form of CAD, CASE, FEA, CMM, and many other Computer-Aided Engineering (CAE) tools, has demonstrated a newfound automation orientation. With this orientation, efficiencies have been obtained for numerous isolated development activities. Though the driver for this orientation may be rooted in the philosophy of maximizing resource utilization, and despite the non-intuitive results of the previous section, we still expect that improvements in this area should result in improved system performance.

In this section, we discuss a simple theory for how improvements in MAIN processing are "supposed to" affect system behavior. For this we use an exceptionally simple subset of our CPP model. Next, we turn to observations of the more complete simulated CPP model relative to MAIN processing efficiencies.

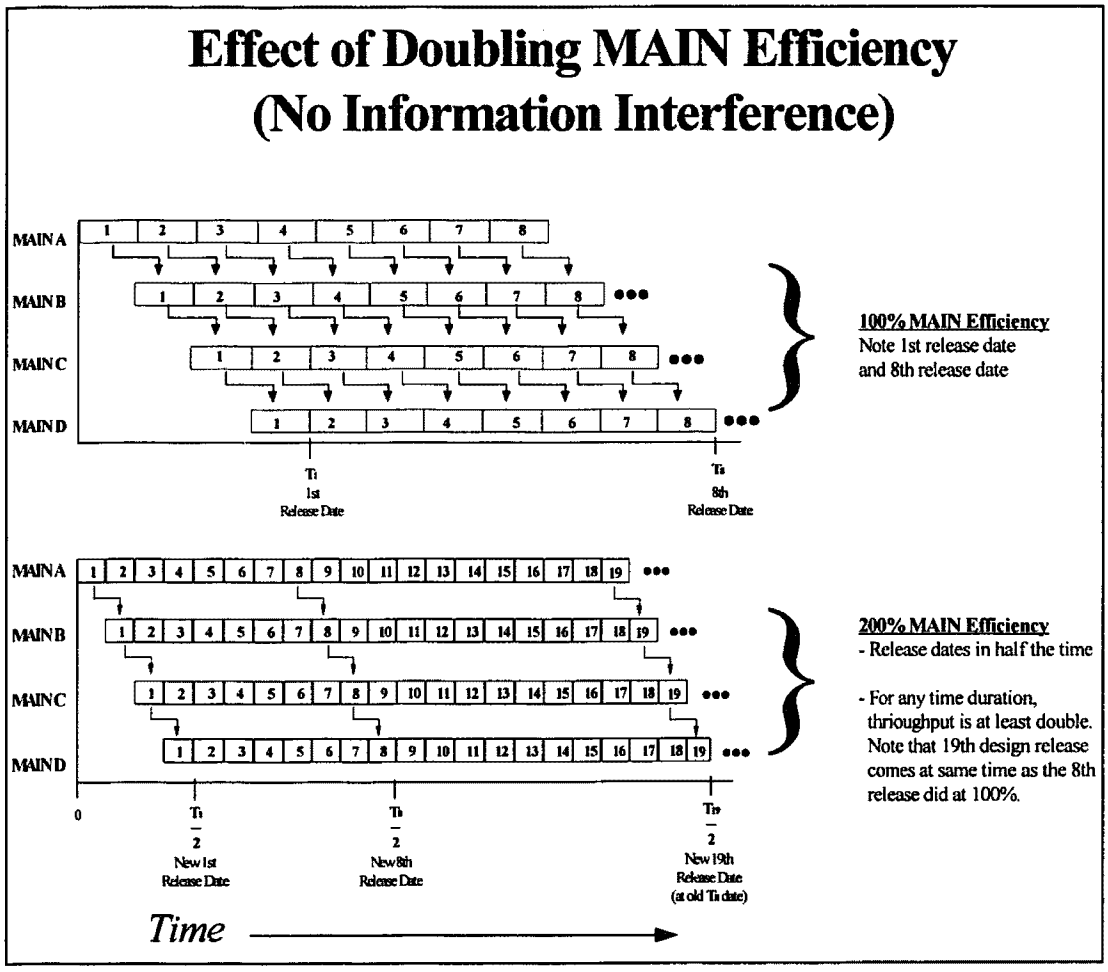
4.3.3.2.1. A linear production paradigm--How the system is "supposed to" behave in response to MAIN Efficiency improvement.

Consider the simple examples presented in Exhibit IV.20. In these Gantt chart-based illustrations, four classes of sequential development activities are delineated. These correspond to the MAIN activities in our CPP structure and are consequently labeled as MAIN A, MAIN B, MAIN C, and MAIN D. It is assumed at this point that there exists

no information interference[75], which would serve to delay the processing of the MAIN functions. Thus, this time series illustration considers the extremely simple case where information-processing is a non-issue. We merely have a sequential set of MAIN activities to perform. As with our CPP model in this study, each MAIN activity possesses equal, non-stochastic process times. Thus, we insure ourselves from interspersed complications of idle time due to downstream "blockage" or "waiting" for upstream activities. One might consider this an optimally "balanced" sequential process.

[75] ***Information Interference*** is formally introduced in our discussion of INFO processing efficiency. It may be described as any information backlogs which block MAIN processing from commencing.

EXHIBIT IV.20.



The numbers within each bar segment correspond to the specific design which is processed by a particular MAIN function at any specific time. Thus, the upper diagram of this exhibit demonstrates the processing of the first eight designs by the MAIN functions at departments A, B, C, and D. A design is not complete until all four departments have been processed. This "process" is demonstrated by the cascading arrows between each of the functions illustrated. By following the first "process" of the upper chart, we see that the first design release date does not occur until time T_1.

While department B is processing the first prototype (MAIN B--1), department A starts the second prototype (MAIN A--2). Thus, the 2nd "process" begins immediately after the first phase of the first process is complete. Such nesting of subsequent developments occurs for the majority of all processing time, with all functions operating continuously when the first prototype reaches department D. The upper chart in this exhibit demonstrates how such nesting occurs for the first eight design releases.

In the upper chart of this exhibit, we consider that MAIN processing efficiency is "normal"--all MAIN functions operate at 100% efficiency. The lower chart demonstrates the identical system, with twice the processing efficiency (MAIN Efficiency = 200%)[76]. Upon initial inspection, this appears to be merely a compressed version of the upper chart. Each of the release dates occur in exactly half the time of those in the upper chart. Further, each of the start dates for each processing function come up twice as fast. One is inclined to think that throughput will double with faster processing time.

For the simple development system illustrated here, however, throughput for any time period is actually ***more than double***. At 200% MAIN processing efficiency, 19 developments are released. Yet, in the same period, the baseline system only releases 8 designs. Thus, our illustration shows a throughput increase of 138%, not 100%. This can be directly attributed to the reduction in idle time ***prior*** to any developmental processing.

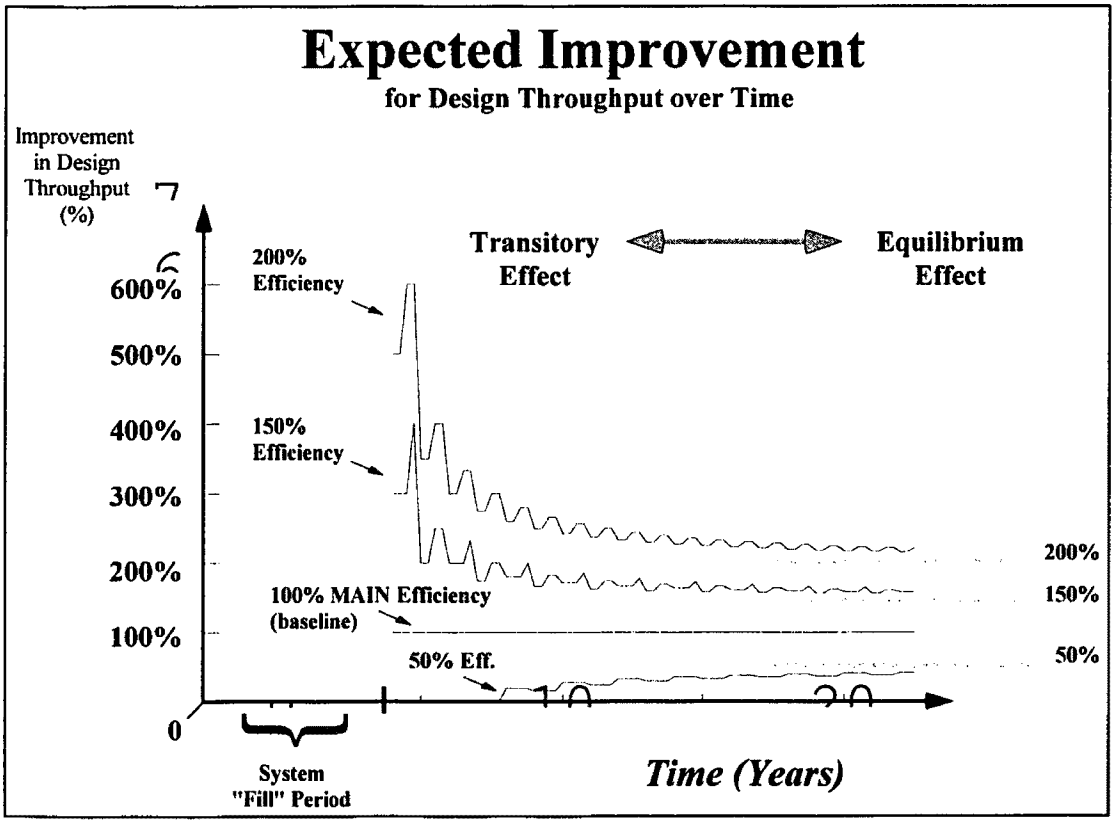
---

[76] Recall that 100% efficiency (of MAIN or INFO processing) is the baseline level for our system. The actual levels of MAINRATE and INFORATE are reflective of the basic standard rates (MAINRATE = 10 days per operation, INFORATE = 1 day per operation) and the respective efficiency level. Thus, 200% MAIN efficiency corresponds to a MAINRATE of 5 days/operation; 150% INFO efficicncy corresponds to an INFORATE of 0.667 days/operation.

by departments B, C, and D. Notice when the first design begins to be processed by department D, under the 200% efficiency scenario. At the same chronological point, under the 100% efficiency scenario, the prototype is only halfway "through" department B. Another way of interpreting this phenomena is to say that the system "warms-up" or conducts "pipeline filling" faster with higher MAIN processing efficiencies.

As time increases, we expect this exceptional improvement to become less and less pronounced. As illustrated in Exhibit IV.21., the gains in throughput should asymptotically approach the originally expected values. For 200% efficiency, we see that throughput approaches twice the baseline as more designs have been released. Note, however, that the average throughput is always higher than the improvement in efficiency. The throughput improvement profile is also demonstrated for 150% efficiency and 50% efficiency. Note the inverted effect of the less efficient (50% efficiency) processing: the initial design throughput rate is even ***lower*** than the expected halving.

EXHIBIT IV.21.



This transitory throughput expectation is reflective of a major difference between the analysis of production processes and the analysis of development processes, regardless of the sophistication of the model being used: the incorporation of ***system transition***. For many high-throughput production systems, it is regular practice to analyze equilibrium system effects. This implies ignoring warm-up time, or "fill" time, for the system under study. ***For the analysis of many stable production systems, it is easier and acceptable to ignore the transitory effects, as they are eclipsed by long-term system behavior.***

Systems of product development, however, are not well suited to such equilibrium analysis. This is because product development functions resemble a collection of job-

shops activities--not a structured, repeatable, high-volume production line. Further, since there tend to be fewer design releases from a development organization than from a well-orchestrated production line, and since subsequent designs are each different from their predecessors, there exist more opportunities to make changes to the structure and efficiencies of the development system between any two releases. With such changes, one can consider the development system to almost always be in a start-up or transitory mode. ***Thus, in the analysis of product development, we must focus on transitory conditions.***

In summary, many production analyses can be acceptably performed by focusing on the asymptotes of Exhibit IV.21. For our dynamic analysis of development, and particularly for *new* product development, we must focus on the transitory, left-side of this illustration. As we find throughout this analysis, such a transitory system can produce non-intuitive, yet field observable results[77].

Nonetheless, we expect TTFR to improve proportionally with improvements in MAIN processing efficiency. As demonstrated with the simple case here, improvements in all subsequent design release dates should also be proportional to MAIN processing efficiency. Design throughput, however, is actually expected to change in a exaggerated manner, in the same direction as MAIN efficiency changes. This is expected as a result of the transitory system of development which we analyze in our CPP model.

---

[77] It is interesting to consider how many convenient scapegoats (and heros) have been created in industry because of ignorance in this area. An alternative, although perhaps equally frequent, problem with managerial avoidance of reality is illustrated by Deming's "Red Bead Experiment." Reference W.E. Deming, "Out of the Crisis," Cambridge (MA):MIT Press, 1986.

4.3.3.2.2. Simulation Results of MAIN Processing Efficiency Variation

The expectations just described appear to be straightforward: increases in MAIN processing efficiency should improve our TTFR and design throughput results. These expectations are based upon a simple sequential set of MAIN activities, which are free from interruptions from information processing. Such ***information interference*** is expected to lengthen the time to first release and to decrease throughput.

If we consider that interfering information processing activities exist to a significant level, and our field observations do indicate this (dedication of nearly 85% of total activity time to INFO-type processing was typical for one site), then some ***dilution of MAIN efficiency impacts*** can be expected. For instance, if each of the prototype processing functions can be improved by, say 50%, and such functions currently occupy 15% of development time, then we could expect total development time to be reduced by (0.15) x (0.5) = 0.075 = only 7.5%.

Naturally, even better MAIN efficiencies and/or increased proportion of these activities to total time "should" provide even better system-wide results. The extreme cases of ***all prototype processing*** and ***all information processing*** are as follows:

- ***No INFO Processing:*** MAIN functions are responsible for 100% of all development time. 50% improvement in MAIN processing efficiency should provide (1.00)(0.50) = 50% improvement in development time.

- ***All INFO processing:*** MAIN functions are never enabled, and thus are responsible for 0% of development time. Since MAIN functions never get a chance to be performed, there is no development. 50% improvement (or any improvement, for that matter) in MAIN processing capability would offer no systematic improvement: (0.00)(0.50) = 0.

Thus, the degree of information processing has a tempering effect on theoretical system performance. For this reason, we present the effects of MAIN processing for a few controlled levels of information processing efficiency. 16 runs were conducted in comparing MAIN efficiency effects: MAIN efficiencies were varied between 50%, 100%, 150%, and 200%. For each of these efficiency levels, INFO efficiency was varied between 50%, 100%, 150%, and 200%, as well. For all 16 of these runs, all other parameters were held constant. Results of these runs are presented in Table IV.4.

TABLE IV.4.

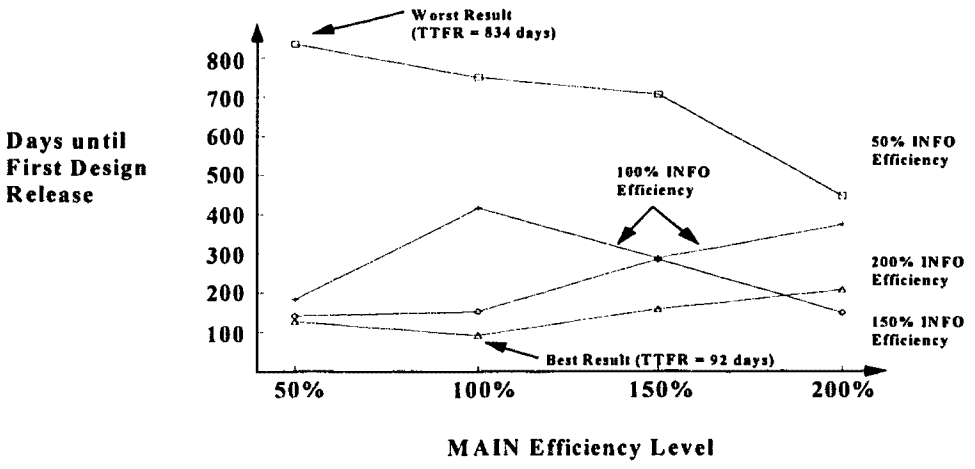
CPP System Performance for Various Efficiency Levels

		MAIN Efficiency			
		50%	**100%**	**150%**	**200%**
INFO Efficiency	**50%**	TTFR = *834* DT = 9	750 *8*	706 9	447 10
	100%	183 18	417 21	288 21	375 20
	150%	141 27	153 31	285 29	150 32
	200%	127 34	**92** 40	159 42	209 days **45** designs

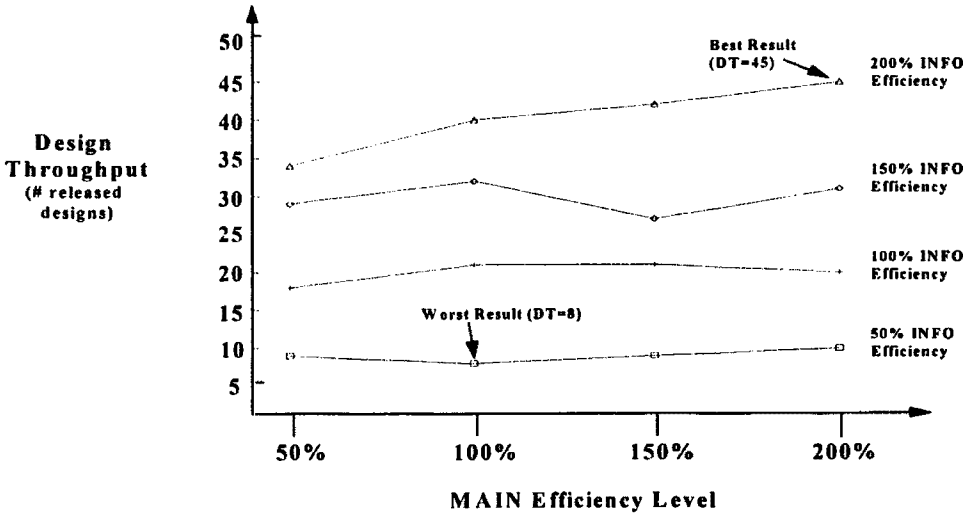
Best Values in **Bold**
Worst Values *Underlined*

EXHIBIT IV.22.

TTFR
vs.
MAIN Efficiency



Design Throughput
vs.
MAIN Efficiency



Graphically, the effects of MAIN efficiency variation are depicted in Exhibit IV.22. (see previous page). Though some characteristics of the results match our expectations (***best overall throughput*** occurs at maximum MAIN efficiency and ***worst overall release date*** occurs at minimum MAIN efficiency), there are several surprises. Note the strange shapes of these curves. Consider the following characteristics:

- ***Global Saddle Point Coordinates***: The most immediately evident surprise is the coordinates of the ***worst overall throughput*** and ***best overall release dates***: both of these occur when MAIN efficiency is at the 100% level. In the former case, INFO efficiency is lowest (50%). In the latter case, information efficiency is highest (200%). For either case, however, these coordinates are ***global saddle points***. Increasing or decreasing MAIN efficiency (by moving right or left in the table or on appropriate isometric INFO efficiency lines) always increases throughput (for the 50% INFO efficiency line) and always worsens release date (for the 200% INFO efficiency line). For a development manager responsible for improving design throughput of a system currently operating at 100% MAIN efficiency and 50% INFO efficiency, it is apparent that changes (improvement or deterioration) in MAIN efficiency may universally ***help*** system performance. For a manager of a (100% MAIN, 200% INFO) system trying to minimize TTFR, deviation from the current MAIN efficiency will serve to ***hurt*** system performance.

- ***Local Saddle Point Coordinates***: Several other ***local saddle points*** exist as well. These points demonstrate a variety of non-uniform convexities of the isometric lines in Exhibit IV.19. From the perspective of a manager focused on design throughput, the following local (and global) minima and maxima were observed:

Table IV.5.

Throughput Saddle Points

	(MAIN Eff, INFO Eff)	Type
(minima)	(100%, 50%)	Local & Global
	(150%, 150%)	Local
(maxima)	(100%, 150%)	Local
	(100%, 100%)	Local
	(150%, 100%)	Local
	(200%, 200%)	Global

If one is inclined to consider first release date as the system performance measure of choice, the following local saddle points were observed:

Table IV.6.

TTFR Saddle Points

	(MAIN Eff, INFO Eff)	Type
(minima)	(100%, 200%)	Local & Global
	(150%, 100%)	Local
(maxima)	(50%, 50%)	Global
	(100%, 100%)	Local
	(150%, 150%)	Local

- ***Competing Effectiveness Measures***: Under various processing efficiency conditions, it is apparent that our system-wide performance measures (TTFR and DT) are not complimentary: ***"optimization" of one system effectiveness measure does not imply optimization of the other.*** This is readily observed by reviewing the location of best system performance using these two measures. For throughput maximization, best performance occurs when both MAIN and INFO efficiencies are 200% (our maximum efficiency test values). At this point, throughput over the 2500 day duration was 45 designs. At this efficiency "coordinate," the first release date was a "better than average" 209 days.

Better first release results are obtained by throttling the MAIN efficiency level back down to 100%, however. At the (100%, 200%) coordinate, TTFR is only 92 days. With this first release improvement, we observe a deterioration of design throughput over the run duration: throughput drops from 45 to 40 designs.

Consider the impact of this feature on real development organizations: working hard at getting the first design out the door sooner can be a prescription for less effective system design throughput over the observation period. This feature has been acknowledged by many experienced development engineers. ***Per their experience, and shown in this model, improvement of design throughput may require slowing the system down during its early stages.***

- ***Some Curves "work"; Others "don't"***: Further review of the local slopes of each line in Exhibit IV.22. are revealing. For each of the system performance measures, TTFR and DT, there exists a single line which qualitatively conforms with the expectations described earlier. For the collection of TTFR measures at 50% INFO efficiency, we can directly observe that performance improves with increased MAIN efficiency. When INFO efficiency is 200%, we also find that design throughput increases in some proportion to MAIN efficiency. ***The other three iso-efficiency lines in each chart do not conform to our expectations***. We shall address this non-linearity in chapter VI, ***Implications of Findings***.

4.3.3.3. INFO Processing Efficiency

As with MAIN (prototype) processing, many managers are concerned with improving ***information processing***. In actual development organizations, this improvement activity is manifested in improvements to engineering change communication processes, configuration management systems, service/technical manual preparation, material handling/tracking systems, voice and data telecommunications, accounting systems/procedures, budgeting/forecasting systems and many other management information systems (automated or not).

We consider the relative rate of INFO processing to the baseline value as ***INFO processing efficiency.*** Efficiency values of 100% indicate that processing rate is equal to the baseline rate; rates higher than 100% are *improvements* (for instance, 200% efficiency means that processing is twice as fast); values lower than 100% represent *deterioration* (50% efficiency means that processing is half as fast; 0% efficiency means that processing has halted altogether.) This is analogous to our concept of ***MAIN processing efficiency*** discussed in the previous section of this chapter.

In this section, we first discuss a simple single-development system which incorporates an "easy" form of the information processing-MAIN processing relationship. We discuss how improvements to INFO efficiency "should" affect such a system. Next, we turn to the results of the more complete dynamic CPP model, relative to INFO processing efficiencies. In this review of results, we also address some interesting relationships *between* MAIN processing efficiency and INFO processing efficiency.

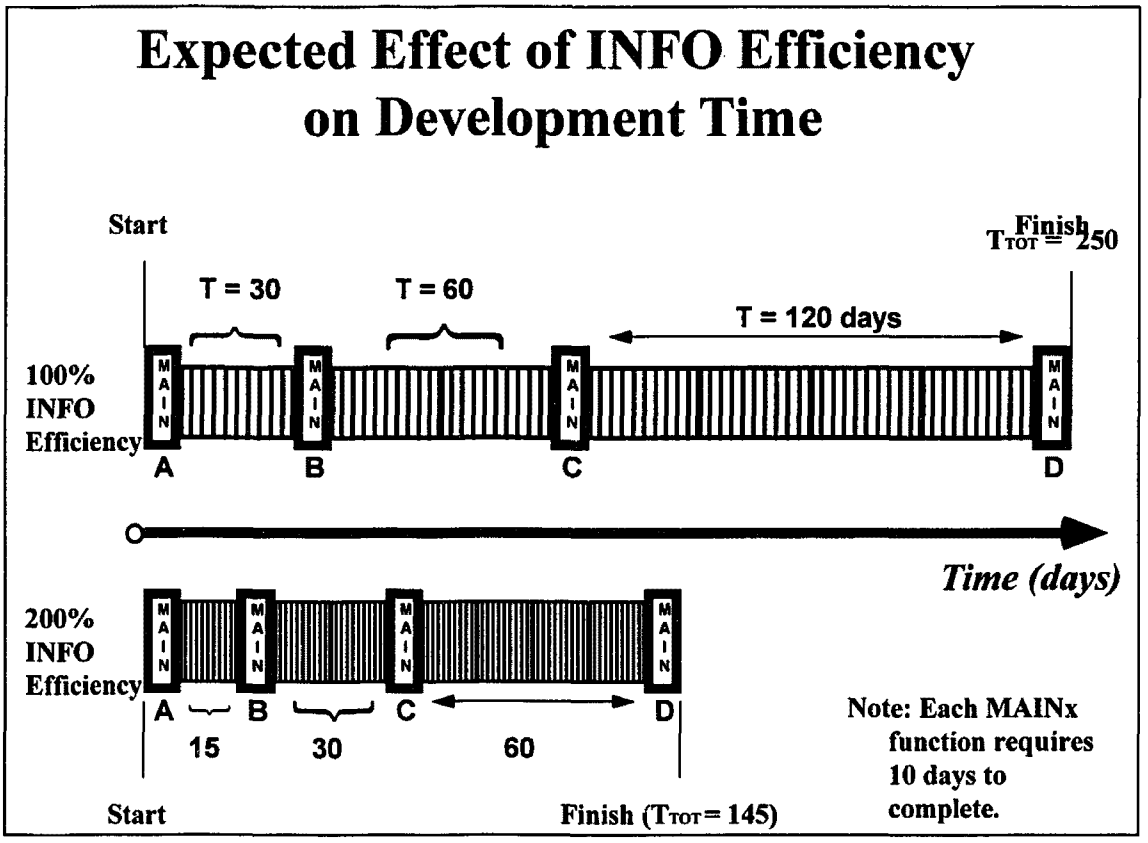
4.3.3.3.1. The Principle of Information Interference

Recall the contrast between our definitions of ***information*** (as a *control* to a function) and ***prototype*** (as value-added *input* to prototype processing). Since our CPP structure is biased towards information dissipation, it follows that increases in INFO processing efficiency will result in faster information dissipation per unit time. This was demonstrated earlier, when we showed increasing information consumption with higher INFO efficiency.

Our objective is not to maximize consumption, however. Our objective is to maximize throughput and/or minimize release times of designs. Thus, an activity which does not directly or indirectly contribute to pushing the design "out the door" may be considered a hindrance in the pursuit of our objectives. We call the collection of process problems arising from such non-contributing activities ***information interference***.

To visualize how variation of INFO efficiency "should" affect our effectiveness measures, consider a simple sequential model in which there exist *no* feedback loops of information or prototype. Further, let us assume that a pre-determined number of information processing activities will be required, regardless of MAIN or INFO processing efficiency. Further, we idealize this case by permitting no slack between functions. Thus, we have a perfectly balanced system with no variances, nor any need for buffers. Refer to Exhibit IV.23.

EXHIBIT IV.23.



First, consider the case of 100% INFO processing efficiency (upper diagram), in which each information processing activity takes one day. Reflective of our baseline CPP model, each prototype processing activity (MAIN, represented by the large blocks) requires 10 days. Note that information processing activities, the accordion-like rectangles are performed "between" MAIN activities. Recalling the priority rules of the CPP structure, downstream MAIN activity cannot be processed until all outstanding department-specific information has been dissipated. In this example, there are 30 INFO activities which need to be performed before MAIN **B** can begin, 60 before MAIN **C**, and

120 before MAIN **D**. Recall that each INFO activity has a baseline processing duration of one day. Thus, the development time is extended by 210 days (30+60+120). Total development time is 250 days, reflective of the four 10-day MAIN functions plus this extension time. Consequently, INFO processing activities are responsible for 84% of total time.

The lower portion of Exhibit IV.23. demonstrates the same system, but with 200% INFO efficiency (MAIN efficiency remains unchanged). Doubling INFO efficiency results in a two-fold reduction of the INFO processing component of development time. Thus, the information to be processed before each MAIN function only takes 15, 30, and 60 days to perform (105 days instead of the previous 210 days). Because of this reduced INFO processing time, the total development time is 145 days, an improvement of 42%. Still, INFO processing accounts for a large portion (72.4%) of this time. Even after doubling information processing capability, the INFO activities extend a 2 month operation (40 MAIN processing days) into seven months. It should be no wonder that we characterize the existence of information processing as ***information interference***: Processing large amounts of information can disrupt developers from their MAIN development tasks, adding significantly to development time.

For a mathematically based estimation of DT and TTFR as a response to improved information efficiencies, refer to Appendix L.

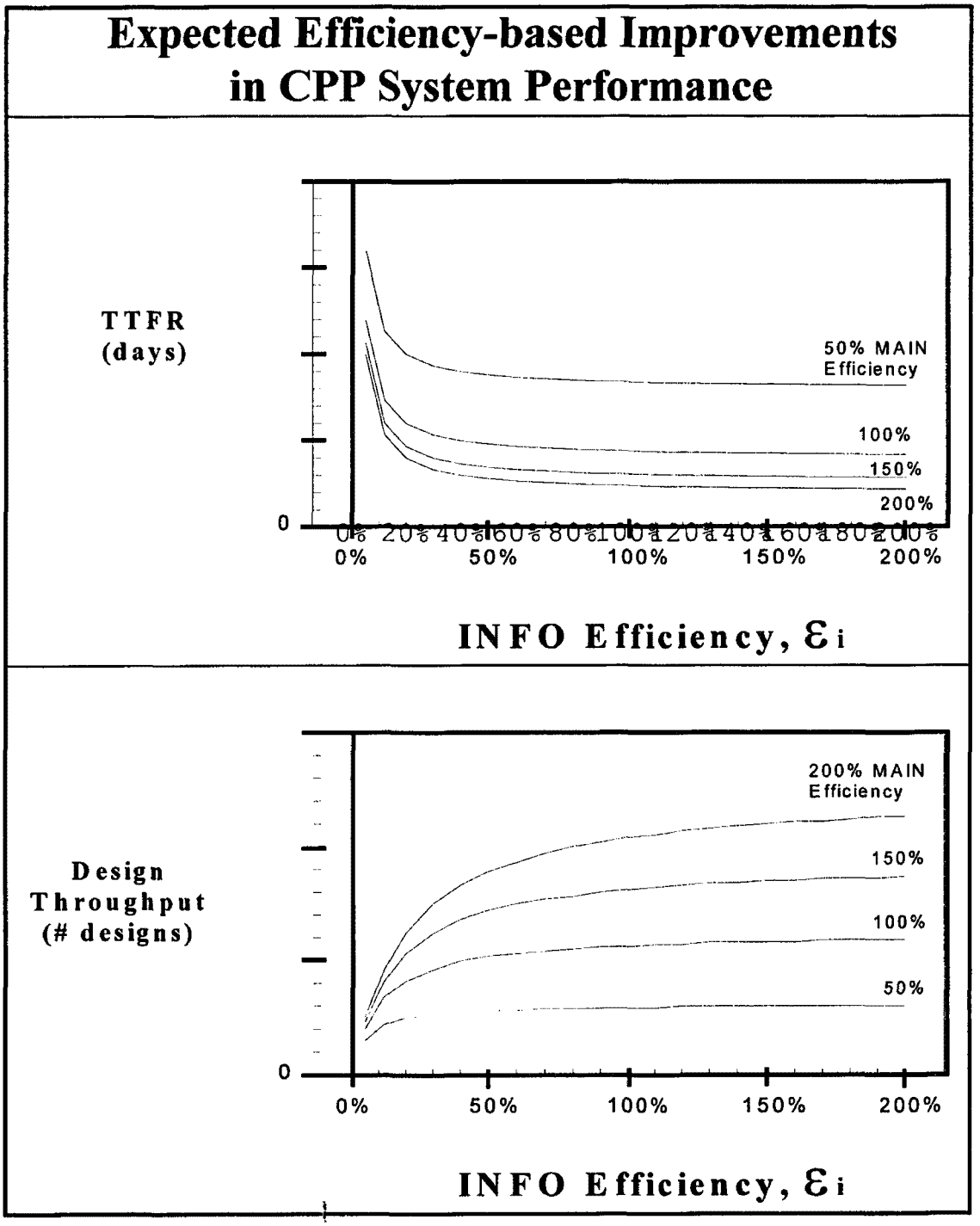
4.3.3.3.2. Simulation Results of INFO Processing Efficiency Variation

The expectations described in the previous section were developed for a simple sequential model, which was illustrated in Exhibit IV.23. Based upon the mathematical framework

of this simple model, improvement in both information and prototype processing efficiencies are expected to result in direct, albeit diminishing, improvements in system performance. The isoquants presented in Exhibit IV.24. illustrate the nature of these expected improvements. For information on the derivation of these curves, refer to Appendix L.

EXHIBIT IV.24.

Expected Efficiency-based Improvements in CPP System Performance



We now return to the CPP Structure, Recall that the CPP models analyzed in this study incorporate iterative information flow. We should thus gain some useful knowledge about the effects of information interference in a system with information feedback. The differences, if any, between the results of this dynamic CPP analysis and the simple sequential analysis of the previous section can give us some indication of the power of information feedback in the CPP model.

Results of the INFO processing efficiency runs (Run ***O*** and Runs ***U*** through ***ZI***) are presented in Table IV.7. This table includes some additional interesting information. Specifically, consider ***OPERATION TIME*** (as contrasted with ***chronological duration***, D, which was held to 2500 days for all CPP runs), as well as its components MAINTIME and INFOTIME. These give a more detailed account of how many engineering man-hours are spent on information processing and prototype processing in the CPP structure, indicators which many managers are well familiar with in their own organizations. Further, we have shown the respective ***functional counts*** for each of these runs, as referenced by COUNT, MAINCOUNT, and INFOCOUNT. Thus, we can gain a sense of the productivity (i.e., how many functions were completed) of system operation, not just consumption of activity time. Although we will not go into detail in describing the significance of these indicators, they were important in analyzing and understanding the complex system dynamics which were observed. Of course, these values become major inputs into a human-resource based cost analysis. There are still many more indicators which can be obtained from the CPP model structure.

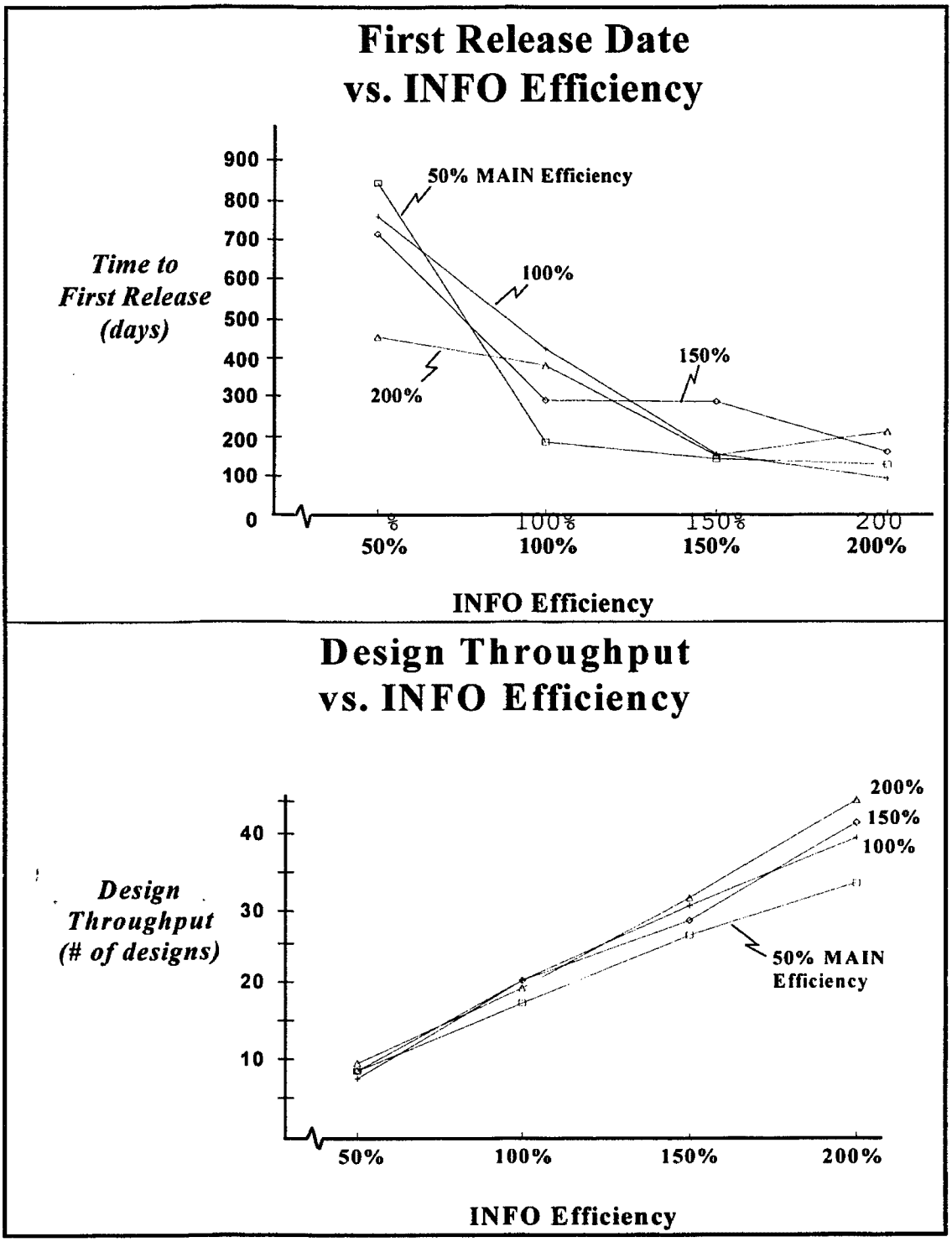
Table IV.7.

PROCESSING EFFICIENCY EFFECTS

RUN ID	INFO EFF (%)	MAIN EFF (%)	DT (# designs)	TTFR (days)	OPN TIME (days)	MAIN TIME (days)	INFO TIME (days)	COUNT (# opns)	MAIN COUNT (# opns)	INFO COUNT (# opns)
Y	50%	50%	9	834	17,774	960	16,814	8,455	48	8,407
X	50%	100%	8	750	16,326	440	15,886	7,987	44	7,943
ZC	50%	150%	9	706	17,195	307	16,888	8,490	46	8,444
ZB	50%	200%	10	447	16,483	225	16,258	8,174	45	8,129
V	100%	50%	18	183	17,226	1,700	15,526	15,611	85	15,526
O	100%	100%	21	417	18,156	960	17,196	17,292	96	17,196
ZD	100%	150%	21	288	17,617	627	16,990	17,084	94	16,990
U	100%	200%	20	375	16,693	445	16,248	16,337	89	16,248
ZI	150%	50%	27	140	16,962	2,400	14,562	21,963	120	21,843
ZH	150%	100%	31	153	17,630	1,340	16,290	24,570	134	24,436
ZF	150%	150%	29	285	17,096	880	16,216	24,457	132	24,325
ZG	150%	200%	32	150	17,452	690	16,762	25,281	138	25,143
ZA	200%	50%	34	126	16,105	2,940	13,165	26,477	147	26,330
W	200%	100%	40	92	17,775	1,740	16,035	32,244	174	32,070
ZE	200%	150%	42	158	17,866	1,213	16,653	33,487	182	33,305
Z	200%	200%	45	209	18,162	940	17,222	34,631	188	34,443

One might ponder how these relate to the expectations of the simple sequential system described earlier. Exhibit IV.25. depicts how variations in INFO efficiency affect our two primary system performance measures, Time to First Release (TTFR) and Design Throughput (DT).

Exhibit IV.25.



Upon initial purview, it appears that the system is behaving according to our expectations. The TTFR curves in Exhibit IV.25. appear to generally decrease with increased INFO processing efficiency. It even appears that the slope of the TTFR curves become more shallow as INFO efficiency increases. Design Throughput also seems to follow our general expectations: DT increases directly with increased INFO processing efficiency. Only the 50% MAIN efficiency and 100% MAIN efficiency iso-curves demonstrate the leveling effect which we described in the previous section, however. In an overall sense, one might reasonably assert that the CPP system is behaving according to our simple linear expectations. Thus, information feedback does not seem to significantly affect the system.

Such a conclusion is incorrect. Although the basic directions and qualitative shapes of the curves more or less follow expectations (their particular slope differentials might be considered to be within "data scatter"), it is surprising to see that the *relative* levels of each curve are so similar. It is also surprising that these curves cross each other in unexpected ways and regimes. Note that 50% MAIN efficiency actually produces the best TTFR performance when INFO efficiency is at the 100% and 150% levels? (We expected that 100%, 150%, and 200% MAIN efficiencies should always produce better results at *any* given INFO efficiency levels.) Perhaps frustrating to managers in actual organizations, and vividly demonstrated here in this simple model, is the observation that ***TTFR and DT each require different efficiency values for best performance***[78]. Such

[78] This is not limited to global saddle points either--comparison of system performance rankings for different efficiency mappings show no mapping that is locally best for both of these system measures simultaneously.

observations indicate that there exists more to CPP system performance than merely additive efficiency changes.

Let us address the first point of quandary raised here. Why do the TTFR and DT curves for different MAIN efficiencies remain so close to one another, even when MAIN efficiencies are so diverse (ranging from 50% to 200%)?

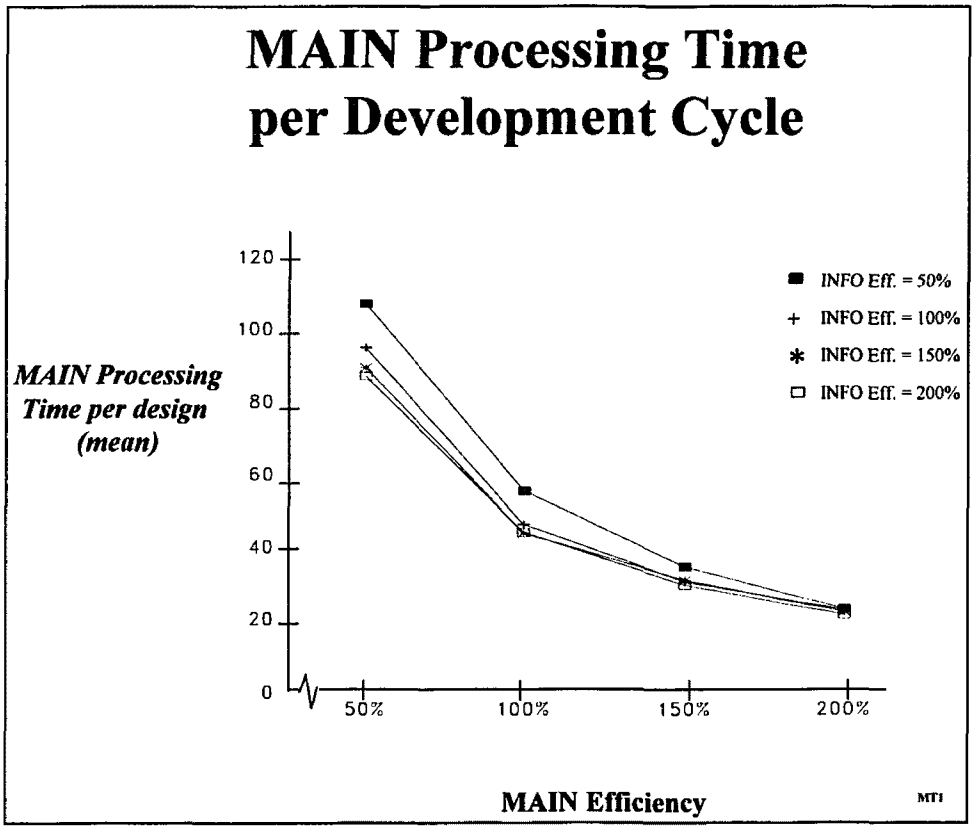
Recall that INFO processing occupies a rather large portion of development time. As indicated earlier, we have documented this to be on the order of 85% of developer's time. One case study ascribes this value to range from 20% to over 70%, depending upon the number of development projects being conducted concurrently[79]. In the CPP models tested, the INFO functions can occupy even higher percentages of a developer's time[80]. Regardless of the specific degree of non-value-added activity at a given point in time, it is widely apparent that INFO processing dominates MAIN processing. As we remarked in our discussion on MAIN processing efficiency variation, even large gains in MAIN processing efficiency will be heavily damped by the presence of INFO processing. Consider the observed improvement of the MAIN ***processing time per development***, as illustrated in Exhibit IV.26. In this illustration, the *average* MAIN processing time per design is contrasted with MAIN efficiency.

---

[79] See Wheelwright & Clark, 1992 pp. 90-91.

[80] Recall, however, that the model assigns any non-value-added task as INFO processing, even if such tasks have little or nothing to do with the development organization. This partially explains our higher value. It is also likely that our field observations located a "worst-case" organization, for the field site at which this value was identified was notorious for inefficient engineering processes.

Exhibit IV.26.



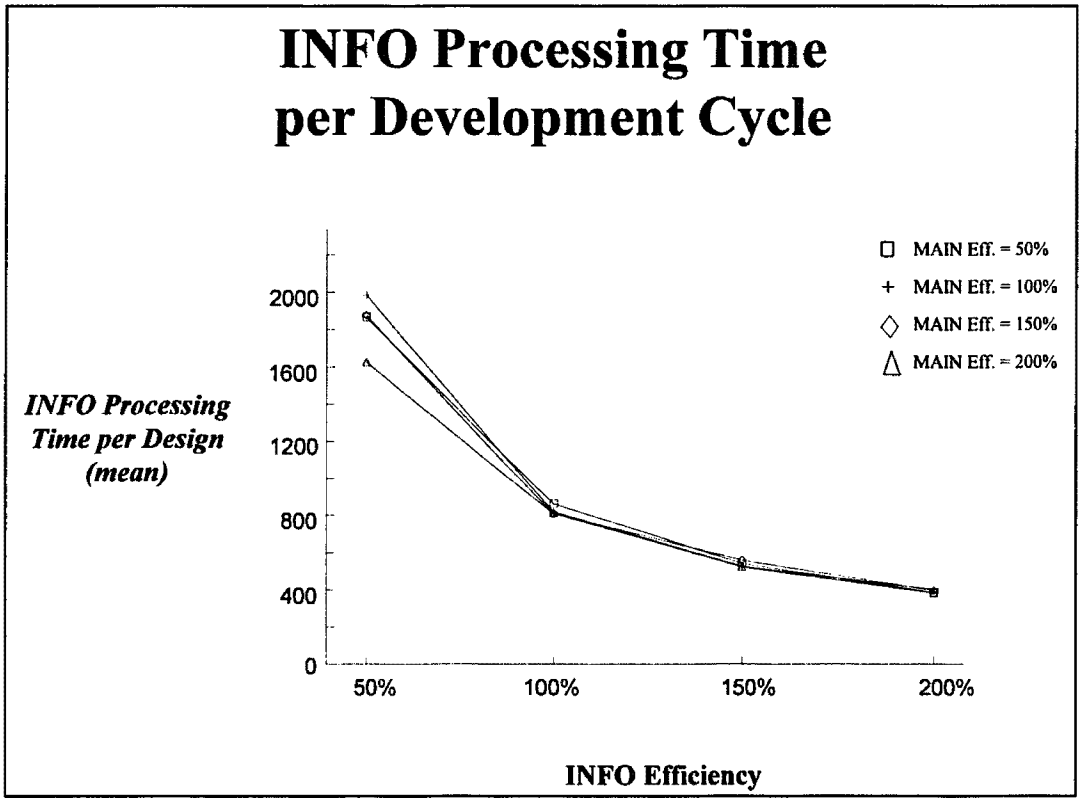
In addition to this factor, however, one must consider the degree of information generation which occurs upon the completion of each MAIN function. Increases in MAIN efficiency, by the rules of the CPP structure, result in increased information generation per unit of MAIN processing time. This serves to "load up" the INFO processing buffers, creating a self-limiting system as more information dissipation is required. This takes more time. ***Thus, increases in MAIN efficiency can increase the time needed between MAIN operations.*** Although one gains time in the processing arena, many such gains are lost through increased waiting times (as demonstrated by more information processing)--the notion of "hurry-up and wait" prevails even in this system!

How does INFO processing efficiency affect this self-limiting system? There are two focal points to consider in this analysis:

1- How INFO efficiency affects observed INFO time per design;
2- How INFO efficiency affects observed MAIN time per design.

With regard to the point, INFO processing time per design, increases in efficiency level produce very expected results: reduction in information processing time is in direct proportion to the increased efficiency level. This is illustrated by Exhibit IV.27. Also note the convergence of these iso-curves as INFO efficiency increases (such convergence is apparent using both absolute and relative measures). This can be attributed to the fact that increased throughput over the constant duration (2500 days) reduces the proportional amount of INFO WIP left in the system at the end date. Note that this characteristic was also true for MAIN efficiency effects on MAIN time per design (e.g., proportionately less MAIN WIP left in the system), which was shown in Exhibit IV.26.

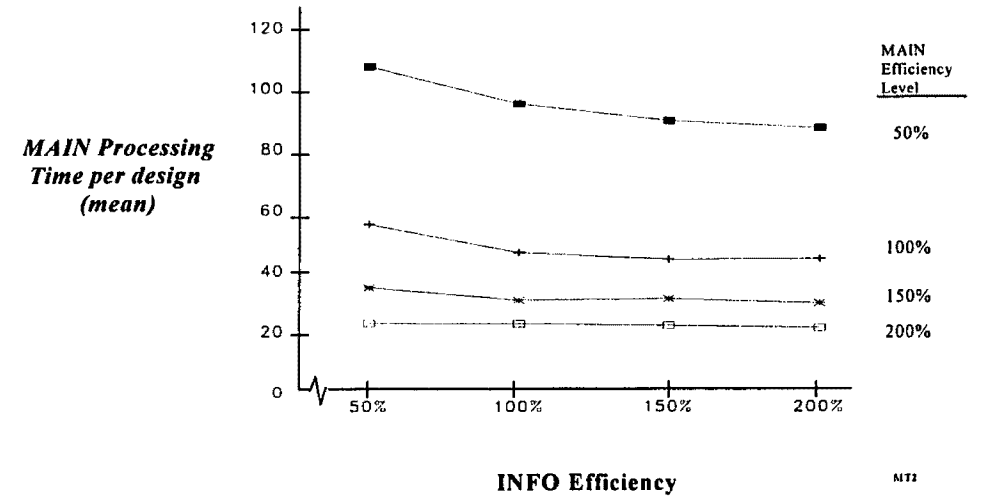
Exhibit IV.27.

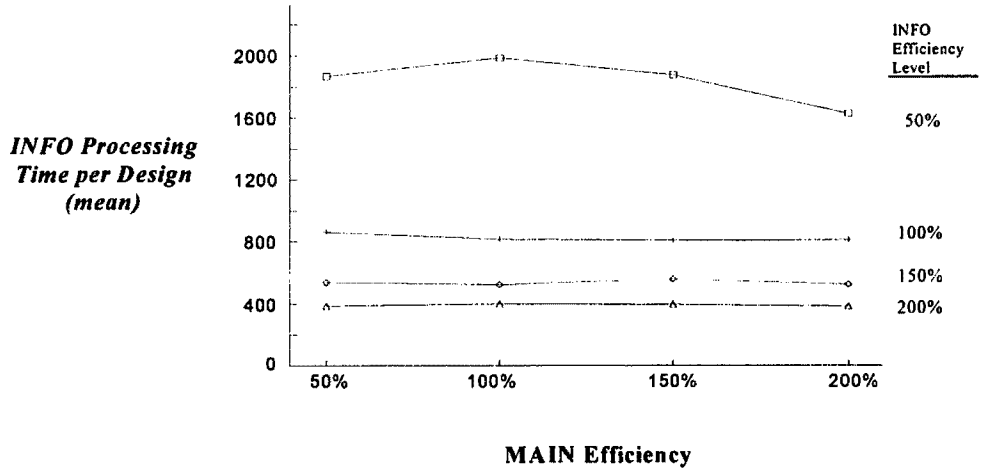


Effective MAIN processing time per design is affected by INFO efficiency in a markedly different manner. In this case (reference the upper graph in Exhibit IV.28.), INFO efficiency has almost *no* effect on effective MAIN processing time, particularly when MAIN efficiency rates are high. When MAIN efficiency is quite low (50%), increases in INFO efficiency do seem to offer a slight improvement in average MAIN processing time per design. This is reflective of the improvement in all of the INFO functions at improving information backlogs, thereby offering more chances for MAIN functions to perform, and increasing throughput--a prerequisite for reducing excess proportional "WIP" at system shutdown.

Exhibit IV.28.

Processing Time per Design as Functions of INFO and MAIN Efficiencies





For reference, the lower portion of Exhibit IV.28 also demonstrates the average INFO processing time per design as a function of MAIN efficiency. Notice that this relationship is very similar to the MAIN time-INFO efficiency relationship just described. The shape of the 50% INFO efficiency curve in this latter diagram is an interesting feature which we shall not delve into here, although it appears to be related to the "nesting coefficient" which we briefly alluded to in Appendix L.

Overall, it is apparent that INFO efficiency can significantly affect system performance. Further, it is apparent that ***INFO efficiency has many more complex, detailed effects on the system which are not easily understood.*** The interaction between INFO efficiency of particular departments and the rest of the system is part of the "ripple-effect" dynamics which make the analysis of CPP structures so interesting. Even more interesting, however, is the observation that similar types of complex dynamics reside in a variety of development organizations, yet are ignored by many managers. Through incorporation of CPP-type models, managers may begin to realize these effects on their organizations. This is the first step towards improving their organizations in new, significant ways.

4.3.3.2. Buffer Size Effects

As we briefly described in the CPP structure description, buffers can have very real associations with entities in actual development organizations. Information buffers, for instance, can effectively reveal the bandwidth of an information processing system. Prototype buffers may also be observed in real development organizations. They serve as stores for completed work from a previous development activity. They are particularly important when multiple products are being developed by the same personnel within an

organization; a situation which inevitably requires prioritization, and integration of such prioritization, among projects.

In the CPP Structure, we have the capability of modifying the sizes of both information buffers and prototype buffers. In this section, we review the effects of changing the sizes of each of these classes of buffers. We begin with a discussion of problematic results which occur with excessively small buffer sizes. Then we turn to a discussion of the impacts of various prototype buffer sizes on the performance of the CPP model. This discussion leads us to some of the interesting dynamics between MAIN and INFO functions within and across departmental borders.

4.3.3.2.1. Information Buffer Size

Information buffers (FileA, FileB, FileC, and FileD) ranging in size from 1 to 100 "units" of information were initially experimented with. If these buffers were held sufficiently small, it was discovered that the system could enter a "paralysis" mode, whereby an INFO processing station becomes dependent on other stations, which themselves could become dependent on yet other stations. When such a dependency chain circled back *to* a "paralyzed" station, a degenerate state could be obtained, as the station became ***dependent on its own completion*** to even continue working! Thus, the system could shut down, producing no output, as *all* activity ceased. Recall that our processing priority rules require INFO processing to be "complete" before prototype (MAIN) processing commences. The dynamics of this paralysis were interesting to witness, for system-wide shutdown did not always occur. It was observed that there existed modes under which stations undergo temporary paralysis, and "recover" before the entire system has the opportunity to "lock-up."

The decision was made to permit the information buffer to be sufficiently large (size=1000) so as not to become a bottleneck to upstream functions. Under this constraint, neither MAIN nor INFO stations could be blocked by limitations of information channel capacity. This permitted full observation of the natural, systematic variation of information transfer. It is expected that future research will investigate interesting variations in the size and balance of information buffers. We discuss some of the ripple dynamics of information transfer, and see how this can be measured using information buffer levels, in Chapter VI, ***Implications of Findings***.

4.3.3.2.2. Prototype Buffer Size

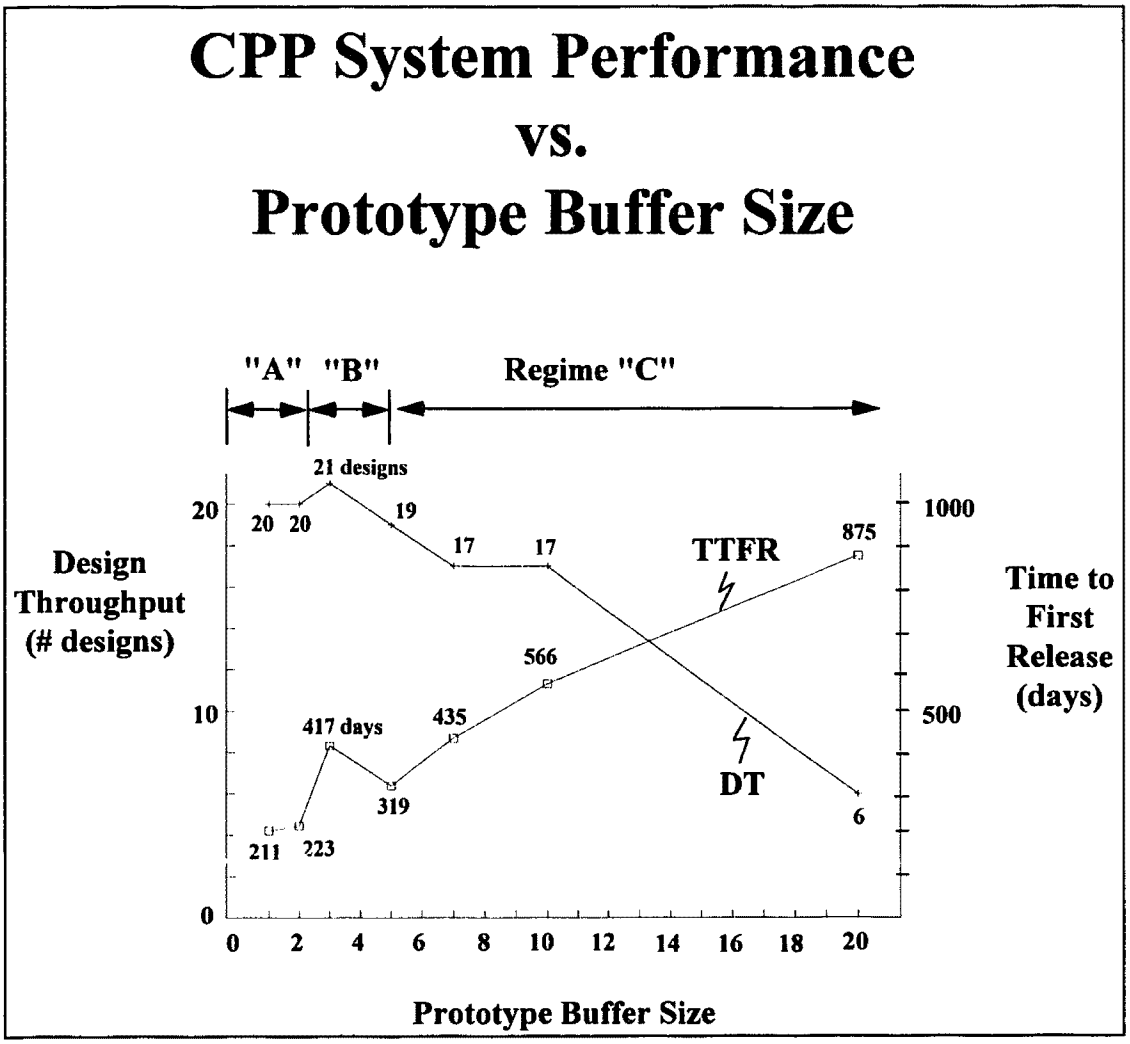
Prototype buffers (ProtoA, ProtoB, and ProtoC) were varied in size from 2 to 20 units, to assess their effect on overall throughput and time until first design release. During this variation, all station processing efficiencies were held at 100%, the information buffers were held at non-binding levels (size= 1000), and transfer probabilities were set to a common baseline. As referenced in the run overview, prototype buffer variation was conducted in runs ***N*** through ***T***.

Given these parameter settings, it was observed that design throughput (DT) and time to first design release (TTFR) are not linear functions of prototype buffer size. Rather, there appear to exist three regimes of buffer size, each offering different results. Exhibit IV.29. demonstrates this system-wide performance variation for a range of buffer sizes.

After investigating the impact of this variation, a baseline buffer size (n=3) was settled upon. This was the size which offered a localized maximum for throughput of MAIN D

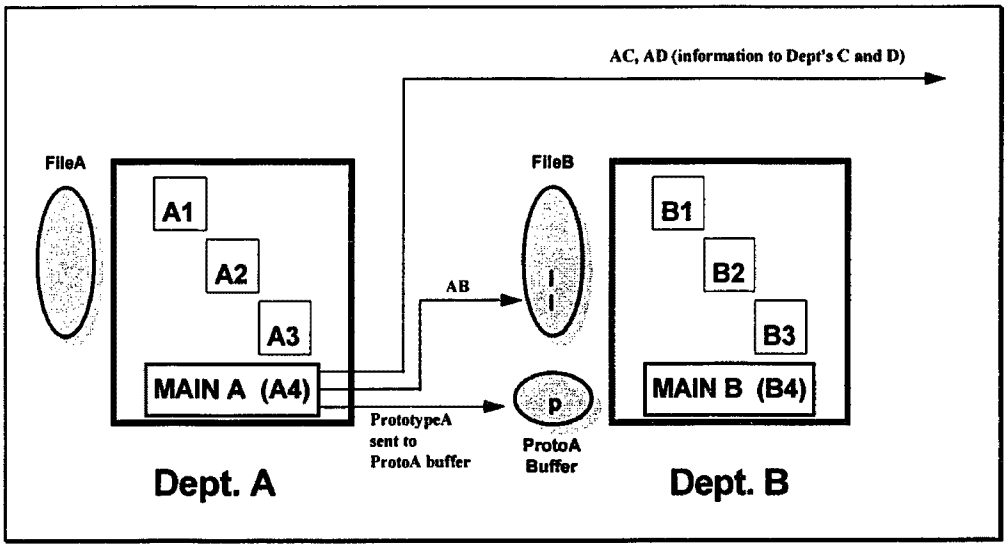
(which equals system-wide throughput), thereby providing better resolution for subsequent parameter variations. At our (100%,100%) efficiency control levels, throughput was 21 designs. First release date was also fairly high at this buffer size, a dwell time of 417 days. All successive simulation runs used this prototype buffer size.

EXHIBIT IV.29.

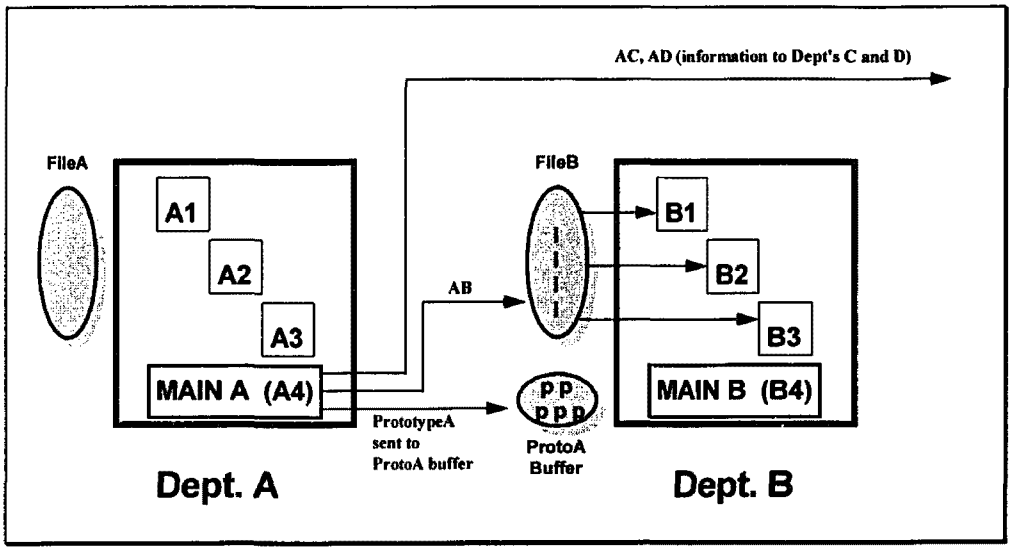


An examination of Exhibit IV.29. offers a worthy observation: ***Increases of prototype buffer sizes generally diminish system performance.*** Aside from the strange behavior in regime B (described more in the next chapter), both TTFR and DT get progressively worse as buffer size increases. Recall that we strive for small TTFR values and large DT values in our assessment of system performance. This finding can be attributed to the nature of the communicative relationship between activities in different departments. As observed in the field, particularly during the study of large-scale configuration management activities, individuals within the system perform their activities ***without regard for the work loads of other individuals*** in the system. This results in parallel, but un-synchronized, activities which actually handicap each other from operating in an effective manner. Direct, frame-by-frame observation of the dynamic CPP model reveals how this can occur:

Frame #1: MAIN A completes processing of a prototype.
As the MAIN function in Dept. A (MAIN A) completes prototype processing, the "finished" prototype (prototype A) is sent to the ProtoA buffer. Simultaneously, the MAIN A function sends information to Depts. B, C, and D (such information is identified as AB, AC, and AD):



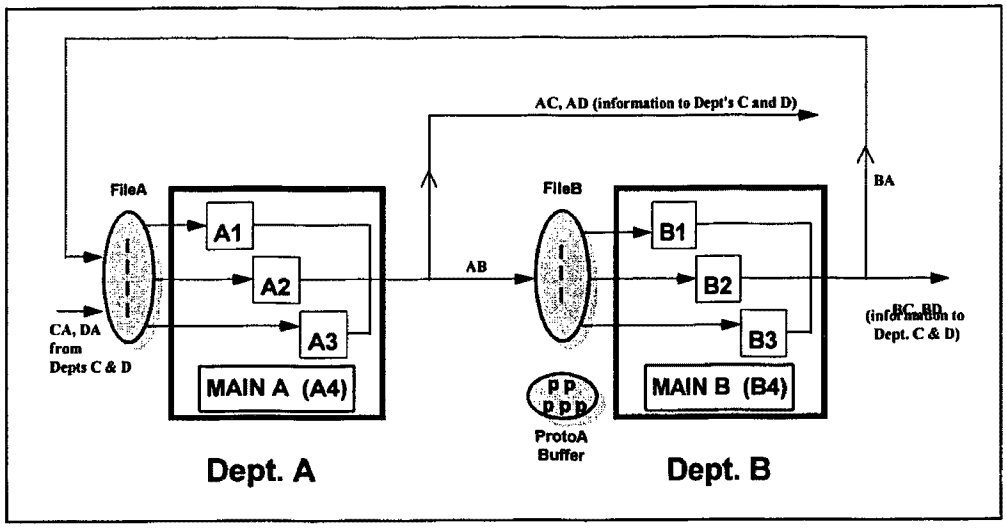
Frame #2: FileB not empty, ProtoA not full--Dept. B falling behind.
As long as Dept. B has information to process, it will not "pull" the prototype from the ProtoA buffer. Provided this buffer is not full, this condition has no adverse effect on the MAIN A activity. In fact, by the rules of this system, MAIN A will provide another prototype A whenever it has the opportunity (i.e., when Dept. A has no information feedback (such as CA, DA, or BA) to process):



Frame #3-A: FileA non-empty, FileB shrinking--Dept. B "catching-up". Thus, if Dept. B is deluged with more information to process than Dept. A, then Dept. B "falls behind". When this happens, Dept. A has greater effective prototype "generating" capability than Dept. B can accept. This causes the ProtoA buffer level to rise. Clearly, a mechanism is needed to prevent MAIN A from producing more information and prototype than Dept. B can manage. This can occur under two conditions:

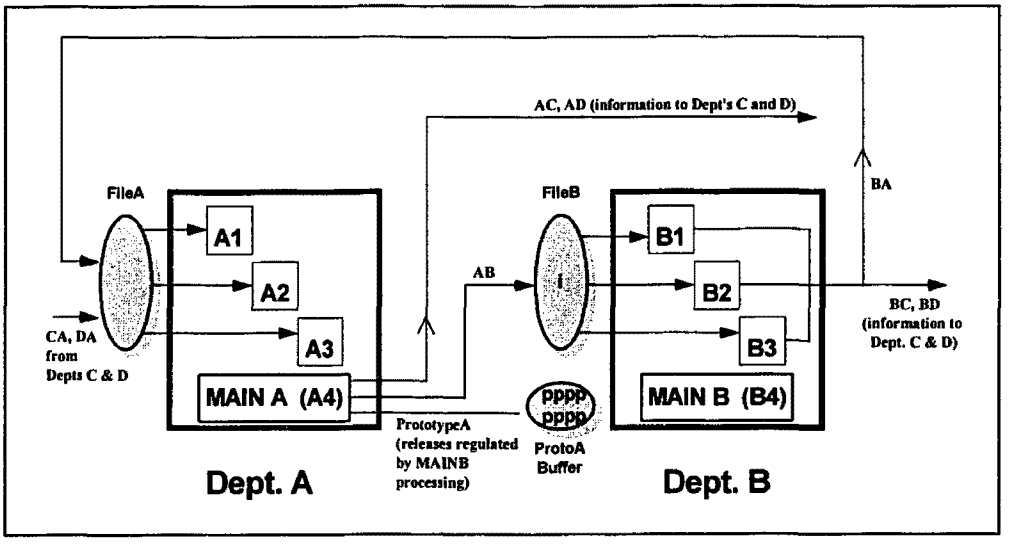
1. Dept. A is forced to process information from any other department, thereby reducing the number of opportunity windows to perform the MAIN A activity;
2. The ProtoA buffer is filled, thereby blocking the MAIN A activity from further processing, regardless of its opportunity to do so.

Under the first condition, Dept. A turns to information processing. Recall that information processing is slightly biased (in this specific model[81]) towards dissipation of information. In time, this gives Dept. B a chance to catch-up on its own information processing:



[81] If this were not the case, we observe a degenerative system, in which information "gluts" the system at some finite time.

Frame #3-B: Dept. A and Dept. B "harmonize" with each other
Under the second condition, Dept. A is prevented from further MAIN processing because the ProtoA buffer is full. This condition gives Dept. B a chance to "equalize" with Dept. A, in terms of both INFO and MAIN processing. This equalization comes at the expense of Dept. A efficiency, rather than efficiency improvement of Dept. B. In effect, MAIN A is regulated by MAIN B:



The prototype buffer size has indirect impact on condition #1 (forced INFO processing) and direct impact on the onset of condition #2 (full prototype buffer). This can be explained by considering the role that prototype buffers play in this system. Just as in a manufacturing process, buffers "cushion" each function from immediately impacting one another. Thus, processing variations in one departmental function do not immediately affect the effective performance of the upstream of downstream functions. This is true as long as the relevant buffer is in use, *but not at its maximum capacity level.* Increases in buffer size can serve to hide (or dampen) larger processing variation among specific functions. This may incline one to consider buffers as favorable entities in the system.

Buffers play another role, however, which can offset their seeming advantage. They serve as repositories for work-in-process (WIP). Thus, an upstream function can produce output after output, unbridled, until the downstream output buffer capacity is reached. In manufacturing, excess WIP (any WIP that is stagnant in a buffer may be considered excess WIP) carries costs along with it. This cost includes the raw material costs that went into the partially completed work, as well as the per-unit processing costs of *all* activities upstream of any WIP item.

There is little contention that selection of appropriate buffer sizes is very much a balancing act. Through analysis of processing rates, processing variances, and understanding of process architecture (e.g., machine sequencing, priorities and material flow paths), appropriate "optimal" buffer sizes can be determined. However, even slight deviations in such established characteristics can render "optimal" sizes inappropriate. Thus, in an unpredictable, continuously changing process, determination of appropriate buffer sizes may be a fruitless, unrewarding exercise. Note that most analyses of buffer sizes make simplifying assumptions about the nature of the system being analyzed, or analyze simple, sequential systems. In fact, one of the most popular manufacturing strategies today, JIT (also known as KANBAN), depends upon a system to have low variances of processing times, and for the system to behave in a predictable manner, usually marked by a linear, though often parallel, process.

In the CPP system, large buffer sizes (more accurately, short-sighted ***behaviors*** which take advantage of the size of the downstream buffer) can result in disastrous system-wide performance. This is because of the dual outputs of each MAIN activity. Upon the release

of prototype from a MAIN processing function, *both* information and prototype are released. The prototype is released to the prototype buffer of the next department, or phase. Simultaneously, information is sent to each of the other departments, in some capacity ("closer" functions get more information). Thus, each MAIN function generates WIP on two counts: ***information WIP*** and ***prototype WIP***.

The prototype buffer size affects the level of both of these WIP types. If the buffer size is low, then the upstream function is regulated by the ability of the downstream function to process the prototype, as the "buffer is full" signal is sent upstream. This restricts the upstream MAIN function from producing more prototype--and more information. As the buffer size is increased, however, the upstream department is given more opportunity to perform its MAIN function, and thus can "deluge" other departments with information, as well as send more prototype to the buffer. ***This gives the upstream function the appearance of being more efficient.*** All downstream functions, however, are negatively impacted by this local efficiency, resulting in system-wide performance deterioration, as demonstrated back in Exhibit IV.29.

Thus, a simple method to help "smooth" upstream efficiency--increasing prototype buffer size--has the net effect of imparting more "problems" (in the form of information processing) on downstream functions. As long as the system is sequential and unidirectional in nature, this problem merely results in systematic slowdown (in an overall performance sense--a few local functions can be extremely busy, never quite able to keep up with their workload), as downstream functions become less effective. If the requirement for the system to be sequential (possessing a singular process path) is

removed, then much of this systematic performance deterioration can be recovered through re-ordering of functions. This is a classical production sequencing problem[82].

Strictly speaking, we have limited the particular CPP models in this analysis[83] to unidirectional, sequential prototype transfer and only let information transfer back upstream or leapfrog ahead of successive functions. This decision was made on the basis of two practical considerations:

- First, most development organizations we visited still operate on a sequential prototype basis. In fact, many development organizations have designed their departments (and their physical plant!) around such sequential prototype transfer;
- Second, the addition of prototype "feedback" in this study complicate analysis so much that the results become extremely difficult to explain[84].

---

[82] One of the most classic solutions to this problem is to order the operations by processing speed--slowest functions first and fastest functions last. This forces balancing of operations and minimizes WIP, but requires that most functions operate far below their capacity. It also results in a net physical ***acceleration*** of the WIP, once the first operation has begun. Contrast this with the practice of many development project managers, who want to "charge out of the starting gate" as fast as possible, only to watch their project decrease its momentum as time elapses.

[83] We have experimented, however, with prototype feedback in test CPP structures.Their results are not presented here.

[84] As a researcher of the subject, I would be the first to admit that the findings described here may seem too complicated to efficiently communicate. Yet, it is ever more readily apparent that actual development "systems" are, in fact, much more complicated than described in this modest report.

4.4. Summary of Dynamic Modeling Results

Let us briefly review our findings from dynamic analysis of the CPP model presented here. We have classified them under the following areas:

- Engineering Resource Allocation
- MAIN Processing Efficiency
- INFO Processing Efficiency
- Buffer Size Effects
- Overall Observations

Engineering Resource Allocation Findings

- ***Concurrent operations*** enable faster ***information dissipation***.
- ***Provisions for concurrency*** do not necessarily result in ***concurrent activity***.
- ***Increases in processing efficiency from resource allocation*** can be tempered by ***increased information processing requirements*** (i.e., internally-driven, workloads).
- ***Inappropriate allocations of additional resources*** can result in ***unchanged or reduced system performance***, as measured by engineering time, first release date, and design throughput.
- ***Resource utilization measures*** are incongruent with New Product Development ***system performance***. In some cases, they are inversely-related metrics.
- ***Increases in engineering time***, via resource allocation methods, generally increase ***design throughput***.
- ***Increases in engineering time***, via resource allocation methods, have uncertain effects on ***first release date***.
- ***How one generates engineering time*** increases may be more important than the metric of ***engineering time*** itself.
- Upon close inspection, ***seemingly unrelated parameters have highly intertwined systematic effects***.

MAIN Processing Efficiency Findings

- We must be prepared to ***focus on transitory system conditions***, not equilibrium systems.
- ***Large increases in MAIN processing efficiency*** return small improvements in ***overall system performance***.
- There exist both ***local and global saddle points of system performance***, when INFO and MAIN processing efficiency are considered together.
- Saddle point locations (i.e., max and min mappings) for ***design throughput (DT)*** do not match saddle points for first ***release date (TTFR)***.
- ***Optimization of TTFR*** does not necessarily result in ***optimization of DT***.
- ***Improved design throughput*** can, under certain conditions, require ***initial system slowdown***, to facilitate workload balancing between departments.
- ***System performance curves*** for different efficiency levels are not necessarily isoquants--they ***may cross one-another***.
- ***Increases in MAIN processing efficiency*** can increase the ***time interval between MAIN functions***, due to the increased amount of information created by more efficient MAIN functions.

Information Processing Efficiency

- ***TTFR*** generally decreases with improvements in ***information processing efficiency***.
- ***DT*** generally increases with improvements in ***information processing efficiency***.
- ***Information feedback*** strongly tempers effects of ***improved information processing efficiency***.
- Per the field-derived priority rules for station processing, ***information processing dominates prototype processing*** in dynamic CPP models.
- ***Increases in information processing efficiency*** can significantly reduce the amount of ***time spent processing information,*** on a per design basis.
- ***Information processing efficiency*** has almost no effect on the amount of ***MAIN processing time spent per design***.
- There exists a myriad of complex, ***higher-order effects*** as a result of changes in information processing efficiency. These may be considered ripple effects.

Buffer Size Findings

- ***Information buffers*** of insufficient size can result in momentary or permanent ***system paralysis.***

- ***DT*** and ***TTFR*** are not linearly responsive to changes in ***prototype buffer size limits.***

- ***Increases in prototype buffer size limits*** generally decrease system performance, as indicated by ***TTFR*** and ***DT.***

- Both ***information and prototype buffers*** serve to cushion the effects of ***functions impacting one another***, provided the buffer is not full.

- ***Once full***, the "pillow" between functions is effectively removed, from the perspective of ***upstream blockage impacts.***

- "Upstream functions" are regulated by the ability of downstream functions to process prototype, and the ability to generate more information. Larger prototype buffers reduce the propensity for such regulation. Thus, prototype buffers affect the level of information in the system. ***Systems with large prototype buffers can be expected to have more information in the system (and more information backlogs).***

Overall Observations

- ***Individuals*** within the CPP system perform their local activities without regard for the workloads of ***other individuals*** in the system.

- Parallel, but ***un-synchronized activities*** handicap the improvements generated by ***local efficiencies***. "Excess improvement" in one location can adversely affect other locations so much that overall system performance deteriorates.

- Under most conditions there exists a ***surging effect***, whereby one department is clearly "gaining ground" on their information backlog, to the detriment of other departments which are losing ground. The "lead" department in this ***information chase*** is not constant, however. It may switch from department to department.

- With appropriate, albeit coincidental, timing between functions, there can exist a ***harmonizing effect***. This effect reduces the propensity for information interference, through better ***dynamic nesting*** of functions.

- There are two types of bottlenecks to consider in these systems: ***information bottlenecks*** and ***prototype bottlenecks***. At any given point in time, ***only one of these is actually limiting the system.***

- There is no single location for bottlenecks in these systems. Rather, the ***"bottleneck" transfers from department to department***, and from station to station within departments, as conditions/situations naturally change.

- ***Transient information bottlenecks***, which are the most prevalent, do not systematically follow the ***prototype path***, but may move in whatever direction has been dictated by accumulated aggregate information transfer. This, in itself, may change over time, as prototype moves through the system, rippled "waves" of information transfer across the system, and as previous information backlogs are whittled away.

In the next chapter, we consider some possible implications of these results on management of new product development. First, however, let us explain a few observed model phenomena and consider some limits of the CPP model with respect to real product development systems.

4.5. Explanation of CPP Behavior

We consider three performance aspects of the CPP structure which are likely to be troublesome to development managers trained with manufacturing-based paradigms. Yet, it is increasingly apparent that these phenomena actually exist in real development organizations today. The aspects we consider here are the following:

- Non-independent effects of "independent" parameters
- Contrasts between islands of efficiency and overall system performance
- Non-linear response from parameters

4.5.1. Independent Parameters are not "Independent"

Earlier, we introduced some basic results of the simulation runs. We addressed how ***engineering allocation, MAIN processing efficiency, INFO processing efficiency, information buffer sizes,*** and ***prototype buffer sizes*** could impact the system. In that discussion, each of these parameters were examined in isolation. Effectively, we controlled conditions so that *direct effects* of each parameter were observable.

Nonetheless, numerous *indirect effects* of parametric changes were also apparent. These are attributed to the inter-relationship among functions within the CPP structure. Such inter-relationships could be synergistic or inhibitory, depending upon the specific parameter values of the system at any point in time.

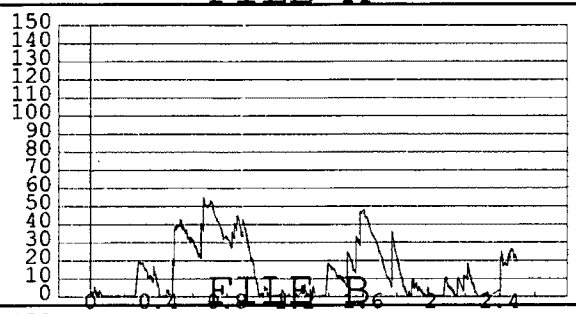
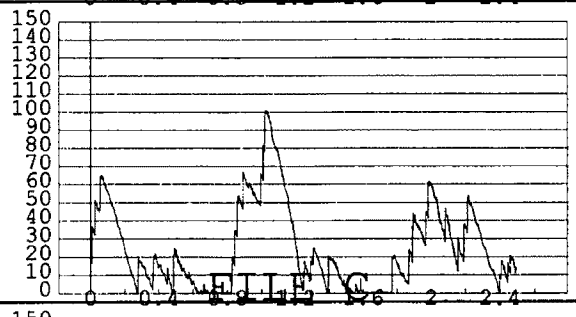
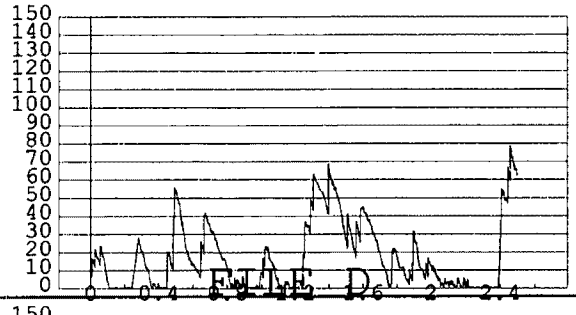
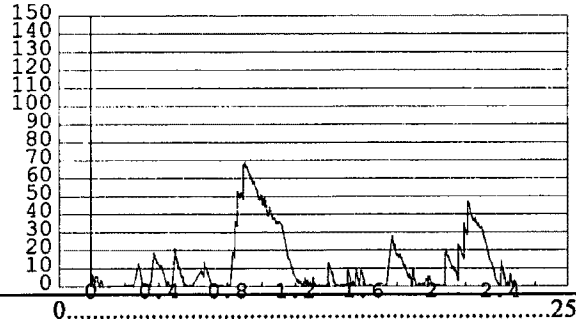
For instance, consider our earlier discussion of prototype buffers. Recall the frame-by-frame observations. For simplicity, those illustrations only considered the relation between Dept. A and Dept. B. In our four department CPP model, the interactions become more complex. This is because Dept. A is instantaneously affected by feedback

information from Depts. B, C, *and* D. As time proceeds, ***each department impacts each of the other departments***. The severity of this impact (measured by the instantaneous information buffer level of each department) can vary significantly over time. Second, third, and higher-order feedback effects are observable, as information is "passed around" the development organization. Depending upon the nature of information transfer at each department (parameters which remained fixed for our analysis), one can begin to appreciate the ***complicated*** and ***complex*** dynamics of information transfer in a true development organization. Exhibit IV.30. demonstrates information backlog profiles over time for each department in one variant of the CPP model.

EXHIBIT IV.30.

Departmental Information Backlogs

(Run ZB)

Department Identification	Backlog Level (# of information units)
150 Department A 0	
150 Department B 0	
150 Department C 0	
150 Department D 0	
Time (x-axis)	0.....2500 days

Notice the generally complimentary nature of buffer levels among these departments over time. This exemplifies a certain degree of MAIN (prototype) activity "toggling" which occurs from department to department. This corresponds very well with the "fits and starts" development atmosphere which one highly successful development manager qualitatively describes. Until now, however, there was little objective explanation of ***why*** this occurs. It is now apparent that ***information processing dominates development schedules considerably more than prototype activities***[85] More importantly, this model has suggested how developers and administrators in one department can affect the performance of other, seemingly unrelated, developers in other departments. This is a ***ripple effect***, not unlike that seen in wave mechanics, for instance. Such effects are expected to occur even more predominantly in large, information-intensive organizations.

4.5.2. Local Efficiencies do not imply system harmony

Recall that the system demonstrated strange performance when prototype buffer sizes were set equal to three. We had referred to this behavioral range as "regime B." Design throughput was slightly higher at n=3, while TTFR was extended. These test values appear to be outlying to the general tendency of smooth system deterioration with increasing prototype buffer size. After checking and re-checking the modeling parameters and structure for errors, it became apparent that this result was, indeed, a systematic behavior. Why was this the case?

---

[85] Yet, anticipated prototype activities still tend to drive posted managerial and engineering schedules.

Based upon visual observation of the simulation, and comparison of the instantaneous information buffer levels for all the simulation runs, we assert the following:

> ***Each CPP structure (including its specific parameter settings) possesses a unique "natural frequency," which enhances and/or restrains system performance. Deviations from this natural frequency normalize the system into "average" behavior.***

Functional coordination can enable such natural frequency. Such coordination takes on two inter-related forms. One is a result of functional processing *time*; the other is a result of functional *timing*. Before we continue, let us define these two concepts:

- ***Functional Processing Time***: The *duration* required for completion of a function, once the function has been initiated.

- ***Functional Timing***: The chronological *point in time* that a function is completed, relative to the start or finish of other functions.

As information is transferred among the organization, various secondary communications are initiated. The degree to which 2nd, 3rd, or higher-order communications exist is a matter of the information transfer parameters at each function. Functions which posses higher information dissipation rates effectively pull more information out of the system. This reduces the amount of information remaining to be further re-directed, and thus reduces the level of "rippled" information. On the other hand, low dissipation rates increase the likelihood that information will stay in the system, offering greater opportunity for higher-order communication dynamics.

Faster processing increases the potential *frequency* with which the function can be performed. As we saw in the previous chapter, such improved processing could either help or hinder system-wide performance. Thus, ***increased frequency in one function may increase or decrease the de facto workload for other functions***. If this increased workload is not within the capability of the "other" functions, backlogs will result.

Similarities between this system and wave dynamics are striking. Initial information generation (which comes from the MAIN functions) is analogous to ***signal generation*** (of a light or sound wave, for instance). When information is being processed by an INFO function, it may be returned (analogous to wave ***reflection***), re-directed (***refraction***), or dissipated (***attenuated*** or ***dampened***). Conceptually, this one-to-one correspondence of information generation and processing is easy to follow. Since we have multiple generation, reflection, refraction, and attenuation devices (4 MAIN functions and 12 INFO functions) and 48 information channels, however, and each device operates pseudo-simultaneously[86], the task of tracking, isolating, and more importantly, ***predicting*** information buffer levels is a non-trivial exercise.

In wave propagation theory, the observed phenomenon of multiple information sources, reflectors, refractors and attenuators calls into the arena the ***superposition principle***. This is the concept that wave amplitudes (in our case, the amount of information) may be added to one another, when such waves overlap. Such amplitudes account for the

---

[86] Recall that the CPP structure used in this analysis has fixed processing rates, and that each operation (except for the MAIN A function) responds to the output of other functions. Therefore, each function is actually "out of phase" with the prior function by an integer multiple of its own processing time. It may be helpful to recall our anticipation of this lag effect in our discussion of MAIN processing efficiency.

inconsistent, instantaneous rises of the information buffer levels at each department. Since our information "waves" are discrete in nature, actually looking more like instantaneous pulses than continuous smooth waves, "nice" continuous functions do not adequately describe our overall wave dynamics[87]. Thus, wave decomposition strategies such as Fourier transformations do not lend themselves well to analysis of this system. Nevertheless, superposition principles appear to hold in this model.

Strange overall waves can arise from such superposition. However, *exceptionally* strange looking waves can be generated when frequencies of component waves are dissimilar. Since information processing and MAIN processing functions possess different processing rates in this CPP structure, information refractions and reflections were expected to occur at frequencies (the reciprocal of the time between outputs of the same function) different than information generation frequencies. In fact, *none* of the 52 model runs in this analysis demonstrated consistent generation frequencies. Nor did information processing functions satisfactorily synchronize with each other or the MAIN functions.

The fact that MAIN functions are regulated by the existence of information at a department (MAIN processing only occurs when the department's information buffer is empty) serves to complicate the dynamics even further. This is because signal generation itself becomes dependent upon the degree of refraction, reflection, and absorption of the

---

[87] As an interesting aside, the velocity of waves are computable, based on the elasticity, F, and the medium density, μ. The relation is $V=(F/\mu)^{1/2}$. Since we have assumed infinite channel capacity in our CPP model, the information velocity is infinite. This means that the elasticity is enormous and/or that the medium density is very close to zero. It appears convenient to consider elasticity as a measure of information channel bandwidth and density to be a measure of channel attenuation level. Thus, our information channels in the CPP systems studied here appear to have infinite bandwidth and/or no attenuation.

system. Moreover, ***each MAIN function is controlled by the operations of its own department's INFO functions, as well as the INFO and MAIN functions at all other departments!***

If such conditions prevail in an actual development system, our model predicts an intuitive result: ***development will not proceed smoothly with a number of "prima donna" engineers.*** Cooperation among developers, to help synchronize their information and prototype processing, would go a long way towards expediting prototype "through" the system.

So how does this all relate to the strange behavior in "regime B" (n=3) of the system? Recall that none of the runs in this analysis demonstrated consistent generation frequencies. This means MAIN functions are performed on an irregular basis[88]. Given that each of the MAIN functions exhibited this behavior, and considering the reverberations of information throughout the system after each MAIN operation, it is evident that the MAIN functions are not synchronized with each other. This was particularly evident when the prototype buffers were not full[89].

Under various conditions, however, the system achieves "better" timing among MAIN and INFO functions, such that more emphasis gets placed upon pushing the design out the door than optimizing individual departments.

---

[88] We can deduce this because only MAIN functions are responsible for creation of new information in the system. Real organizations do not generally have such convenient information creation rules.

[89] When the buffers were full, recall that some " harmonization" (albeit slower performance) among MAIN functions can occur. This was illustrated in Frame 3-B, of the prototype buffer discussion.

It was also apparent that average engineering time, by itself, was *not* a good indicator, nor predictor, of system performance. Rather, ***balance*** among the MAIN functions affects overall performance. The ***ratio of average engineering time to engineering time balance*** (measured by the range of highest to lowest engineering time of each department) seems an excellent predictor of the system's overall performance. Table IV.8. demonstrates some of the best and worst performances of the CPP system along with these three indicators.

Table IV.8.

Best Performing CPP Systems

Run ID	TTFR (days)	Throughput (designs)	Average Engineering Time (%)	Engineering Time Range (High-Low)	ET Average to ET Range (Ratio)	Average Max Buffer Level
Z	209	45	9.4	0.8	11.75	56.00
ZE	159	42	12.1	2.1	5.76	61.00
W	92	40	17.4	3.2	5.44	70.75
ZA	127	34	29.4	5.6	5.25	82.00
ZG	150	32	6.9	1.4	4.92	53.75
ZH	153	31	13.4	2.8	4.79	61.50

Worst Performing CPP Systems

Run ID	TTFR (days)	Throughput (designs)	Average Engineering Time (%)	Engineering Time Range (High-Low)	ET Average to ET Range (Ratio)	Average Max Buffer Level
X	750	8	4.4	3.2	1.37	60.50
Y	834	9	9.6	5.6	1.71	53.25
ZC	706	9	3.1	1.6	1.94	72.25
ZB	447	10	2.1	0.8	2.62	77.00

For the buffer size variation runs, this ratio seems to hold up as well. Refer to Table IV.9., below.

Table IV.9.

CPP Performance as a function of Buffer Size

Run ID	Prototype Buffer Capacity	TTFR (days)	DT (designs)	Average Engineering Time (%)	Engineering Time Range (High-Low)	ET Average to ET Range (Ratio)	Average Max Buffer Level
R	1	211	20	9.1	2.0	4.55	47.25
N	2	223	20	9.3	2.8	3.32	55.75
O	3	417	21	9.6	2.8	3.42	54.75
Q	5	319	19	9.2	4.0	2.30	54.00
S	7	435	17	8.9	4.0	2.22	13.90
P	10	566	17	8.8	6.0	1.47	58.00
T	20	875	6	8.2	13.2	0.62	21.85

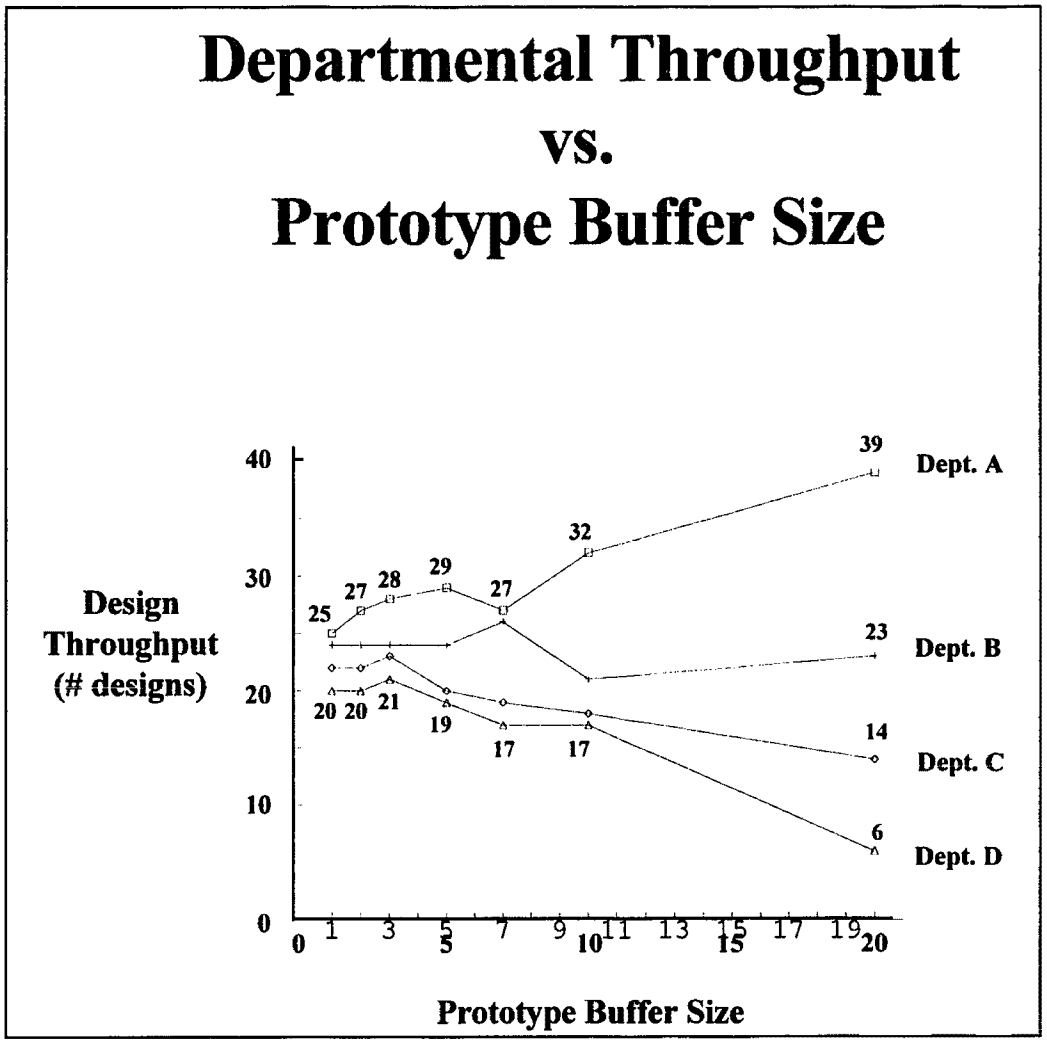
Notice that run "O" demonstrated a higher ratio (3.42) than adjacent runs "N" (3.32) and "Q" (2.30). What these findings indicate is that balance (smallest range) among MAIN function helps keep the system under control, in such a manner that no single department ever gets so far "behind" as to become a drain on the rest of the system. Using wave propagation terms, we can say that the *amplitude* of the overall wave (as seen in the information buffers) does not become excessive. This permits higher frequencies of both prototype and information processing. Contrast this with "poor" systems which exhibit large amplitudes and low frequencies of MAIN operation. This phenomenon is analogous

to the constant work-in-process (CONWIP) scenario used in some advanced manufacturing plants[90].

Run "O" did not show the best system performance in terms of TTFR. This can be attributed to the tradeoff between establishing good functional balance and delivering the first designs in a hurried manner. Exhibit IV.31. demonstrates how variations in buffer size can result in great divergence of departmental efficiency. Note, of course, that MAIN D performance (the "system effectiveness" measure) is the only one which we really care about in this analysis. Nevertheless, the managers of Dept. A and Dept. B seem to be doing a "great job" of maintaining or increasing their efficiency, *despite the deterioration of the rest of the system.* Engineering executives should consider such localized effects during their project review process.

[90] For more information on CONWIP, see Woodruff (1990).

Exhibit IV.31.



4.5.3. MAIN-INFO interaction--Linear predictions don't work

Many developers have suggested that their development organizations perform better if they restrain their early actions, in favor of better requirements definition. In large measure, this means ***slowing the system down early, so that it may ultimately finish much faster (and sooner)***. For certain conditions in the CPP Structure, we have confirmed this behavior, but for altogether different reasons! Functional coordination

seems to require moderate speed building on a system-wide basis, not localized flashes. Our field observations that requirements definition may have diffusion lags serves to reinforce this need for "smooth" system speed ramp-up. Yet, this can be a counter-intuitive principle to many managers, for *any* speed improvement, particularly for functions on the "critical path," is usually considered to be desirable.

Non-linear development processes, however, do not seem to comply with traditional CPM-type approaches[91]. The primary reason for this is that the ***critical path*** (as seen *a priori*) ***often changes its course***. What looked like an inevitable delay ahead, soon looks minor when compared to the new delay which literally pops up. By the time this new delay is rectified, the old delay may no longer be a problem, but then another new one may arise...

The CPP Structure does not carry the sophistication for us to predict the *exact* nature or time of the delay which will "pop up." It does, however, show the *types* of consequences which can occur if such a delay were to become manifest.

As with many linear systems, ***certain processing delays may have no immediate effect*** on the chronological performance of our system. This is true as long as no other functions are critically dependent on completion of such tasks. For linear systems, it may be deduced that only "upstream" functions need to worry about delaying the work of others.

---

[91] *Ex post facto*, *any* process may be considered to have *had* a critical path. By stringing out the functions as they were conducted (and repeated), we could historically recreate the actual paths which were followed, and then identify the critical path. Actually, this was much of the activity which we engaged in during the field studies as a background to the development of the CPP Structure. Fundamentally, however, there is no tool for accurately ***anticipating*** these convoluted paths.

As long as the process is unidirectional, only functions in the direction of the process (i.e., downstream) "feel" the delays[92]. For hundreds, perhaps thousands, of years multi-stage unidirectional systems have had to worry about this concept. Various distribution systems, communication systems, and production systems have all been victims of this problem. Through the incorporation of backup/redundancy systems and slack (buffer) systems, the problem was improved...until *cost* became a growing consideration. Hence, the surge over the past 15 years (which incidentally corresponded to huge costs of capital and competing investment opportunity) towards inventory reduction...and many studies on how to get this reduction without reducing service.

For development systems, however, *delays are a two-way street*. Functions that may have been "downstream" yesterday are now, due to process (or product requirement) changes, upstream functions. And tomorrow, they may be downstream functions again! Given this changing condition, how can a participant know if s/he is on the critical path? The most common solution is to "always act like you are on the critical path, because one day you might be." Given the problems we see with working too fast, however, this puts developers in a dilemma: ***speed up*** (in case you are on the critical path), but ***slow down*** (because your speed may hurt those who are on the critical path).

Currently, we cannot predict the significance of any functional delay on the overall performance of the system. Delays may *negatively* affect the system (the intuitive effect),

---

[92] Naturally, it has always been possible for products of the upstream functions to be stalled by inadequate readiness of the next downstream function. From a customer's perspective, this delay could be considered the downstream function's fault.

or have *no effect* (if, after process changes, it is not on the critical path). Or, it may even *improve* the system! Consider the following:

> During some experimental simulation runs, subsequent to the results presented in the previous sections, we gave the CPP Structure extremely high prototype processing rates. This resulted in such prolific output of prototype and information from Department A that Departments B, C, and D were deluged with large backlogs of work by the end of the first simulation "day"! Moreover, since Department A had such a head start on the other departments, and did not have the same amount of work being sent back to it (recall the bias of information functions toward information *dissipation*), it remained in control for the duration of the control time, essentially by keeping the ProtoA buffer filled to its limits. The resulting imbalance caused an ***overall system slow down*** (i.e., reduce its system-wide output). Thus, vast increases of speed on the system's components did not return commensurate performance increases.
>
> Upon turning the speed even higher, the system could ***cease producing output entirely***. Similar results can also be obtained by increasing prototype buffer sizes. Even increasing the speed of all functions (to fifty times their normal rate) resulted in fantastic initial performance (TTFR= 6 days, DT= 79 designs), followed by system stall after only 176 days. How's that for a developmental flash in the pan?

Thus, if you are currently operating in an "over-efficiency" mode (where have you heard that term before?), ***reduction in speed may actually help the system.***

What does this say about present paradigms for improvement? We believe it implies at least three things:

First, as a practical matter, ***local performance improvements must be considered in context of the overall system*** before they can be deemed favorable improvements. Wouldn't every manager dream of improving his local engineering efficiency by 5000%? With the CPP structure, we can see that this really might not help overall performance.

Second, ***be careful not to make linear, traditional assumptions of effects.*** An across the board increase in station processing rates for this type of system is *not* the same as merely increasing the clock speed or changing time units during the computer simulation. For instance, can you be prepared to say that tenfold efficiency over one tenth the time will produce the same results as half efficiency over twice the time? Or that twice the efficiency over the same time will produce twice the output? It becomes increasingly clear that answer to these questions is "maybe."

Third, when managing any "improvements", one must ***consider what other changes may be necessary*** to accommodate these improvements. For instance, systematic deterioration with increasing prototype processing efficiency was a problem of inadequate information processing to cope with the large "blasts" of information from the super-efficient MAIN functions. In severe situations, even very high information buffer levels can become filled, causing blocking effects. Increasing such buffers even higher, however, is only treating symptoms, not the root problem. Either MAIN functions must reduce their information bundle size, or INFO functions must increase their processing speed.

4.6. Limits of the Model

In the next chapter, we discuss some implications of our research for management of real development projects. Before we do so, let us review some of the constraints of the current model, so that it may be put into better perspective.

Purpose/Use: Per the categorizations forwarded by Richard Cyert[93], our model is a *descriptive* and *illustrative* device, not a *normative* or *man-machine* simulation. Thus, its purposes are limited to demonstrating our theories and field observations of new product development, utilizing structures and parameters which are partial derivative of the real-world. It was not developed with the intent of designing new organizations, nor as an integral training-tool or real-time decision-support system. We do expect, however, that such purposes may be fulfilled in future evolutions and refinements of our model.

Simplicity: The model developed in this study is an extremely simplified model of reality. We do not offer this model as a representative facsimile of any real organization, but only as a visualization of the *types* of structures and resulting behaviors which organizations may exhibit. It may be worth considering that even the smallest development organizations we visited engage in several hundred different functions. Compare this to our present model, which demonstrates four prototyping functions and twelve information-processing functions.

---

[93] For information on *descriptive, illustrative, normative,* and *man-machine* simulation forms, refer to Cyert (1988), pp. 179-198.

Focus: The present model was developed to examine the nature of interactions between *information processing* and *prototype processing*. In doing so, we have also addressed some interesting effects of information and prototype buffer sizes, resource allocation strategies, and tradeoffs between performance measures. We have not incorporated a multitude of other factors which affect development projects in real organizations. A small sampling of such factors include externally and internally induced requirements changes, labor productivity changes, technological change, cultural dynamics, configuration management methods and delays, supplier relations, cross-functional PDT structures, managerial influences, and non-engineering responsibilities such as preparation of technical documentation, manuals, accounting, purchasing, and human resource tasks. Such factors, and others, need to be investigated and incorporated if one expects to have more representative, realistic models.

Information Value: In the current model, information is treated as a *commodity*. Every "piece" of information is similar in size (evaluated by required processing time) and value (degree of desirability). Few provisions have been made to grade information's influence on the prototyping activity[94], other than as a time distraction. In real systems, of course, information comes in an infinite variety of forms and sizes, occupies variable amounts of time, and may be both helpful and harmful to engineering tasks.

---

[94] Though we have not made provisions for grading information as good or bad, we have developed priority structures to help classify one piece of information as more critical than another. We also have developed a preliminary quality structure, based upon the amount of information processed during the time interval between releases. For more discussion about this preliminary work, refer to Appendix K, Quality Assessment Strategies.

Unidirectional Prototype Movement: Currently, the CPP Structure offers multi-directional information transfer, but only uni-directional prototype movement. Yet, field observations reveal that prototype, as well as information, may travel along multi-directional paths throughout the functional development architecture. The system dynamics inherent in such movement can become extremely complex. Our focus has been on examining how the incorporation of information into the process could interfere with development progress; non-linear movement of prototype materiel could cloud our insight, for now. Non-linear prototype movement can be incorporated into the CPP Structure quite readily, however.

Static Structure: We have presumed that the form and relations between functions is constant. We have observed that functional creation and metamorphous during development can be a key features of real systems. In future work, we expect to develop dynamic CPP Structures (i.e., CPP-like models whose structures *change* as they perform). Also, the incorporation of *dynamic parameters* could provide some additional insight into the learning (and forgetting) behavior of real development systems.

Requirements Definition: Though we have observed numerous field problems of incomplete and/or changing requirements during development, our current model treats such requirements as fixed. This is a discrepancy which should be incorporated into future evolutions of the model. Mechanisms for incorporating such a feature are not difficult to establish. Some reasonable *algorithms* for changing requirements during the process, however, need to be developed. Lacking adequate field data for such dynamics[95], we chose to omit this feature at this time. Future models which integrate *changing requirements definitions*, *requirements diffusion processes*, and *information-prototype processing tradeoffs* would provide even better managerial insights than we have developed in this study.

---

[95] We had access to plenty of field data which demonstrated the number and timing of requirements changes. We still lack, however, fundamental drivers for these changes.

Empirical Evidence/Validity: The CPP Structure was developed from observations of, and participation in, actual development projects. Though many of the structures and behaviors observed in the field can be "seen" in our model, we have a validation dilemma which faces many simulations. On the one hand, it is difficult to relay the characteristics of development without such models. On the other hand, such models only possess *face* and *content* validity. Criterion based validation methods, using empirical evidence, cannot be used in good faith for two reasons:

1) The models are simplified conceptual renderings. Direct use of "real parameters" may be inappropriate, due to alternative semantics.
2) Objective parametric data is not generally available for these models at present. We are examining aspects of these processes which are largely outside the domain of most development managers.

In time, we hope that CPP-like visualizations of development processes can help us obtain adequate data to better refine and verify model structures and behavior. As development is more widely recognized as a non-linear, contingent process, bearing the types of behavior we identify in this study, this eventuality may happen sooner than later.

Such limitations should not be construed as a deterrent for future use, but rather as an opportunity. We are excited about the eventual use of CPP-type analysis techniques in actual development organizations. We briefly discuss some of these possibilities in the next chapter. Based upon just our field studies and the dynamics of this model, we have gained enough insight to offer several concrete suggestions for manager of product development. These are also discussed in the next chapter.

CHAPTER V: IMPLICATIONS AND CONCLUSIONS

"Please would you tell me," said Alice, a little timidly, "why your cat grins like that?"
"It's a Chesire Cat," said the Duchess, "and that's why."

--Lewis Carroll

"Men occasionally stumble over the truth, but most of them pick themselves up and hurry off as if nothing had happened."

(Attributed to Winston Churchill)[96]

This chapter is dedicated to addressing real managerial concerns. We consider how the CPP methodology may be used by managers of development projects. We contemplate improvement strategies for managers who are concerned about obtaining better results from their development organizations. And we look towards the future of new product development, upon reflection of the findings from this study.

The remainder of this chapter is divided into three sections:

- Use of the CPP Methodology
- Performance Improvement Guidelines for Development Management
- Looking to the Future

[96] Arthur T. Winfree, The Timing of Biological Clocks (New York:Scientific American Books), p. 47.

5.1. Use of the CPP Methodology

The CPP methodology was developed with the philosophy that one cannot satisfactorily manage a process without thorough *understanding* of that process. Once one has developed suitable understanding, then appropriate analysis and management may begin. There exist many process improvement methodologies. The recent craze of business process re-engineering (BPR) has increased their popularity. Unfortunately, many process documentation efforts stop with documentation; follow-up analysis methods then suffer from inadequate use. Analysis methods which are used can be inappropriate, due to inadequate understanding (by analysts) of the nature of development processes.

Existing methodologies can assist in documenting structural aspects of organizational processes. Unfortunately, such tools only offer static descriptions or "pictures" of processes. Often, they offer historical ***indications*** (or historically based predictions) of the net result (e.g., How much did the process cost? How long did it take? What throughput was delivered? What quality level was generated?, etc.) They do not satisfactorily portray actual process behavior (e.g., Why did it cost so much? Why did it take so long? Why was throughput at that level? Why was quality so good or bad?, etc.) The CPP methodology has been developed as a tool to document *and* analyze development processes. It was specifically developed to help managers examine dynamic, often turbulent aspects of development processes. Thus, CPP-type analyses can help investigate the ***why's***, not merely the what's, of development process behavior.

The CPP methodology is oriented around the simulation of *non-linear* processes. Manufacturing-based simulations, though appealing, do not lend themselves very well to

these types of processes[97]. By incorporating high feedback rates (which are realistic in product development), CPP offers a more non-reductionist view and analysis of the process. By incorporating visual, dynamic representations of the system, we ***expect managers and engineers to participate in CPP model construction and analysis.***

As we have discussed in this study, there can be non-intuitive system-wide results from "improvements" in human resource allocations, information processing efficiency, prototype processing efficiency, and other areas. What analyses have offered before, for instance, that improvements in engineering function processing could actually *reduce* overall development system performance? Such findings are just the beginning, we believe, of the kinds of insights which can be obtained by looking at development organizations from a non-reductionist, dynamic point of view.

Thus far, it is apparent that there exist two basic categories of product development for which CPP methods can prove beneficial. The first category is ***new product development.*** This is the case for which there have been few, if any, process precedents established for development management. Not only may the process be new, but the product undergoing development has no baseline (i.e., it is not a mere twist on an existing product concept). Such a process is considered by some as *innovation*. Usually, new product development processes resemble incremental innovation rather than seminal innovation, however.

---

[97] In fact, in developing the CPP Structure, we were told by some software engineers from very reputable manufacturing companies that their products were not designed for high-feedback behavior, because they *had not been approached* with this need. In fact, for the dynamic analysis in this report, we utilized such a manufacturing simulation program, incorporating very high levels of rework. Though painstaking, one can *make* it work!

The other development class for which CPP can prove useful is ***routine product development***. In this case, the product being developed is one more in a line of existing products. The product does not differ fundamentally in concept from previous developments; even the development organization is largely similar from product to product. In some cases, routine product development is contemplated as a stable, predictable process, not too dissimilar from a manufacturing process.

The CPP methodology may be helpful to managers of new product development in at least two ways. First, the dynamics of the model, even as established thus far, are indicative of the *types of delays* and uncertainties to expect in a new product development project. The observation that the first design takes much longer to develop than subsequent design intervals (in our CPP analysis, this ratio ranged between 1.58 and 22.3, with an average of 9.74) is indicative of a learning process within the organization, even though we have not incorporated "learning" into the processing rates. This delay is actually a result of both pipeline filling and information "catch-up" by all departments. By constructing and analyzing a CPP-type model, one can make better judgements about and delineation between the true reasons for such types of performance peculiarities.

Second, managers of new product development could utilize CPP *to anticipate the magnitude of delays* and *associated expenses* for an interruption in any part of the organization at any given time. Developers of new software, for instance, would be wise to heed the need for accurate, stable requirements before engaging in detailed code writing[98]. Compared to the catch-it-as-it-comes managerial style often employed today,

---

[98] A popular existing approach to avoiding the uncertainty of such requirements changes is to develop products in more simplistic stand-alone "modules", so that requirements changes may only require

CPP can serve as a *development road map*. For such purposes, even a simple CPP model such as we have presented in this analysis could prove useful.

For managers of both routine and new product development, there are many other potential benefits of exploiting the CPP methodology. Because managers of routine product development typically have some experience from previous developments, and have more predictable processes, CPP can be used in a more detailed, more interactive manner. A sampling of possible uses include the following:

- As a ***real-time resource management tool***, CPP could offer managers the ability to "see" shortages and excesses of human resources, materials, or equipment.
- As an ***information transfer illustration tool***, CPP demonstrates the frequency and timing of communications between various players in the development project. Thus, it can be used as a mechanism to determine when and where information systems need improvement, and where there are currently excess capabilities.
- As an ***orientation tool*** for newer managers, CPP can be utilized to better communicate the effects of interdependencies of functions, at an earlier stage in their professional development. This could help accelerate the knowledge base of developing managers.
- As a ***real-time, dynamic statistical analysis tool***. Due to inherent uncertainties in the process, it is often difficult to predict the results (time, cost, prototype performance) of *specific* activities in development. With CPP, however, aggregates of activities can be modeled, so that managers can have more realistic estimates of best-case and worst-case scenarios.
- As an ***explanatory mechanism***. For development managers and their executives, budget, time, and resource requests are an everyday part of business. The CPP methodology can be utilized as a demonstration device for internal communication

---

modifications of a few modules, not the entire product. Though this approach is laudable, we have observed ever increasing amounts of module-to-module interfacing, which can eventually cause similar problems of the pre-module development strategy.

with those not familiar with process. For manufacturing, this is done routinely. Where are the executive or managerial models of engineering? Even a simplified CPP model can assist this process.

- As a ***real-time process analysis tool.*** For enterprises without sophisticated process analysis departments, CPP-based analysis can help managers better "see" their processes in action, document the results of alternative parameters, and even perform many "what-if" scenarios. Just as engineers build FEA models to evaluate alternative physical designs, managers can use CPP to evaluate their processes, without the risks of actually implementing unfavorable scenarios.

- As a ***process documentation tool*** (and recovery from "re-engineers"). Working with existing documentation methodologies, such as IDEF, the existence of an "as-is" CPP model can offer helpful reminders of critical process needs, lest some consultant(s) determine they are going to re-engineer the process from scratch!

- As a ***process integration telltale.*** Using CPP, managers can gain a better understanding of how their processes affect the processes of their colleagues. This can expedite and improve product integration during development.

- As a ***bottleneck finder.*** A CPP model, when running in real-time demonstrates how moving bottlenecks can rapidly transfer across the organization and blind-side unsuspecting players. If a "live" CPP model (in synchronization with the real organization) were utilized, managers could instantly see where the development process is being strangled.

Many other specific uses can be expected, as managers and/or their engineering process analysts document their organizations into specific CPP models. For instance, we have already developed a variety of quality prediction techniques for use with the existing, simple demonstration model[99]. Activity-based cost analyses can be drawn from the model with relative ease. Of course, development time has been the focus of this analysis. Based upon our observations in the field, and the demonstration CPP structure developed in this study, we are confident that development time can be significantly shortened, along with improvements to quality, and cost. Much of this, however, depends upon the dedication which managers have towards accomplishing these goals.

---

[99] For more information on some of this work, refer to Appendix K.

Based upon our experience during the development of the CPP methodology, CPP does not have to be limited to product development organizations. We have observed non-engineering organizations which appear well-suited to the CPP methodology. These include human resource management, logistics, even marketing program development. Fundamentally, dynamics illustrated by the CPP methodology can be similar across a wide range of organizations. This is particularly true of organizations charged with the task of performing some project never conducted before.

5.2. Performance Improvement Guidelines

5.2.1. Consideration One--What is the Goal Here, anyway?

When one hears the word "innovation," it is easy to imagine a Thomas Edison-type individual, trying to *sell by day* inventions dreamt-up the night before. There seems to be a stereotypical understanding of what comprises innovation: Mix a little invention with a strong entrepreneurial spirit, and engage in a swashbuckling marketing blitz to communicate newfound benefits to a previously ignorant populous of buyers. If this scheme makes the inventor and his colleagues wealthier than their wildest dreams, then they have "successfully" innovated. If the new idea doesn't pan out, then they have failed to innovate. Even separating the inventor of a concept or product from the marketer of the product does not fundamentally change this vision of innovation.

Such an image may stem from a lack of suitable definition. Masked in a cloud of subjective (and often, *ex post facto,* pecuniary) measures, innovation has been defined in a variety of ways. This definitional array reflects many personal images (and resulting frameworks) which authors have developed. Unfortunately, no single framework or image has yet shown or sufficiently described the ever more dynamic and complex concept of innovation.

Likewise, the term "New Product Development" can also conjure up a set of stereotypical images: Engineers (perhaps, in white coats and carrying clipboards!) working in a laboratory, testing, evaluating, cobbling, modifying, tweaking, improving, and refining designs... Customer focus groups, evaluating the appeal of product features...Production engineers, incorporating running changes of designs into their tooling and fabrication

processes (and then re-engineering the process to better meet cost, time, and/or quality objectives)...Marketing and logistics personnel, working closely with engineers to develop cohesive product roll-out and distribution plans which are consistent and cohesive with production schedules...

These images are, at best, optimistic views of new product development. In reality, diverse disciplines which are involved in basic research, applied research, concept development, prototype engineering, requirements definition, full-scale development, testing/evaluation, production engineering, assembly/production, distribution, promotion, field service, and so forth tend to be highly *isolated* from one another. Regardless of official titles, geographic or legal separation of organizations, organizational hierarchies, or process flow diagrams, actual communication processes and patterns are not well-defined, nor simple-to-understand. Unfortunately, integration seems to be "accidentally stumbled upon" in a piecemeal fashion in many organizations we visited. It is rare to find a well-orchestrated (much less well-understood) development process, regardless of the scope of the project being conducted.

New product development is dependent upon the ability to *draw from whatever disciplines are necessary* to complete the task (i.e., develop the new product). It comprises elements of innovation, invention, research, politics, finance, and more using both structured and unstructured methods.

There is a case to be made that new product development is a critical seed for the growth and sustainability of an economy. This synthesis of effort, coming from all sorts of players of varying background, has such important potential consequences that we cannot

afford to pass it off as a bit player "that supports manufacturing." It may well be the *driver* of manufacturing! At the very least, it deserves a good close examination, to see what kinds of tools or methods might be necessary to help make it work better.

We engaged in this study to help open the door to the process of development, not close the book as the final word. During this study we have seen that there is much more to look at--that we are just scratching the surface. Given this ever widening view, there appear to be two important considerations. The *future* and the *how to get there*.

We begin with four basic issues facing management in current development organizations, then consider several suggestions for improvement. In the final section of this chapter, we turn to some concepts which may hold promise as future considerations in the study of new product development.

5.2.1.1. At Issue: What is the Appropriate Managerial Scope?

In the field, we encountered a classical problem, which seems to stifle managers everywhere: *managerial scope*. As documented in the field, and supported in the model, new product development can be a contorted, continuously changing process which seems to have different character, depending upon one's process viewpoint. Changes to the system do not necessarily result in uniform changes to overall system performance, though such changes may drastically affect local conditions within the system. Based upon such views, it is reasonable to conclude that managers should possess holistic, cohesive understanding of the development process.

This can be problematic in organizations of rigid structure and constrained communications[100]. The problem is exacerbated when there is confusion between the need (or want) for information dissemination and the appearance of transferring control or power from one person or organization to another.

Rectification of this problem is not an easy task. As we saw in the model, and have evidenced in the field, there are times when activities have been diligently performed, only to be seen in retrospect as wasted effort, because inadequate or incorrect information had been available[101]. Reducing or limiting one's understanding about a system is a sure method for increasing the likelihood of ill-judgement. Process colleagues need to understand that this affects them as well, and that trustworthiness on such issues is of utmost interest to all involved.

So what is an appropriate scope? This can only by determined by understanding some basics about the system you are managing. One's own department should be considered the bare minimum scope of understanding. How far and in what directions from this "home base" one needs to venture will be a reflection of the nature of the development process, relative to that base. Basic investigations should reveal if and when the process engages in feedback... where that feedback tends to go... who are your "upstream" and "downstream" functional colleagues and why... is that sequence a stable one... what are their process needs which you can help resolve? Find out what downstream functions do

---

[100] It is not limited to large organizations; some of the most autocratic low-communication systems we have encountered were closely-held small organizations.

[101] Anybody who has written a dissertation, I'm sure, could testify to the prevalence of this problem!

with your functional output and show your upstream colleagues what you do with theirs. We also have found it useful to investigate why certain "non-interfaces" have that status. Was this condition policy driven... and is it still relevant? There are many innocuous questions that can be asked, without probing into the responsibility areas of others. If done properly, most others are delighted that you are taking interest in their work. Naturally, to foster trust, you'll need to offer useful aspects of your department to them.

When assessing the performance of the system and relating it to your own sphere of influence, it is important to distinguish between ***process drivers, parametric effects,*** and ***noise***. The proficiency of this skill seems to separate the experienced individuals, regardless of rank, from the not-so-experienced. Process drivers are the *underlying structure* of the system, which remains in place despite specific situations. They may be considered reflective of the overall direction and generally do not vary, despite changes in parameters, administrations, and most technology. Parametric effects are the major system *behavioral responses* to fundamental changes in product requirements, departmental efficiencies, budgets, and so forth. They tend to be shorter in duration and smaller in significance than drivers, but may still significantly affect large numbers of players in the system. These are the types of effects that trigger the creation of investigative process analysis teams. Yet, for the well prepared manager, they are basically expected, though their particular nature may be unpredictable. Then there is process noise. This a highly prevalent effect which developers and managers see regularly, but for which their responses are *only symptomatic*. Noise may be considered random "surface effects," which merely obscure underlying parameter and driver effect.

Astute managers need to be aware of these three classes of effects, and understand their capabilities to respond to and control them. Often, we see managers creating new policies as a direct response to transient system noise, only to ignore process drivers or parameters. This points to the problems of timely and appropriate process indicators. It is also the responsibility of managers to be sure that the indicators being utilized are relevant to the overall process. Further, it is important to use such indicators with care. This means employing a discriminating eye towards the readings these indicators provide.

It is possible to succumb to aliasing problems, as well. Those who are familiar with electronic signal processing methods are intimately familiar with the problems of collecting data at *too low a frequency* to effectively see the true pattern. Naturally, this can happen for organization performance data, as well. There is another problem, however, which is infrequently addressed in this context: *oversampling*. This is the condition whereby too many data points are received over too short a duration. Utilizing statistical techniques, extrapolations of such data may provide incorrect conclusions, if the underlying phenomena has a much longer periodicity than the decision-making process. In many cases, such oversampling is associated with "noise-oriented" management--management that chases symptoms rather than focusing on root causes.

5.2.1.2. At Issue: What does your organization really look like?

Some "process representations" have served to obstruct and misconstrue the view of the actual conduct of new product development. A major reason for this is the inherent rigidity of such representations. Because the development process is in continual flux, it is not convenient to officially change "effective" organizational structures in real-time.

Thus, a static view is only a temporary, often highly simplified representation. In these views, functions are often assigned to specific organizational departments, with the expressed purpose of improving the "understandability index" of the representation. Whether intentional or otherwise, such simple and reductionist views can starve us of the true interfacing between functions. In short, many attempts to ease understanding can severely abate the accuracy of the representation, which is necessary for adequate management of the process. Even our CPP Structure, in its current form, is still too simplistic for managers to use as an integrative tool.

There are notoriously ***many partial views*** of most development processes. Local participants may posses some informal drawings or charts which illustrate their part of the process. Most participants merely carry mental images of how the process works. When we tried to piece these decentralized process understandings together, however, we discovered numerous contradictions between members of the same process!

Often the partial views are falsely enhanced *historical views*--each individual recalls certain features which s/he remembers best, and may unconsciously "fill-in" or leave out other features. In time, actual processes bear less and less resemblance to recalled processes. As a manger, be sure the ***processes you are managing*** are congruent with the ***processes you think you are managing.***

It may sound incredible, but most participants in development organizations had never seen an *overall* process representation of any kind, dynamic or not. It cannot be over-emphasized that accurate process views, despite their trouble and expense, are invaluable

assets to progressive managers. Without such understanding, managers and developers may be described as organizational Cyclopes, possessing little or no dynamic spatial visualization of their processes, and as a result, constantly in *response* mode rather than interactive or proactive *control* mode.

5.2.1.3. At Issue: Traditional Process Improvement Strategies

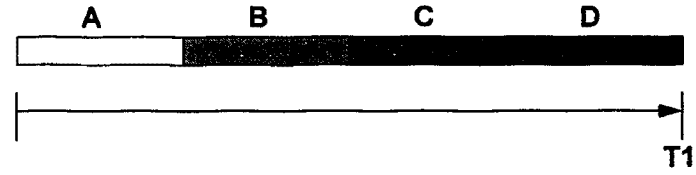
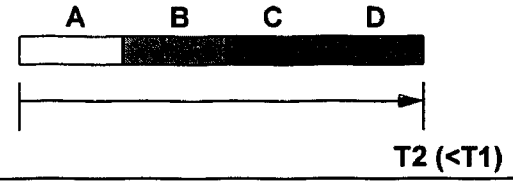
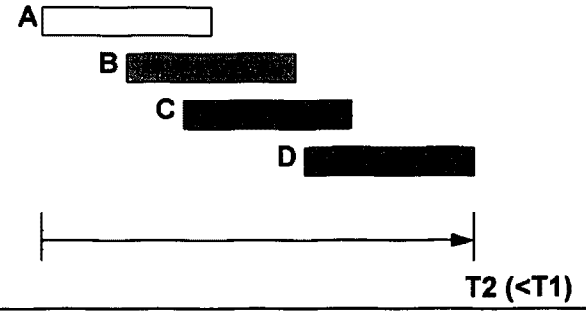
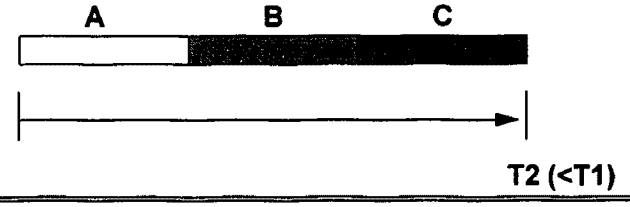
It has become well-established in the minds of many process analysts, particularly in the minds of the management consultants, that there are three solution strategies to the problem of excessive processing time:

1. ***The Technical Strategy:*** *Shorten* the process time for each function (particularly the "critical path" functions) in the system;
2. ***The Concurrence Strategy:*** Enable *simultaneous* operation of functions.
3. ***The Simplification Strategy:*** *Eliminate* non-value-added functions.

Each of these established strategies are characterized in Exhibit V.1.

EXHIBIT V.1.

Traditional Process Time Improvement Strategies

Strategy	Linear Process Representation
Baseline system	
Technical Strategy (shorten processing time for each activity)	
Concurrence Strategy (Overlap activities)	
Simplification Strategy (Eliminate activities)	

Given the current research, it is evident that *none* of these strategies will necessarily guarantee system performance improvement. We have observed technical improvement (e.g., functional efficiency) in local functions which create *excess* work for other functions. Increased levels of activity concurrency can result in *de-stabilizing communicative gaps* and delays. Finally, the simplification strategy is much more *difficult to implement effectively* than its theory implies. How does one determine the

"non-value added" status of an activity? This is equivalent to the problem of separating "information" from "prototype", which we discussed in our initial description of the CPP structure. An activity which may be useful to some, may be "non-value added" to others.

In reflection of the current research, it is apparent that there exist at least two other major strategies which can improve overall (system-wide) process time. I call these the ***reordering*** strategy and the ***re-circulation process improvement*** strategies. Although their effectiveness may vary with the particular system under study, and their temporal requirement for accurate application may be higher than the basic three strategies, we assert that these two strategies can have much more pronounced effects on system performance than any of the other three.

The reordering strategy is basically no more complicated than its namesake implies: consider the totality of activities necessary for development, and arrange them in appropriate order, *according to chronological dependency*. Though this is the basis by which PERT-type diagrams were originally developed, we encountered several situations where the dependency structure being utilized was merely a *facsimile of what had been* used for previous development projects. At times, engineers had little contribution to the official dependency charts. To managers at such organizations, this was considered a natural and beneficial state of affairs, for there was a prevailing belief that "all our development projects are the same," and that re-evaluation of functional dependencies was a waste of time. Yet, our experiences with development engineers told a different story. At times, dependency analyses were reflections of how managers felt they "should look," rather than engineering-derived conclusions of how they *would* look. To presume that a *new* product development process should necessarily resemble its predecessors or

can be effectively drawn from managerial aspirations is, we believe, prescription for incremental, product *evolution*, rather than innovative product *revolution*.

Our observation that adequate evaluation activity ordering is not conducted leads directly into the problem of process re-circulation. We have discussed problems of feedback quite often in this research. With increased confidence, we believe that the problem is *even more acute* than we have addressed in these pages. Yet, there seem to be only cursory solutions to the problem in the field. Through better dependency analysis and functional requirements definition, we expect that the number of design iterations can be substantially reduced[102]. By addressing this problem directly, however, we expect both managers and developers to broaden their process focus, and engage in more integrated and effective development. The next section considers some specific suggestions for realizing this goal.

5.2.1.4. At Issue: Process Stability

As the CPP structure demonstrated, increases in local efficiencies are not necessarily commensurate with improved system performance. In extreme cases, it was possible for the unbridled system to effectively work itself into a degenerate state, whereby the system literally shut itself off.

In real development systems, such system-wide shut-down may seem pretty unrealistic. Yet, we have seen *portions* of development projects halt for varying lengths of time, in

---

[102] During our field studies, we found that developers typically performed the same functions five or more times during a single development project.

conditions similar to the onset of CPP system degeneration. Whenever two aspects in a real development system are in conflict, say because of an ambiguity in product requirements, the process essentially stops until the problem has been rectified.

We consider a development system which stops its progress during the ex post facto critical path to be engaging in ***instability***. The number one priority among managers should be reducing the number and intensity of such instable states during development. After stability has been established one can consider process improvement strategies such as we discussed in the previous section.

How can one reduce the propensity for instabilities? We see a three component strategy, composed of ***functional velocity, size,*** and ***structural complexity***. These are described as follows:

> ***Functional velocity*** is a measure of the number of "turns" which a function performs in the typical development process, over a given time period. Such velocity may be externally or internally generated. Internal generators (e.g., "continuous improvement" of local functions) have been observed as the predominant driver for rising functional velocity. External drivers for functional velocity have been observed to occur less frequently than internal velocity drivers (Such drivers may be illustrated by broad policy changes in response to market changes). When they do occur, however, external drivers seem to have a larger impact on process performance.
>
> Although the concept of reducing velocity to help improve effectiveness is foreign and counter-intuitive to many production-oriented managers and researchers, one must recall that development performance is not graded by how many functions are performed, but rather by how soon the production process can begin (i.e., how soon all ***necessary*** development functions have been ***adequately*** completed).

When the process is operating in a non-linear (high-feedback) mode, faster velocity may result in more iterations, not necessarily less total time. This was observed in the field, and reproduced in the dynamic CPP analysis.

Contrast this with a stable linear process, such as is found in nearly all production processes. In stable systems, one should expect any velocity increase to be desirable.

Size is characterized by the number of personnel, number of functions, number of departments, or number of channels currently in use in the process. Assuming that some scaling or transformation factor between these measures is available, the particular measure used may be immaterial. Size is a factor because is it is an indicator of *communication delays* in process feedback loops. As related in the classical stock management problem[103], with linear (sequential) communication and materiel channels, longer delays can affect the magnitude of fluctuation of inventories. Increasing the number of sequential nodes can effectively delay communications further.

Size is also a concern from a psychological and practical standpoint. We have observed that an increased number of functions, personnel, or departments *reduces* the likelihood of any given person, department, or function to understand the structure of the overall process, much less the behavior of all other functions. In the CPP model of this analysis, the size was limited to a somewhat reasonable 4 departments, each department itself composed of 4 functions. For the particular CPP structure tested, these 16 functions interfaced via 147 distinct communication channels. Without the use of automated data acquisition, the dynamic behavior of even this simple-case development "organization" would have been unreasonable to track.

---

[103] For an interesting account of the stock management problem known as "The Beer Game" refer to Mosekilde et al (1991).

> ***Structural complexity*** is an indicator of the *resistance* of an average communications channel between two functions within the organization. It is composed of factors of hierarchical structure, number of intermediate "junctions" between two independent information sources, and information transfer media. Naturally, any hierarchical organization has a distribution of complexities; individuals higher in the organizational hierarchy may encounter less communicative resistance than operations level personnel. Further, it is proposed that simplicity of process is inversely related to the simplicity of supporting information systems and technologies. Thus, a "simplified process" appears more likely to require more sophisticated information transfer mechanisms, just to keep it "simple!"
>
> In the CPP Structure analyzed in this research, we assumed that all hierarchies were "decomposed" to the lowest functional levels (the CPP Structure itself is based upon a flat functional structure, not a hierarchical structure), and structural complexity values were assumed to be known. Specific complexity values were applied through a priority system for information and prototype processing. Changes to channel priorities are easily accommodated in the CPP Structure.

It is clear that ***velocity, size,*** and ***complexity*** are not independent functions of one another. Yet, our observations of large, medium, and small companies suggests that reduced velocity, size, and complexity results in better performance. Because of the interdependence of these three factors, however, it is likely that each industry, organization, and specific development project has its own complexity, velocity, and size relationship. For instance, in today's competitive industries, it may be very difficult to reduce the externally driven component of velocity. Multi-disciplinary products may require more specialists, which may increase the size and complexity of interactions.

5.2.2. Consideration Two--Suggestions for Improvement

We ask for a different role of managers: Stop ***using*** convenient, customary, inappropriate tools--Start ***thinking*** about and start ***developing*** appropriate, site-specific tools. Management of processes as complicated and complex as new product development should probably not be conducted according to some recipe of techniques. Rather, it should be conducted in a manner which enables the most complete and accurate assessment of situations, in the appropriate context.

There are two basic objectives for our suggestions:

1) ***Stability Attainment:*** Those which are offered to help organizations *reduce their level of instability*;
2) ***Stability Preservation:*** Those which are intended to help organizations *from falling back into an unstable mode*, and improve the *effectiveness* of their new product development process.

Our three basic suggestion categories are ***situation analyses, communication capabilities,*** and ***training***.

Situation analyses

From a managerial point of view, this can be considered the "thinking/evaluation" category. Participants throughout development projects need to consistently review their operations from an ***effectiveness*** point of view, not just an ***efficiency*** point of view. It requires an assimilation of varying perspectives and objectives among participants. Specifically, situation analyses need the following characteristics:

- DECENTRALIZED: It should be performed at all levels by all personnel.
- CONTINUOUS: Participants at all levels need to constantly ask why their function adds value to the process.
- RELEVANT INDICATORS: Establishment of relevant, unambiguous *process indicators* is very difficult. However, such indicators can provide an objective, consistent, well-understood incentive framework.
- EARLY: Development organizations cannot afford to be bashful of conducting many engineering changes prior to Full-Scale Development. Making changes (resolving disputes) early in the process can be much less expensive than late or not at all.
- INTERFACE: Emphasize Interface monitoring over financial or even schedule monitoring. Most budgets and schedules are notoriously optimistic and are ignored by the real innovators in organizations. If the mission is well understood and agreed upon by developers, then gains from successful new/innovative development will dwarf the returns of penny-wise tactics of many managers.
- PRIORITY--(INFORMATION vs. PROTOTYPE): Diligent decentralized efforts should be made towards recognizing the differences between prototype and information. By separating these elements (or at least establishing a philosophical separation between them) better process prioritizations can be made. If development organizations expect to behave in "war-time" mode in the future, then one must be able to perform triage on unnecessary tasks. Separating prototype from information is a first step in this cultural transformation.

- COST: Stop worrying about development cost. Let developers focus on quality (i.e., meeting customer requirements) and production costs. In doing so, development costs per unit may actually fall, as more high-quality (and high-return) products are created.

- CONTROL/ADMINISTRATION: Think about ways of helping the overall process, not administrative ways of controlling the actions of engineers. Engineers are bright people; too often they are treated like children by administrators.

Communication capabilities

Though "better communication" seems to be a perpetual conclusion in the business literature, it is important enough to re-emphasize here. Given our observation that communication deficiencies are still prevalent, it is safe to conclude that this category has not been heeded sufficiently. Particular attention needs to be made on the HIGH FREQUENCY/LOW AMPLITUDE nature of such communication. This helps compensate for many of the cognitive limitations which we, as humans, posses.

- OPEN: Establish open communications channels between *every* employee (from CEO to the newest hiree). This implies

  $$\frac{N \cdot (N-1)}{2}$$

  communication channels for an organization of N people. For large organizations, the number of interfaces can seem unreasonable. If the organization has grown this far, then perhaps a breakup into smaller development organizations is in order.

- 2-WAY: Make sure that each communication channel is *bi-directional*. Open door policies should not be established exclusively for superiors and peers. Subordinates need to be welcomed as part of managerial solutions. There are many simple ways that this can be accomplished. Most often, however, "executive attitude" interferes with this strategy. Candor should not be treated as a cancer.

- FREQUENT/SIMPLE: Endorse high frequency communications throughout. Coupled with such frequency, reduced amplitude signals (especially if repeated) will be better understood. Contrast this with existing practices of monthly or quarterly management meetings, within which only a few, highly publicized issues get attention.

- INTERDISCIPLINARY: Require all personnel to meet with at least one different employee each day -- from outside of one's own department.

- VOCABULARY: From a cultural-linguistic standpoint, there are many interesting uses of terminology during development. Unfortunately, such semantics are often incongruent *among members of the same development project*. By establishing a common vocabulary base, communication precision will improve. This is particularly important during requirements definition and

diffusion. It also takes on increasing importance when engaging in multi-firm development projects.

- KNOWLEDGE-BASED: Develop on-line system for all engineers to gain first-hand access to company specialists via real-time interest area databases. Personnel departments are typically full of useful information that is rarely tapped. Hence, most personnel departments become one-way repositories of information, which become out of date over time. Turn this around.

- PRO-ACTIVE: Establish technical information transfer services which *anticipate* the needs of developers, by understanding their process.

- CUSTOMER ORIENTED: Endorse field visits by engineers. Recall that development is a low proportion of total life cycle cost of the product, but may have great effect on the revenue capability of the products. Successful products satisfy the ***customer requirements***, not ***managerial requirements***.

Training

Although specialization has been the watchword for over the last century, we conclude that superior product development requires *an interdisciplinary approach* which will necessitate a new attitude towards orienting and nurturing development participants.

- PROCESS, NOT-POLICY: Use activity or process diagrams of entire development process for training new employees. Rather than just instructing ***what*** the company policies are, demonstrate ***why*** certain actions are needed to assist other functions within the process.

- FIRST-HAND: Engage in intensive "walk-throughs" of the entire development process (from concept initiation through the assembly line). This should be done by ***all*** engineers and managers. This provides a visual image to aid in recall for interfacing, for instance.

- INTERDISCIPLINARY: Cross-train employees. The more they know about fields outside their specialty, the more capably they will experiment with new techniques from those fields. In doing so, more integrative (and more innovative!) solutions can be expected.

- DYNAMIC: Teach principles of dynamics of the organization to all managers (including, but not limited to system dynamics, production principles, logistics, etc.). Communicate how and why the process is continually undergoing change, so that all participants can better adopt and control such change in a reasonable manner.

- ON-GOING: Effective learning is a never-ending activity. Effective managers constantly seek more information about how their organizations operate. By establishing ***and maintaining*** accurate knowledge about the pulse of their processes (and the state of fellow personnel in these processes), their environmental awareness is improved. This can only enhance their decision-making capability. In this vein, the above principles need to be carried out...not once, ***but over and over again***.

Table V.1. is a basic mapping of these basic suggestions with the ***stability attainment*** and ***stability preservation*** objectives. Naturally, each organization can and should develop its own mission, objectives, goals, and specific strategies for development. Stability attainment, however, appears to be a precursor to objectively managing development in a highly effective manner. Once the process is stable, then a variety of techniques (even many traditional techniques) can be used to improve performance. Care must be taken, however, to keep the process from relapsing into instability. This is the purpose of the stability preservation classification.

TABLE V.1.

Suggestion Category	Stability Objective: Stability Attainment	Stability Objective: Stability Preservation
Situation Analysis	• Decentralize decision-making • Establish process indicators • Continuously evaluate functional requirements • Ignore Development Costs	• Use process indicators • Heed early warnings • Focus on Interface needs • Separate Information from Prototype • Control/Administration
Communication Capability	• Establish open bi-directional channels • Establish information dissemination centers • Establish concise corporate vocabulary	• Interact with multiple disciplines • Pro-active information transfer • Maintain Customer orientation • High-Frequency/Low Amplitude
Training	• Walk-throughs • Cross-training	• Dynamic Analysis • Process before Policy

5.3. Looking to the Future

Prior to this study, there was some suspicion that little was left to be uncovered regarding New Product Development. Upon closer examination, however, we have found that many long-held beliefs about product development were not necessarily true. We have discovered many new areas to explore; some areas may be esoteric while some may hold rich practical promise. Regardless, it is clear that there is quite an interesting future ahead for researchers of new product development.

The CPP methodology, when further developed, will open the door to dynamic management tools; not only to control non-linear process, but to understand dynamic cultural process within organizations, as well.

We visualize NPD projects as continuously changing processes which are not easily represented by static modeling structures. The development of advanced, dynamic, continuously changing structures, which incorporate changing requirements and participant behavior could be an exciting area. Future analysis and measurement techniques could include cellular automata, dynamic fractal structures, chaos and anti-chaos theory, virtual reality and a variety of yet to be developed process policy/decision tools.

Perhaps the brightest and most intriguing paths ahead belong to those who consider the effects of interaction between disciplines. As we become more specialized, NPD will have a growing need to become more integrated. How this might change our view of specialization and reshape the research frontier is a question worth investigating.

These, and many more new concepts are being defined and refined for future use. Meanwhile, further resolution of the CPP methodology will make it a most useful tool for the product development executive, process manager and design engineer currently struggling in today's real development systems.

Throughout history, the only constant has been change, but it has been a painful and fearsome constant. With the development of new tools to help us understand and facilitate our processes for responding to change, perhaps change could become, if not pleasant, then at least less fearsome and painful.

BIBLIOGRAPHY

Ackoff, R., *The Art of Problem Solving*, New York:John Wiley, 1978.

Ackoff, R., "The Future of Operational Research is Past", *Journal of the Operational Research Society*, Vol 30, No 2, 1979, pp. 93-104.

Altshuler, A.; Anderson, M.; Jones, D.; Roos, D.; and Womack, J., *The Future of the Automobile*, Cambridge (MA):MIT Press, 1984, pp. 1-321.

Amendola, M. and Gaffard, J.L., *The Innovative Choice*, New York:Basil Blackwell, 1988.

Anderson, J.R., *The Architecture of Cognition*, Cambridge, MA:Harvard University Press. 1983.

Anderson, J.R., *Cognitive Psychology and its Implications*, 3rd edition, New York:W.H. Freeman, 1990.

Argyris, C.; and Schon, D.A., *Theory in Practice: Increasing Professional Effectiveness*, San Francisco:Jossey-Bass, 1974.

Armstrong, J.S., *Long Range Forecasting: From Crystal Ball to Computer*, 2nd edition, New York:Wiley, 1985.

Arrow, K.J., "A Difficulty in the Concept of Social Welfare," *Journal of Politcial Economy*, Vol 58, 1950, pp. 328-346.

Austin, J.L., *How to do Things with Words*, Cambridge, Mass:Harvard University Press, 1962.

Ayer, A.J., *The Problem of Knowledge*, London:Penguin, 1956.

Bacon, G; Beckman, S.; Mowery, D; and Wilson, E, "Managing Product Definition in High-Technology Industries: A Pilot Study", *California Management Review*, Spring 1994, pp. 32-56.

Baily, M.N., and Chakrabarti, A.K., *Innovation and the Productivity Crisis*, Washington, D.C.:Brookings, 1988.

Baker, G.L. and Gollub, J.P., *Chaotic Dynamincs: An Introduction*, New York:Cambridge Univ. Press, 1990.

Bartholomew, D.J., *Social Applications of Semi-Markov Processes, in Semi-Markov Models:Theory and Applications*, New York:Plenum, 1986, pp. 463-474.

Bauer, H., *Scientific Literacy and the Myth of the Scientific Method*, Urbana:University of Illinios Press, 1992.

Baumol, W.J., "Entrepreneurship: Productive, Unproductive, and Destructive," *Journal of Politcal Economy*, vol 98, no 5, 1990, pp. 893-921.

Beaumont, J.C.; Young, A.W., and McManus, J.C., "Hemisphericity: A Critical Review," *Cognitive Neuropsychology*, Vol 1, 1984, pp. 191-212.

Bell, D.; Keeney, R.L.; and Raiffa, H., *Conflicting Objectives in Decisions*, London:John Wiley, 1977.

Bellman, R. and Zadeh, L.A., "Decision-Making in a Fuzzy Environment," *Management Science*, Vol. 17, 1970, pp. 141-164.

Benedict, R., *Patterns of Culture*, Boston:Houghton-Mifflin, 1934.

Berger, S.; Dertouzos, M.L.; Lester, R.K.; Solow, R.M.; Thurow, L.C., "Toward a New Industrial America," *Scientific American*, June 1989, pp. 39-47.

Blattberg, R.C. and Hoch, S.J., "Database Models and Managerial Intuition: 50% Model + 50% Manager," *Management Science*, Vol 36, No 8, August 1990, pp. 887-899.

Blumenthal, A.L., "Psychology and Linguistics:The first Half-century", Lecture delivered at APA centennial symposium, New York, 1979 (Per Gardner, 1987).

Bobrow, D.G., "Natural Language Input for a Computer Problem-solving System", In *Semantic Information Processing*, M.L.Minsky, ed., Cambridge, MA:MIT Press, 1968.

Bobrow, D.G.;and Winograd,T., "An Overview of KRL, a Knowledge Representation Language", *Cognitive Science*, No 1, 1977, pp. 3-46.

Boden, M., *Artificial Intelligence and Natural Man*, New York:Basic Books, 1977.

Boucher, Thomas, "Adam Smith & the Humanists: An Enquiry into the Productivity of Labor Controversy", *IIE Transactions*, Vol. 20 #1, March 1988, pp. 73-82.

Bourgeois, L.J. and Eisenhardt, K.M., "Strategic Decision Processes in High Velocity Environments: Four Cases in the Microcomputer Industry", *Management Science*, Vol 34, No 7, July 1988, pp. 816-835.

Bower, Bruce, "Darwin's Minds", *Science News*, Vol 140, No 15, October 12, 1991, pp. 232-234.

Bower, Bruce, "True Believers", *Science News*, Vol. 139, January 5, 1991, pp. 14-15.

Bowers, R.V., "The Direction of Intra-societal Diffusion", *American Sociological Review*, vol 2, 1937, pp. 826-36.

Bowers, R.V., "Differential Intensity of Intra-societal Diffusion", *American Sociological Review*, vol 3, 1938, pp. 21-31.

Brady, M. and Lee, H., "Is there an Allais Paradox? A Note on its Resolution", *Psychological Reports*, Vol. 64, 1989, pp. 1223-1230.

Branch, Ben, "Research and Development Activity and Profitability: A Distributed Lag Analysis", *Journal of Political Economy*, Vol. 4, No. 5, September/October 1974, pp. 999-1011.

Braybrooke, David, *Philosophy of Social Science*, Englewood Cliffs:Prentice-Hall, 1987.

Braybrooke, David and Lindblom, Charles E., *A Strategy of Decision*, New York:Free Press of Glencoe, 1963, pp. 1-268.

Bresnan, J., "A Realistic Transformational Grammar", In *Linguistic Theory and Psychological Reality*, M. Halle, et al., eds., Cambridge, MA:MIT Press, 1978.

Bresnan, J., "An Approach to Universal Grammar and the Mental Representation of Language", *Cognition*, Vol 10, 1981, pp. 39-52.

Broadbent, D.E., "The Role of Auditory Localization in Attention and Memory Span", *Journal of Experimental Psychology*, Vol 47, 1954, pp. 191-96.

Brock, W.A., "Distinguishing Random and Deterministic Systems", *Journal of Economic Theory*, 40, 1986, pp. 168-95.

Brock, W.A., "Chaos and Complexity in Economic and Financial Science", in *Acting Under Uncertainty--Multidisciplinary Conceptions,* G.M. Von Furstenberg, ed. Boston:Kluwer Academic Pub., 1990.

Brock, W.A. and Sayers, C.L., "Is the Business Cycle Characterized by Deterministic Chaos?", *Journal of Monetary Economics*, 22, 1988, pp. 71-90.

Brockhoff, K., "Decision Quality and Information", *Empirical Research on Organizational Decision-Making*, E. Witte and H.-J. Zimmermann, eds., Amsterdam:Elsevier Science Pub., 1986, pp. 249-265.

Bronner, R., "Perception of Complexity in Decision-Making Processes:Findings of Experimental Investigations", *Empirical Research on Organizational Decision-Making*, E. Witte and H.-J. Zimmermann, eds., Amsterdam:Elsevier Science Pub, 1986, pp. 45-64.

Bronner, Rolf, *Decision Making Under Time Pressure*, Lexington (MA):DC Heath, 1982.

Brown, Lawrence, *Innovation Diffusion: A New Perspective*, New York:Methuen, 1981.

Brown, R., "Language and Categories", In J.S. Bruner et al, *A Study of Thinking*, New York:John Wiley, 1956.

Brown, Rex V., Kahr, Andrew S., and Peterson, Cameron, *Decision Analysis: An Overview*, Holt, Rinehart and Winston: New York, 1974.

Bruner, J.S., *In Search of Mind*, New York:Harper & Row, 1983.

Bruner, J.S.; Goodnow, J.; and Austin, G., *A Study of Thinking*, New York:John Wiley, 1956.

Buerger, Martin J., *Contemporary Crystallography*, New York:McGraw-Hill, 1970.

Burke, James, *The Day the Universe Changed*, Boston:Little, Brown, 1985.

Carpenter, T.P. and Moser, J.M., "The Development of Addition and Subtraction Problem-Solving Skills", in *Addition and Subtraction:A Cognitive Perspective* (Carpenter, Moser, and Romberg, eds.), Hillsdale, NJ:Erlbaum, 1982.

Carraher, T.N.; Carraher, D.W.; Schliemann, A.D., "Mathematics in the Streets and in the Schools", *British Journal of Development Psychology*, Vol 3, 1985, pp. 21-29.

Casti, John L., *Paradigms Lost: Images of Man in the Mirror of Science*, New York:Morrow, 1989.

Casti, John L., *Searching for Certainty: What Scientists Can Know About the Future*, New York, William Morrow, 1990.

Casti, John L. and Karlqvist, Anders, eds., *Beyond Belief: Randomness, Prediction and Explanation in Science*, Boca Raton (FL):CRC Press, 1991.

Cazeneuve, J., *Lucien Levy-Bruhl*, translation by P. Riviere, Oxford:Basil Blackwell, 1972.

Cherry, E.C., "Some Experiments on the Recognition of Speech with One and with Two Ears", *Journal of the Acoustical Society of America*, Vol 25, 1953, pp. 975-979.

Chiarella, C., "The Elements of Non-Linear Theory of Economic Dynamics", PhD Thesis, University of South Wales, 1986.

The Chicago Manual of Style, 13th edition, Chicago:University of Chicago Press, 1982.

Chomsky, N., "A Transformational Approach to Syntax", in J.A. Fodor and J.J. Katz, eds., *The Structure of Language: Readings in the Philosophy of Language*, Englewood Cliffs (NJ):Prentice-Hall, 1964, pp. 211-245.

Chomsky, N., "The Logical Structure of Linguistic Theory", Monograph of original PhD. diss., University of Pennsylvania, 1975, .New York:Plenum Press 1975.

Chomsky,N., *Syntactic Structures*, The Hague:Mouton, 1957.

Chomsky,N., *Aspects of the Theory of Syntax*, Cambridge (MA):MIT Press, 1965.

Churchman, C.W., *The Systems Approach and Its Enemies*, Mew York:Basic Books, 1979.

Churchman, C.W. and Mason, R.O. (eds.), *World Modeling*, Amsterdam:North Holland Publishing, 1976.

Churchman, C.W. and Schainblatt, A.H., "The Researcher and the Manager: A Dialetic of Implementation", *Management Science*, Vol 11, No 4, February 1965, pp. B69-B87.

Clowes, M., "On Seeing Things", *Artificial Intelligence*, Vol 2, 1971, pp. 79-116.

Cohen, B.; Harwood W.T.; and Jackson, M.I, *The Specification of Complex Systems*, Wokingham (England):Addison-Wesley, 1986.

Cohen, G., *The Psychology of Cognition*, New York:Academic Press, 1977.

Cohen, J. and Stewart, I, *The Collapse of Chaos: Discovering Simplicity in a Complex World*, New York:Viking, 1994.

Colding-Jorgensen, M., "A Model for the Firing Pattern of a Paced Nerve Cell", *Journal of Theoretical Biology*, 101, 1983, pp. 541-568.

Collins, Allan, "Why Cognitive Science?", *Cognitive Science*, Vol. 1, No 1, 1977, pp. 1-2.

Cornell, Alexander H., *The Decision-Maker's Handbook*, Englewood Cliffs (NJ):Prentice-Hall, 1980, pp. 1-262.

Crawford, C.Merle and Gerard J. Tellis , "The Technological Innovation Controversy", *Business Horizons*, Vol. 24, No. 4, July/August 1981, pp. 76-88.

Cyert, Richard M., *The Economic Theory of Organization and the Firm*, New York: New York University Press, 1988.

Cyert, Richard M. and Mowery, David, C., eds, *Technology and Employment: Innovation and Growth in the U.S. Economy*, Panel on Technology and Employment Committee on Science, Engineering, and Public Policy, National Academy of Sciences, Washington (DC):National Academy Press, 1987.

Davidson, Andrew and Morrison, Diane, "Social Psychological Models of Decision Making", *Research in Marketing*, Supplement #1, 1982, pp. 91-112.

Day, R.H., "Irregular Growth Cycles", *American Economic Review*, 72, 1982, pp. 406-414.

Deming, W.E., *Out of the Crisis*, MIT Center for advanced Engineering Studies, 1986.

Dennett, D.C., "Cognitive Wheels: The Frame Problem of A.I.", Unpublished paper, Tufts University, 1983.

Dennis, M., "Language Acquisition in a Single Hemishere: Semantic Organization", in D. Caplan, ed., *Biological Studies of Mental Processes*, Cambridge, MA:MIT Press, 1980.

Dennis, M.; and Whitaker, H.A., "Language Acquisition Following Hemidecortication: Linguistic Superiority of the Left over the Right Hemisphere", *Brain and Language*, Vol 3, 1976, pp. 404-33.

Dertouzos, M.L.; Lester, R.K.; Solow, R.M.; and the MIT Commission on Industrial Productivity, *Made in America: Regaining the Competitive Edge*, Cambridge (MA):MIT Press, 1989.

Dewdney, A.K., "Computer Recreations: A Cellular Universe of Debris, Droplets, Defects, and Demons", *Scientific American*, Vol 261, No 2, August 1989, pp. 102-105.

Dixon, J.R.; Libardi, E.C.;Luby, S.C.;Vaghul, M.; and Simmons, M.S., "Expert Systems for Mechanical Design: Examples of Symbolic Representations of Design Geometries", *Applications of Knowledge-Based Systems to Engineering Analysis and Design*, November 17-22, 1985, pp. 29-46.

Douglas, G.; Kemp P.; Cook, J., *Systematic New Product Development*, Aldershot, England:Gower Publishing, 1983.

Dow, Gregory W., "Information, Production Decisions, and Intra-Firm Bargaining", *International Economic Review*, Vol 29, No 1, February 1988, pp. 57-78.

Dresher, B.E.; and Hornstein, N., "On Some Supposed Contributions of Artificial Intelligence to the Scientific Study of Language", *Cognition*, Vol 4, 1976, pp. 321-348.

Dreyfus, H., *What Computers Can't Do: A Critique of Artificial Reason*, New York:Harper & Row, 1972.

Drucker, P.F., *Innovation and Entrepereneurship*, New York:Harper & Row, 1985.

Drucker, P.F., *The New Realities*, New York:Harper & Row, 1989.

Drucker, P.F., *Managing for the Future*, New York:Dutton, 1992.

Dyer, James S., "Remarks on the Analytic Hierarchy Process", *Management Science*, Vol 36, No 3, March 1990, pp. 249-258.

Dym, C.L., ed., *Applications of Knowledge-Based Systems to Engineering Analysis and Design*, New York:American Society of Mechanical Engineers (ASME), 1985.

Easton, Allan, *Complex Managerial Decisions Involving Multiple Objectives*, John Wiley & Sons: New York, 1973, pp. 1-421.

Eilon, S., "The Role of Management Science", *The Journal of the Operational Research Society*, 31 (10), 1980, pp. 17-28.

Ettlie, John E. and Stoll, Henry W., *Managing the Design-Manufacturing Process*, New York:McGraw-Hill, 1990.

Etzioni, Amitai, "Humble Decision Making", *Harvard Business Review*, July-August 1989, pp. 122-126.

Evans, T.G., "A Program for the Solution of Geometric-Analogy Intelligence Test Questions", in M.L. Minsky, ed., *Semantic Information Processing*, Cambridge, MA:MIT Press, 1968.

Farmer, James, *Decisions, Communication, And Organization*, The RAND Corp., 1961.

Faucheux, C; Laurent, A.; and Makridakis, S., "Can We Model the Wild World or Should We First Tame It?", in *World Modeling: A Dialogue*, C.W. Churchman & R.O. Mason, eds., Amsterdam:North Holland Publishing, 1976, pp. 107-115.

Feigenbaum, E.A.; and McCorduck, P., *The Fifth Generation:Artificial Intelligence and Japan's Computer Challenge to the World*, Reading, MA:Addison-Wesley, 1983.

Feigenbaum, E.A.;and Buchanan, B.G.; and Lederberg, J., "On Generality and Problem Solving: A Case Study Using the DENDRAL Program", in B. Meltzer and D. Michie, eds., *Machine Intelligence*, Vol 6, Edinburgh: Edinburgh University Press, 1971.

Feldman, Allan, *Welfare Economics and Social Choice Theory*, Boston:Martinus Nijhoff Publishing, 1980.

Fishburn, P.C., "Foundations of Decision Analysis: Along the Way", *Management Science*, Vol 35, No 4, April 1989, pp. 387-405.

Fisher, M. and Rinnooy Kan, A.H.G., "The Design, Analysis and Implementation of Heuristics", *Management Science*, Vol 34, No 3, March 1988, pp. 263-265.

Fisher, W.D., *Clustering and Aggregation in Economics*, Baltimore:John Hopkins Press, 1969.

Fleming, A., "Time is Money: Reducing Product Development Lead Time Means Big Savings", *Automotive News*, March 23, 1992, p. 12i.

Fodor, J.A., *The Langauge of Thought*, New York:Thomas Y. Crowell, 1975.

Fodor, J.A., *Representations: Philosophical Essays on the Foundations of Cognitive Science*, Cambridge:MIT Press, 1981.

Fodor, J.A., *The Modularity of Mind*, Cambridge, MA:MIT/Bradford Press, 1983.

Forrester, Jay W., "Educational Implications of Responses to System Dynamic Models", in *World Modeling: A Dialogue*, C.W. Churchman & R.O. Mason, eds., Amsterdam:North Holland Publishing, 1976, pp. 27-35.

Foster, J.M., *List Processing*, London:Macdonald, 1967.

Foxall, G.R., *Corporate Innovation:Marketing and Strategy*, London:Croom Helm, 1984.

Frazer, J., *The Golden Bough: A Study in Magic and Religion*, 3rd edition, New York: St. Martin's Press (Originally Published in 1890), 1955.

French, Michael J., *Invention and Evolution*, Cambridge:Cambridge Univ. Press, 1988.

French, Michael J., *Conceptual Design for Engineers*, London:Pitman Press, 1985. Originally published as *Engineering Design: The Conceptual Stage*, London:Heinemann, 1971.

Frey, Donald N., "Learning the Ropes: My Life as a Product Champion", *Harvard Business Review*, Vol 69, No. 45, Sept-Oct 1991.

Fries, C.C., "The Bloomfield School", in , *Trends in European and American Linguistics 1930-1960*, C. Mohrmnn, A. Sommerfelt, and J. Whatmough, eds., Utrecht:Spectrum, 1963.

Fritsch, G.; and Hitzig, E., "Uber die electrische Erregbarkeit des Grosshirns", *Arch. Anat. Physiol. u. Wisse. Med.*, Vol 37, 1870 (from Gardner, 1987).

Fulcher, Gordon S., *Common Sense Decision Making*, Evanston:Northwestern Univ. Press, 1965.

Fung, L.W and Fu, K.S., *An Axiomatic Approach to Rational Decision-Making in a Fuzzy Environment, Fuzzy Sets and the Application to Cognitive and Decision Processes*, New York:Academic Press, 1975, pp. 227-256.

Gabele, E., "The Management of Redesigning Organisational Decision-Making Structures", in *Empirical Research on Organizational Decision-Making*, E. Witte and H.-J. Zimmermann, eds., Amsterdam:Elsevier Science Pub., 1986, pp. 65-97.

Gardner, Howard, *The Mind's New Science: A History of the Cognitive Revolution*, New York:Basic Books, 1987.

Gelernter, David, "The Metamorphosis of Information Management", *Scientific American*, Vol 261, No 2, August 1989, pp. 66-73.

Giavini, O. and Louberge, H., *The Diminishing Returns of Technology: An Essay on the Crisis of Economic Growth*, Oxford:Pergamon Press, 1978.

Gilbert, Daniel T., "How Mental Systems Believe", *American Psychologist*, February, 1991, pp. 107-119.

Gilbert, Krull, Malone, "Unbelieving the Unbelievable", *Journal of Personality & Social Psychology*, Vol. 59, No. 4, October 1990, pp. 601-613.

Gilley, O.W. and Karels, G.V., "In Search of Giffen Behavior", *Economic Inquiry*, January 1991, pp. 182-189.

Gleick, James, *Chaos: Making a New Science*, New York:Viking Penguin, 1987.

Glismann, H.H. and Horn, E., "Comparative Invention Performance of Major Industrial Countries: Patterns and Explanations", *Management Science*, Vol 34, No 10, October 1988, pp. 1169-1187.

Goldberger, A.L. and Rigney, D.R., "Sudden Death is not Chaos", *Complex Patterns in Dynamic Systems*, Keslo, Mandel, Schlesinger, eds., Teaneck (NJ):World Sceintific, 1988, pp. 23-34.

Goldman, P.S.; and Galkin, T.W., "Prenatal Removal of Frontal Association Cortex in the Fetal Rhesus Monkey: Anatomical and Fuctional Consequences in Postnatal Life", *Brain Research*, Vol 152, 1978, pp. 451-85.

Goldman-Rakic, P.S.; Isseroff, A.; Schwartz, M.L.; and Bugbee, N.M., "Neurobiology of Cognitive Development in Non-human Primates", Unpublished manuscript, Yale School of Medicine, New Haven (CT), 1982.

Goldratt, E.M. and Cox, J., *The Goal: A Process of Ongoing Improvement*, Croton-on-Hudson, NY:North River Press, 1986.

Goldratt, E.M., *It's Not Luck*, Great Barrington (MA), North River Press, 1994.

Gollub, J.P. and Benson, S.V., "Many Routes to Turbulent Convection", *Journal of Fluid Mechanics*, 100, 1980, pp. 449-70.

Gottfries, N. and Persson, T., "Empirical Examinations of the Information Sets of Economic Agents", *The Quarterly Journal of Economics*, February 1988, pp. 251-259.

Griliches, Z., Hybrid Corn: "An Exploration in the Economics of Technological Change", *Econometrica*, October, 1957.

Griliches, Z., "Research Costs and Social Returns: Hybrid Corn and Related Innovations", *Journal of Political Economy*, October, 1958, pp. 419-31.

Griliches, Z., "Hybrid Corn and the Economics of Innovation", *Science*, July 29, 1960, pp. 275-80.

Grinnell, F., *The Scientific Attitude*, 2nd ed., New York:Guilford, 1992.

Gruenwald, G., *New Product Development: What Really Works*, Chicago:Crain Books, 1985.

Gutzwiller, Martin C., "Quantum Chaos", *Scientific American*, Vol 266, No 1, January 1992, pp. 78-84.

Hacking, I., "Wiggenstein the Psychologist?", *New York Review of Books*, April 1, 1982.

Hagerstrand, T., *Innovation Diffusion as a Spatial Process*, Chicago:University of Chicago Press, 1967.

Hagerstrand, T., "On Socio-technical Ecology and the Study of Innovation", *Ethnologica Europaea*, Vol 7, 1974, pp. 17-34.

Haken, H., "Evolution of Order and Chaos in Physics, Chemistry, and Biology", *Springer Series in Synergetics*, Vol 17, 1982.

Haken, H., *Advanced Synergetics: Instability Hierarchies of Self-organizing Systems*, New York:Springer-Verlag, 1983.

Harris, M, *The Rise of Anthropological Theory: A History of Theories of Culture*, New York:Thomas Y. Crowell, 1968.

Harris, Z.S., "Discourse Analysis", *Language*, Vol 28, 1952, pp. 18-30.

Harth, E., *Windows on the Mind: Reflections on the Physical Basis of Consciousness*, New York:William Morrow, 1982.

Hartman, S.J.; Lundberg, O.; and White, Michael, "Effect of Background and Organizational Position on Executive Planning", *Journal of Social Psychology*, Vol 130, No 6, December 1990, pp. 801-811.

Hauschildt, J., "Goals and Problem-Solving in Innovative Decisions", in *Empirical Research on Organizational Decision-Making*, (E. Witte and H.-J. Zimmermann, eds.), Amsterdam:Elsevier Science Pub., 1986, pp. 3-19.

Hawking, S.W., *A Brief History of Time*, New York:Bantam Books, 1988.

Hayles, N.K., *Chaos Bound: Orderly Disorder in Contemporary Literature and Science*, Ithaca:Cornell University Press, 1990.

Hayles, N.K., *Chaos and Order: Complex Dynamics in Literature and Science*, Chicago:University of Chicago Press, 1991.

Hebb, D.O., *Organization of Behavior*, New York:John Wiley, 1949.

Heller, Frank, Drenth, Pieter, Koopman, Paul, and Rus, Veljko, *Decisions in Organizations*, London:SAGE Pub., 1988.

Helmer, Olaf, "Interdisciplinary Modeling", in *World Modeling: A Dialogue*, C.W. Churchman & R.O. Mason, eds., Amsterdam:North Holland Publishing, 1976, pp. 73-80.

Herskovitz, M.J., Franz Boas: *The Science of Man in the Making*, Cambridge, Mass:Harvard University Press, 1953.

Hertz, D. and Rubenstein, A.H., *Team Research*, Cambridge (MA):Eastern Technical Pub., 1953.

Heslot, F.; Castaing, B.; Libehaber, A., "Transitions to Turbulance in Helium Gas", *Physical Review A.*, 43, 1987, pp. 5870-73.

Hickson, David J., Butler, Richard J., Cray, David, Mallory, Geoffrey R., and Wilson, David C., *Top Decisions: Strategic Decision-Making in Organizations*, San Francisco:Jossey-Bass, 1968.

Himmelblau, David M., ed., *Decomposition of Large Scale Problems*, Amsterdam:North-Holland, 1973.

Hirshleifer, Jack, *Price Theory and Applications*, Englewood Cliffs, NJ:Prentice Hall, 1988.

Hockett, C., *A Course in Modern Liguistics*, New York:Macmillian, 1958.

Hockett, C., *The State of the Art*, The Hague: Mouton, 1968.

Holloway, Charles A., *Decision Making Under Uncertainty*, Englewood Cliffs (NJ):Prentice-Hall:, 1979.

Hooper, J., "Interview with Karl Pribram", *Omni*, October 1982, pp. 129-35,169-76.

Howard, Ronald, "Decision Analysis: Practice and Promise", *Management Science*, Vol 34, No 6, June 1988, pp. 679-695.

Hubel, D.H.; and Wiesel, T.N., "Receptive Fields, Binocular Interaction and Fuctional Architecture in the Cat's Visual Cortex", *Journal of Physiology*, Vol 160, 1962, pp. 106-54.

Hunt, Morton, *The Story of Psychology*, New York:Doubleday, 1993.

InfoWorld (anonymous), "The Bigger They Are, The Harder They Fall", *InfoWorld*, Feb 6, 1995, p. 62.

Jackson, J.Hughlings, *Selected Writings*, London:Hodder & Stoughton, 1932.

Jackson, K., "U.S. Project Aims to Chop New-Car Development Time", *Automotive News*, May 31, 1993., pp. 6, 35.

Jaffe, Adam B., "Technological Opportunity and Spillovers of R&D: Evidence from Firms' Patents, Profits, and Market Value", *American Economic Review*, Vol. 76, No. 5, December 1986, pp. 984-1001.

James, William, *Psychology*, Cleveland:World, 1948.

Janssen, J. (ed.), *Semi-Markov Models:Theory and Applications, Proceedings of the International Symposium on Semi-Markov Processes and Their Applications*, New York:Plenum, 1986.

Jensen, K.S.; Mosekilde, E.; Holsern-Rathlow, N.H., "Self-Sustained Oscillations and Chaotic Behavior in Kidney Pressure Regulation", *Laws of Nature and Human Conduct*, I. Prigogine and M. Sanglier, eds, Brussles, 1986, pp. 91-109.

Jensen, R., "Information Cost and Innovation Adoption Policies", *Management Science*, Vol 34, No 2, February 1988, pp. 230-239.

Johne, F.A., *Industrial Product Innovation: Organisation and Management*, New York:Nichols Printing, 1985.

Jones, A.J., *Game Theory: Mathematical Models of Conflict*, Chischester (UK):Ellis Horwood Ltd., 1980.

Jones, P.N. and North, J., "Japanese Motor Industry Transplants: The West European Dimension", *Economic Geography*, Vol 67, no 2., April 1991, pp. 105-123.

Jornsten, K.O.; Larson, T.;Lundgren, A.; and Migdalas, A., "An Entropy Model with Variable Target", *Environment and Planning*, Vol 22, No 4, April, 1990, pp. 493-506.

Kamien, Morton and Li, Lode, "Subcontracting, Coordination, Flexibility, and Production Smmothing in Aggregate Planning", *Management Science*, Vol 36, No 11, November 1990, pp. 1352- 1363.

Kamien, Morton I. and Schwartz, Nancy L., *Market Structure and Innovation*, Cambridge University Press, 1982., pp. 1-33, 58-79, 105-22, 176-82.

Kandel, E.R., "Small Systems of Neurons", *Scientific American*, Vol 241, 1979, pp. 66-84.

Katz, J. and Fodor,J., "The Structure of a Semantic Theory", *Language*, Vol 39, 1963, pp. 170-210.

Kaufmann, Arnold, *The Science of Decision-Making*, New York:McGraw Hill, 1968.

Kaye, Brian H., *A Random Walk Through Fractal Dimensions*, Weinheim (Germany):VCH, 1989.

Khan, A.M. and Manopichetwattana, V., "Innovative and Noninnovative Small Firms: Types and Characteristics", *Management Science*, Vol 35, No 5, May 1989, pp. 597-606.

Kiefer, N.M. and Nyarko, Y., "Optimal Control of an Unknown Linear Process with Learning", *International Economic Review*, 1989, pp. 571-586.

Kifer, Yuri, *Random Pertubations of Dynamical Systems*, Boston:Birkhauser, 1988.

Kleinmuntz, Benjamin, "Why We Still Use Our Heads Instead of Formulas: Toward an Integrative Approach", *Psychological Bulletin*, vol 107, no 3, May 1990, pp. 296-310.

Knapp, C.M., "Factors Influencing Formal and Informal Key Commnicator Effectiveness in a Multidivisional Industrial Corporation", PhD Dissertation, Northwestern University, 1984.

Konishi, M., "Experimental Studies in the Ontogeny of Avian Vocalizations", in R.A. Hinde, ed., *Bird Vocalization*, Cambridge:Cambridge University Press, 1969.

Kroeber, A., "The Super-Organic", *American Anthropologist*, 17, 1917, pp. 163-213.

Kroeber, A., *Anthropology*, New York:Harcourt, Brace, 1948.

Kuhn, T, *The Structure of Scientific Revolutions*, Chicago:University of Chicago Press, 1962.

Lakoff, G.;and Ross, J.R., "Is Deep Structure Necessary?", in J.McCawley, ed., *Syntax and Semantics*, Vol 7, New York:Academic Press, 1976.

Langdon, R. and Rothwell, R., *Design and Innovation: Policy and Management*, New York:St. Martin's Press, 1985.

Lashley, K.S., *Brain Mechanisms and Intelligence*, Chicago:University of Chicago Press, 1929.

Lashley, K.S.; Chow, K.L.;and Semmes, J., "An Examination of the Electrical Field Theory of Cerebral Integration", *Psychological Review*, Vol 58, 1951, pp. 123-36.

Lautenborn, W. and Cramer, E., "Subharmanic Route to Chaos Observed in Acoustics", *Physical Review Letters*, 47, 1981, pp. 1445-1449.

Layton, C.; Harlow, C.; De Hoghton, G., *Ten Innovations: An International Study on Technological Development and the Use of Qualified Scientists and Engineers in Ten Industries*, London:George Allen & Unwin, 1972.

Lees, R.B., "Review of Noam Chomsky's Syntactic Structures", *Language*, Vol 33, 1957, pp. 375-408.

Lehner, Paul, "Cognitive Factors in User/Expert system Interaction", *Human Factors*, February 1987, pp. 97-109.

Lenat, D.B., "A.M.: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search", *Report STAN-CS-76-570*, Stanford University, Computer Science Department, 1976.

Lenat, D.B., "Computer Software for Intelligent Systems", *Scientific American*, Vol 251, 1984, pp. 204-13.

Leonard-Barton, D. and Deschamps, I., "Managerial Influence in the Implementation of New Technology", *Management Science*, Vol 34, No 10, October 1988, pp. 1252-1265.

Levi-Strauss, C., *Structural Anthropology*, translated by C. Jacobson & B. Grundfest, New York:Basic, 1963.

Levi-Strauss, C., *Tristes Tropigues*, Translated by J. Russell, New York:Antheneum, 1964.

Levi-Strauss, C., *The Raw and the Cooked*, translated by J.& D. Weightman, New York:Harper & Row, 1969.

Levitt, T., "Decisions", *Harvard Business Review*, vol 67, Jan-Feb 1989, p. 6.

Levy, David M. and Nestor E. Terleckyj , "Effects of Government R&D on Private R&D Investment and Productivity: A Macroeconomic Analysis", *Bell Journal of Economics*, Vol. 14, No. 2, Autumn 1983, pp. 551-61.

Lewis, William; and Samuel, Andrew, *Fundamentals of Engineering Design*, Sydney:Prentice Hall, 1989.

Lichtenberg, A.J. and Lieberman, M.A., *Regular and Stochastic Motion*, New York:Springer, 1983.

Lighthill, J., "A Report on Artificial Intelligence", Unpublished manuscript, Science Research Council, 1972.

Little, J.D., "Research Opportunities in the Decision and Managment Sciences", *Management Science*, Vol. 32 #1, January 1986, pp. 1-13.

Littler, D.A.; Sweeting, R.C., "Innovative Business Development", *Futures*, April 1987.

Longuet-Higgins, H.C., "The Perception of Music", *Proceedings of the Royal Society of London series B 205*, 1979, pp. 307-22.

Longuet-Higgins, H.C., "Artificial Intelligence-A New Theoretical Psychology?", *Cognition*, Vol 10, 1981, pp. 197-200.

Lorenz, C., *The Design Dimension: A New Competitive Weapon for Business*, New York:Basil Blackwell, 1986.

Lorenz, E.N., "Deterministic non-periodic Flow", *Journal of Atmospheric Sciences*, 20, 1963, pp. 130-141.

Lorenz, H.W., "International Trade and the Possible Occurance of Chaos", *Economic Letters*, 23, 1987, pp. 135-38.

Luria, A.R., *Higher Cortical Fuctions in Man*, New York: Basic Books, 1966.

Malinowski, B., "Culture", *International Encyclopedia of Social Sciences*, D.L. Sills, ed., New York:Macmillan (Originally published in 1935), 1968.

Mandler, G., "What is Cognitive Psychology? What Isn't?", Address to the APA Division of Philosophical Psychology, Los Angeles, 1981.

Mansfield, E., "Technical Change and the Rate of Imitation", *Econometrica*, Vol 29., 1961, pp. 741-66.

Mansfield, E., *Industrial Research and Technical Innovation*, New York:Norton, 1968.

Mansfield, E., *The Economics of Technological Change*, New York:Norton, 1968.

Mansfield, E., "Public Policy Toward Civilian Technology", *Microeconomics: Selected Readings*, 3rd edition, W.W. Norton & Co.: New York, 1979, pp. 579-592.

Mansfield, E., *Managerial Economics and Operations Research*, New York:Norton, 1980.

Mansfield, E., "The Speed and Cost of Industrial Innovation in Japan and the United States: External vs. Internal Technology", *Management Science*, Vol 34, No 10, October 1988, pp. 1157-1168.

March, J.G. and Simon, H.A., *Organizations*, New York:Wiley, 1958.

Marcus, M., "A Theory of Syntactic Recognition for Natural Language", in *Artificial Intelligence: An MIT Perspective*, P.H. Winston and R.H. Brown, eds., Vol 1, Cambridge, MA:MIT Press, 1979, pp. 191-229.

Marr, D., *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*, San Francisco:W.H.Freeman, 1982.

McAllister, P., "Rational Behavior and Rational Expectations", *Journal of Economic Theory*, Vol. 52, 1990, pp. 332-363.

McCarthy, J.; Abrahams, P.W.; Edwards, D.J.; Hart, T.D.;and Levin, M.I., *LISP I.5 Programmer's Manual*, Cambridge, MA:MIT Press, 1962.

McCarthy, J.;and Hayes, P.J., "Some Philosophical Problems from the Standpoint of Artificial Intelligence", in B. Meltzer and D. Michie, eds., *Machine Intelligence 4*, Edinburgh:Edinburgh University Press, 1969.

McVoy, E.C., "Patterns of Diffusion in the United States", *American Sociological Review*, vol 5, 1940, pp 219-27.

Merrifield, D.B., *Strategic, Analysis, Selection and Management of R&D Projects*, New York:AMACOM, 1977.

Meyers, P.W. and Wilemon, D., "Learning in New Technology Development Teams", *Journal of Product Innovation Management*, No 6, 1989, pp. 79-88.

Mikhailov, A.S. and Loskutov, A.Y., *Foundations of Synergetics II: Complex Patterns*, Berlin:Springer-Verlag, 1991.

Miller, G.A., "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processsing Information", *Psychological Review*, Vol 63, 1956, pp. 81-97.

Miller, G.A., "Some Psychological Studies of Grammer", *American Psychologist*, Vol 17, 1962, pp. 748-762.

Miller, G.A., "A Very Personal History", Talk to Cognitive Science Workshop , MIT, Cambridge, MA., June 1, 1979, (as accounted by Gardner (1987)).

Mindlin. G.B.; Hou, X.; Solari, H.G.; and Tufillaro, N.B., "Classification of Strange Attractors by Integers", *Physical Review Letters*, Vol. 64, No. 20, May 14, 1990, pp. 2350-53.

Minsky, M, "The Society Theory", in P.H.Winston and R.H. Brown, eds., *Artificial Intelligence: An MIT Perpective, Vol 1*, Cambridge, MA:MIT Press, 1979, pp.423-50.

Minsky, M., "Steps toward Artificial Intelligence", in E.A Feigenbaum and J. Feldman, eds., *Computers and Thought*, New York:McGraw-Hill, 1963.

Minsky, M., "A Framework for Representing Knowledge", in P.H. Winston, ed., *The Psychology of Computer Vision*, New York:McGraw-Hill, 1975.

Minsky, M., "Learning Meaning", Unpublished Manuscript, MIT, 1982 (accounted by Gardner (1987)).

Minsky, M.;and Papert, S., *Perceptrons*, Cambridge, MA:MIT Press, 1968.

Mitchell, Russel, "Masters of Innovation: How 3M Keeps its New Products Coming", *Business Week*, April 10, 1989, pp. 58-63.

Monastersky, Richard, "Forecasting into Chaos", *Science News*, Vol 137, No 18, May 5, 1990, pp. 280-282.

Monastersky, Richard, "Shaking Up Seismic Theory", *Science News*, Vol. 141, No. 9, February 29, 1992, pp. 136-137.

Moore, C., "Unpredictability and Undecidability in Dynamical Systems", *Physical Review Letters*, Vol. 64, No. 20, May 14, 1990, pp. 2354-2357.

Moore, J.W.; Jensen, Brian; and Hauck, W.E., "Decision-Making Processes of Youth", *Adolescence*, Vol XXV, No 99, Fall 1990, pp. 583-592.

Mosekilde, E.; Larsen, E.; and Sterman, J., "Coping with Complexity: Deterministic Chaos in Human Decisionmaking Behavior", in *Beyond Belief*, J.L. Casti and Karqvist, A., eds., Boca Raton (FL):CRC Press, 1991, pp. 199-229.

Murray, Michael, *Decisions: A Comparative Critique*, Marshfield (MA):Pitman, 1986.

Naj, Amal Kumar, "In R&D, the Next Best Thing to a Gut Feeling", *Wall Street Journal*, May 21, 1990.

National Academy of Engineering, *Time Horizons and Technology Investments*, Washington, D.C.:National Academy Press, 1992.

National Academy of Sciences, *Technology & Economics*, National Academy Press: Washington DC, 1991.

National Research Council, *Improving Engineering Design: Designing for Competitive Advantage*, Washington, D.C.:National Academy Press, 1991.

Newell, A., "Intellectual Issues in the History of Artificial Intelligence", in F.Machlup and U. Mansfield, eds., *The Study of Information: Interdisciplinary Messages*, New York:John Wiley, 1983.

Newell, A. and Simon, H.A., *Human Problem Solving*, Englewood Cliffs, N.J.:Prentice-Hall, 1972.

Newell, A.; Shaw, J.C.;and Simon, H.A., "Elements of a Theory of Human Problem Solving", in R.J.C. Harper, C.C. Anderson, C.M.Christensen, and S.M. Hunka, eds., *The Cognitive Processes: Readings*, Englewood Cliffs, N.J.:Prentice-Hall, 1964.

Newmeyer, F, Linguistic *Theory in America: The First Quarter Century of Transformational Generative Grammar*, New York: Academic Press, 1980.

O'Boyle T.F., "GE Refrigerator Woes Illustrate the Hazards in Changing a Product", *Wall Street Journal*, pg.1, May 7, 1990.

Olsen, L.F. and Degn, H., "Chaos in Enzyme Reaction", *Nature*, 267, 1977, pp. 177-8.

Oosterhaven, Jan, "The Supply Driven Input/Output Model: A New Interpretation but still Implausible", *Journal of Regional Science*, Vol. 29, #3, 1989, pp.459-465.

Ott, E.; Grebogi, C.; Yorke, J.A., "Controlling Chaos", *Physical Review Letters*, Vol. 64, No. 11, March 12, 1990, pp. 1196-99.

Ottino, Julio M., "The Mixing of Fluids", *Scientific American*, Vol 260, No 1, January 1989, pp. 56-67.

Payne, Stanley L., *The Art of Asking Questions*, Princeton:Princeton University Press, 1951.

Pecora, D. and Zumsteg, J.R., "An Application of Expert Systems to Composite Structural Design and Analysis", *Applications of Knowledge-Based Systems to Engineering Analysis and Design*, November 17-22, 1985, pp. 135-147.

Peery, N.S., Jr., "A Structural Theory of Political Behavior Within Organizations", Ph.D. Thesis, University of Washington, Ann Arbor, 1974.

Pelton, Warren J., Sackman, Sonja, and Boguslaw, Robert, *Tough Choices: The Decision-Making Styles of America's Top 50 CEO's*, Homewood (IL):Dow Jones-Irwin, 1990.

Pemberton, H.E., "The Curve of Culture Diffusion Rate", *American Sociological Review*, Vol 1, 1936, pp. 547-56.

Pemberton, H.E., "Culture Diffusion Gradients", *American Journal of Sociology*, Vol 42, 1937, pp. 226-33.

Pemberton, H.E., "The Spatial Order of Cultural Diffusion", *Sociology and Social Research*, vol 22, 1938, pp. 246-51.

Peters, T.J. and Waterman, R.H,, *In Search of Excellence: Lessons from America's Best Run Companies*, New York:Harper & Row, 1982.

Peterson, I., "Unknotting a Tangled Tale", *Science News*, Vol 133, No 21, May 21, 1988, pp. 328-330.

Peterson, I., "Knot Physics", *Science News*, Vol 135, No. 11, March 18, 1989, p. 174.

Peterson, I., "Recipes for Artificial Realities", *Science News*, Vol 138, No 21, November 24, 1990, pp. 328-329.

Peterson, I., "Back to the Quantum Future", *Science News*, Vol 140, No 18, November 2, 1991, pp. 282-284.

Peterson, I., "Step in Time: Exploring the Mathematics of Synchronously Flashing Fireflies", *Science News*, Vol 140, No 9, August 31, 1991, pp. 136-137.

Peterson, I., "Achieving Control of Chaotic Laser Output", *Science News*, Vol 141, No 8, February 22, 1992, p. 119.

Peterson, I., "Chaos in the Clockwork: Uncovering Traces of Chaos in Planetary Orbits", *Science News*, Vol 141, No 8, February 22, 1992, pp. 120-121.

Peterson, I., "Looking-Glass World", *Science News*, Vol 141, No 1, January 4, 1992, pp. 8-10,15.

Phillips, Lawrence D., "Generation Theory", *Research in Marketing*, Supplement 1, 1982, pp. 113-139.

Poensgen, O.H. and Hort, H., "Situative Influences on Corporate Planning", in *Empirical Research on Organizational Decision-Making*, E. Witte and H.-J. Zimmermann, eds., Elsevier Science Pub.:Amsterdam, 1986, pp. 269-299.

Porter, M.E., *Competitive Advantage: Creating and Sustaining Superior Performance*, New York: Free Press, 1985.

Power, C; Kerwin, K.; Grover, R.; Alexander, K.; Hof, R.D., "Flops: Too Many New Products Fail", *Business Week*, August 16, 1993, pp. 76-82.

Pribram, K.H., *Languages of the Brain: Experimental Paradoxes and Principles in Neuropsychology*, Englewood Cliffs, N.J.:Prentice-Hall, 1971.

Quillian, M.R., "Semantic Memory", in M. Minsky, ed., *Semantic Information Processing*, Cambridge, MA:MIT Press, 1968.

Radcliff-Brown, A.R., "On Social Structure", *Journal of the Royal Anthropological Institute*, 70, 1940.

Radcliff-Brown, A.R., *Structure and Function in Primitive Society*, Glencoe, IL:Free Press, 1952.

Ravinder, H.V.; Kleinmuntz, D.N.; Dyer, J.S., "The Reliability of Subjective Probabilities Obtained Through Decomposition", *Management Science*, Vol 34, No 2, February 1988, pp. 186-199.

Reddy, D.R.; Erman, L.D.; Fennell, F.D.;and Neely, R.B., "The HEARSAY Speech Understanding System", in *Proceedings, Third International Joint Conference on Artificial Intelligence*, Vol 3, 1973, pp. 185-193.

Rivers, W.H.R., "A Geneological Model of Collecting Social and Vital Statistics", *Journal of the Anthropological Institute of Great Britain and Ireland*, No. 30, 1900, pp. 74-82.

Roberts, L.G., "Machine Perception of Three Dimensional Solids", in T.T. Tippett et al., *Optical and Electro-optical Information Processing*, Cambridge (MA):MIT Press, 1965.

Roberts, R.M., *Serendipity: Accidental Discoveries in Science*, New York:John Wiley, 1989.

Robins, R.H., *A Short History of Linguistics*, Bloomington:Indiana University Press, 1967.

Robinson, W.T., "Product Innovation and Start-Up Business Market Share Performance", *Management Science*, Vol 36, No 10, October 1990, pp. 1279-1289.

Rockwell, J.R. and Particelli, M. C., "New Product Strategy; How the Pros Do It", *Industrial Marketing*, Pages 49-60, May, 1982.

Rogers, E.M. and Shoemaker, F.F., *Communication of Information: A Cross-Cultural Approach*, New York:Free Press, 1971.

Rosenau, Milton D., Jr., *Faster New Product Development*, New York:AMACOM, 1990.

Rosenberg, Nathan, *The Economics of Technical Change*, Middlesex (UK):Penguin, 1971.

Rosenbloom, P.S. and Newell, A., "The Chunking of Goal Hierarchies: A Generalized Model of Practice", *Proceedings of the International Machine Learning Workshop*, 1983.

Rosser, J.B., "Chaos Theory and the New Keynesian Economics", *The Manchester School*, LVIII, No. 3, Sept. 1990, pp. 265-291.

Rothwell, R. and Zegveld, W., *Innovation and the Small and Medium Sized Firm*, Boston:Kluwer-Nijhoff, 1982.

Routh, G., "Economics and Chaos", *Challenge*, July-August, 1989, pp. 47-52.

Rubenstein, A.H., "Liaison Relations in R&D", *IRE Transactions on Engineering Management*, EM-4(2), June 1957, pp. 72-78.

Rubenstein, A.H., "The Role of Imbedded Technology and the R&D/Innovation Process", Joint Economic Commitee, Congress of the United States, *Special Study on Economic Change, Vol.3, Research and Innovation: Developing a Dynamic Nation*, December 29, 1980.

Rubenstein, A.H., "Trends in Technology Management", *IEEE Transactions on Engineering Management*, Nov 1985.

Rubenstein, A.H., *Managing Technology in the Decentralized Firm*, John Wiley & Sons, 1989.

Rubenstein, A.H. and Ettlie, J., "Innovation Among Suppliers to Automobile Industry: An Exploratory Study of Bearers and Facilitator", *R&D Management*, 9(2), February 1979.

Rubenstein, A.H. and Geisler, E., "The Use of Indicators and Measures of the R&D Process in Evaluating Science and Technology Programs", in Roessner (ed.), *Government Policies for Industrial Innovation: Design, Implementation, Evaluation*, New York:St.Martin's Press, 1988, pp. 185-203..

Rubenstein, A.H.; Beal, W.; Kegan, D.l.; Moore, E.; Moore, W.C.; Rath, G.; Thompson, C.W.; Trusewell, R.; Werner, D.J., "Exploration on the Information Seeking Behavior of Researchers", in *Communication among Scientists and Engineers* (C.E., Nelson and D.K. Pollock, eds.), Lexington (MA):Heath Lexington Books, 1970.

Rubinstein, Gwen, "Impresario of Risk", *Association Management*, February 1987, pp. 30-34.

Ruelle, D., *Chaotic Evolution and Strange Attractors: The Statistical Analysis of Time Series for Deterministic Non-Linear Systems*, Cambridge:Cambridge University Press, 1989.

Russmussen, D.R. and Mosekilde, E., "Bifurcations and Chaos in a Generic Management Model", *European Journal of Operational Research*, 35, 1988, pp. 80-88.

Ruttan, V., "Usher and Schumpeter on Invention, Innovation and Technological Change", *Quarterly Journal of Economics*, November 1959, pp. 596-606.

Ryan, B. and Gross, N.C., "The Diffusion of Hybrid Seed Corn in Two Iowa Communities", *Rural Society*, vol 8, 1943, pp. 15-24.

Rychener, Michael D., *Expert Systems for Engineering Design*, San Diego:Academic Press, 1988.

Ryle, G., *The Concept of Mind*, London:Hutchinson, 1949.

Saaty, Thomas L., "An Exposition of the AHP in Reply to the Paper 'Remarks on the Analytic Hierarchy Process'", *Management Science*, Vol 36, No 3, March 1990, pp. 259-273.

Samet, Dov, "Ignoring Ignorance and Agreeing to Disagree", *Journal of Economic Theory*, Vol 52, No 1, October 1990, pp. 190-207.

Sapir, E., *Language*, New York:Harcourt, Brace, & World, 1921.

Saussure, F. de, *Course in General Linguistics*, New York:Philosophical Library, 1959.

Schank, R.C., "Conceptual Dependency: A Theory of Natural Language Understanding", *Cognitive Psychology*, Vol 3, 1972, pp. 552-631.

Schank, R.C.;and Abelson, R., *Scripts, Plans, Goals, and Understanding*, Hillsdale, N.J.:Lawrence Erlbaum, 1977.

Schon, Donald A., *Beyond the Stable State*, London:Temple Smith, 1971.

Schroeder, Manfred, *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*, New York:W.H. Freeman, 1991.

Schumpeter, J., "The Explanation of the Business Cycle", *Economica*, No. 21, 1927.

Schumpeter, J., "The Instability of Capitalism", *Economic Journal*, 1928, pp. 361-86.

Schumpeter, J., *The Theory of Economic Development*, (Translation by John E. Elliot), New Jersey:Transactions Books, 1934.

Schumpeter, J., *The Theory of Economic Development*, Cambridge (MA):Harvard University Press, 1934.

Schumpeter, J., *Business Cycles*, McGraw-Hill Book Co.:New York, 1939.

Schumpeter, J., *Capitalism, Socialism, and Democracy*, (5th ed.) Allen and Unwin:London, 1976.

Schuster, H.G., *Deterministic Chaos: An Introduction*, Deerfield Beach (FL):Physik-Verlag, 1984.

Schwartz, H.S., *Narcissistic Process and Corporate Decay: The Theory of the Orgnization Ideal*, New York: New York University Press, 1990.

Scott, W.G., "The Theory of Significant People", *Public Administration Review*, 33, 1973, pp. 308-313.

Searle, J., "Minds, Brains, and Programs", *The Behaviorial and Brain Sciences*, Vol 3, 1980, pp. 417-57.

Searle, J., *Intentionality: An Essay in the Philosophy of Mind*, Cambridge: Cambridge University Press, 1983.

Shannon, C.E., "A Symbolic Analysis of Relay and Switching Circuits", *Transactions of the American Institute of Electrical Engineers*, Vol57, 1938, pp. 1-11.

Shannon, C.E., *The Mathematical Theory of Communication*, Urbana:University of Illinois Press, 1949.

Sheth, J.N.; Ram, S., *Bringing Innovation to Market*, New York:Wiley, 1987.

Shinbrot, T.; Ott, E.; Grebogi,C.; Yorke, J.A., "Using Chaos to Direct Trajectories to Targets", *Physical Review Letters*, Vol. 65, No. 26, December 24, 1990, pp. 3215-18.

Shuman, J.B. and Roseneau, D., *The Kondratieff Wave*, New York:World Publishing, 1972.

Shurkin, Joel N., *Engines of the Mind; An History of the Computer*, New York:Norton, 1984.

Singley, K. and Anderson, J.R., *The Transfer of Cognitive Skill*, Cambridge, Mass: Harvard University Press, 1989.

Skinner, B.F., *Verbal Behavior*, New York:Appleton-Century-Crofts, 1957.

Smith, P. and Reinertsen, D. *Developing Products in Half the Time*, New York:Van Norstrand, Reinhold, 1991.

Sperry, R.W., "Mechanisms of Neutral Maturation", in S.S. Stevens, ed., *Handbook of Experimental Psychology*, New York:John Wiley, 1951.

Sperry, R.W.;and Miner, N., "Pattern Perception Following Insertion of Mica Plates into Visual Cortex", *Journal of Comparitive and Physiological Psychology*, Vol 48, 1955, pp. 463-69.

Spinoza, Benedictus de, *The Collected Works of Spinoza* (edited and translated by Edwin Curly), Princeton:Princeton University Press, 1985.

Stahl, Michael and Zimmerer, Thomas, "Modeling Strategic Acquisition Policies: A Simulation of Executives' Acquisition Decisions", *Academy of Management Journal*, Vol. 27, June 1984, pp. 369-383.

Stanford Design Forum, *Design in the Contemporary World: A Paper based on the Proceedings of the 1988 Stanford Design Forum*, Stanford:Pentagram Design, 1989.

Stewart, I., *Does God Play Dice? The Mathematics of Chaos*, Cambridge (MA):Penguin, 1991.

Stinchcombe, Arthur L., *Information and Organizations*, Berkely:University of California Press, 1990.

Stocking, G.W., *Race, Culture, and Evolution: Essays in the History of Anthropology*, Chicago:University of Chicago Press, 1982.

Strassmann, P.A., *Information Payoff: The Transformation of Work in the Electronic Age*, New York, Free Press, 1985.

Suh, Nam P., *The Principles of Design*, New York:Oxford University Press, 1990.

Sussman, G.J., *A Computer Model of Skill Acquisition*, New York:American Elsevier, 1975.

Swinney, H.L. and Gollub, J.P., "Characterization of Hydrodynamic Strange Attractors", *Physica D*, 18, 1986, pp. 448-54.

Thiel, Stuart, E., "Some Evidence on the Winner's Curse", *American Economic Review* Vol 78, No 5, December 1988, pp. 884-895.

Thompson, R.F., *The Brain: An Introduction to Neuroscience*, New York:W.H. Freeman, 1985.

Townsend, R., *Up the Organization*, New York:Knopf, 1970.

Townsend, R., *Further Up the Organization*, New York:Knopf, 1984.

Travis, John, "Electronic Ecosystem", *Science News*, Vol 140, No 6, August 10, 1991, pp. 88-90.

Trevino, Linda, "Ethical Decision Making in Organizations: A Person -Situation Interactionist Model", *Academy of Management Review*, Vol 11 No. 3, 1986, pp. 601-617.

Tufillaro, N.B.; Abbot, T.; Reilly, J., *An Experimental Approach to Non-Linear Dynamics and Chaos*, Redwood City, CA:Addison-Wesley, 1992.

Turing, A.M., "On Computable Numbers, with an Application to the Entsceidungs-Problem", *Proceedings of the London Mathematical Society*, Series 2, Vol 42, 1936, pp. 230-65.

Tversky, A.; Slovic, P.; Kahneman, D., "The Causes of Preference Reversal", *The American Economic Review*, Vol. 80, #1, March 1990, pp. 204-217.

Twiss, B.C., *Managing Technical Innovation*, London:Longman, 1974.

Tylor, E., *Primitive Culture*, London:John Murray, 1871.

Ulrich, Alvin, "Public and Private Returns from Joint Venture Research: An Example from Agriculture", *Quarterly Journal of Economics*, Vol. 101, No. 1, February 1986, pp. 103-29.

Urban, G.L. and Von Hipple, E., "Lead User Analysis for the Development of New Industrial Products", *Management Science*, Vol 34, No 5, May 1988, pp. 569-582.

Usher, A.P., "Historical Implications of the Theory of Economic Development", *Review of Economic Statistics*, vol. 33, 1951, pp. 158-162.

Usher, A.P., *An History of Mechanical Inventions*, Cambridge (MA):Harvard Univ. Press, 1954.

Usher, A.P., "Technical Change and Capital Formation", *Capital Formation and Economic Growth*, National Bureau of Economic Research, 1955, pp. 523-50.

Utterback, J.M., "The Dynamics of Product and Process Innovation in Industry", in Hill, C.T. & Utterback, J.M. *Technical Innovation for a Dynamic Economy*, New York:Pergamon, 1979, pp. 40-65.

Utterback, J.M. & Abernathy, W.J., "A Dynamic Model of Process and Product Innovation", *Omega 3*,6: pg 639-656, 1975.

Van der Duyn Schouten, F.A., *Markov Decision Drift Processes, in Semi-Markov Models:Theory and Applications*, New York:Plenum, 1986, pp. 63-78.

Van Der Duyn Schouten, F.A., *Markov Decision Processes with Continuous Time Parameter*, Amsterdam:Mathematisch Centrum, 1983.

Van der Ploeg, F., "Rational Expectations, Risk and Chaos in Financial Markets", *Economic Journal*, 96, 1985, pp. 151-162.

Van Lehn, K.; Seely Brown, J.;and Greeno, J., *Competitive Argumentation in Computational Theories of Cognition*, Cognitive and Instructional Science Series, Xerox Parc, Palo Alto, 1982.

Veneris, Y., "Modeling the Transition from the Industrial to the Information Revolution", *Environment and Planning*, Vol 22, No 3, March 1990, pp. 399-416.

von Neumann, J. and Morgenstern, O., *Theory of Games and Economic Behavior*, Princeton:Princeton University Press, 1944.

Vonk, Roland, *Prototyping: The Effective Use of CASE Technology*, New York:Prentice Hall, 1990.

Wallace, A.F.C. & Atkins, J., "The Meaning of Kinship Terms", *American Anthropologist* 62, 1960, pp 58-80.

Waltz, D., "Understanding Line Drawings of Scenes with Shadows", in P.H. Winston, ed., *The Psychology of Computer Vision*, New York:McGraw-Hill, 1975.

Warner, A.W.; Morse, D.; and Cooney, T.E., *The Environment of Change*, New York, Columbia University Press, 1969.

Watson, Stephan R. and Buede, Dennis M., *Decision Synthesis: The Principles and Practice of Decision Analysis*, Cambridge:Cambridge University Press, 1987.

Weintraub, E.R. (ed.), *Toward a History of Game Theory*, Durham:Duke University Press, 1992.

Weiss, P., "Central versus Peripheral Factors in the Development of Coordination", *Research Publications of the Association for Research in Nervous and Mental Disease*, Vol 30, 1952, pp. 3-23.

Weiss, R., "A Flight of Fancy Mathematics", *Science News*, Vol 137, No 11, March 17, 1990, p. 172.

Weiss, W.H., *Decision Making for First-Time Managers*, New York:AMACOM, 1985.

Weizenbaum, J., *Computer Power and Human Reason*, San Francisco:W.H.Freeman, 1976.

Weizenbaum, J., "ELIZA- A Computer Program for the Study of Natural Language Communication between Man and Machine", *Communications of the Association for Computing Machinery*, Vol 9, 1966, pp. 36-45.

Wexler, K., "A review of John Anderson, Language, Memory, and Thought", *Cognition*, Vol 6, 1978, pp.327-51.

Wheelwright, S.C. and Clark, K.B., *Revolutionizing Product Development*, New York:Free Press, 1992.

White, L.A., "The Ethnography and Ethnology of Franz Boas", *Memorial Museum Bulletin*, No. 6 (Austin, TX),1963.

Wiggenstein, L., *Tractatus Logico-Philosophicus*, Translated by Pears and McGuinnes, London:Routledge & Kegan Paul (original work in 1921),1961.

Wilensky, R., *Planning and Understanding: A Computational Approach to Human Reasoning*, Reading (MA):Addison-Wesley,1983.

Wilson, John and Rutherford, Andrew, "Mental Models: Theory & Application in Human Factors", *Human Factors*, December 1989, pp. 617-634.

Winkler, Robert, "Decision Modeling & Rational Choice: AHP and Utility Theory", *Management Science*, Vol 36 #3, March 1990, pp. 247-8.

Winograd, T., *Understanding Natural Language*, NewYork:Academic Press, 1972.

Winograd, T., "Frame Representation and the Declaritive-Procedural Controversy", in D.G. Bobrow and A. Collins, eds., *Representation and Understanding: Studies in Cognitive Science*, New York:Academic Press. 1975.

Winston, P.H., *Artificial Intelligence*, Reading, MA:Addison-Wesley, 1977.

Witte, E., "Field Research on Complex Decision-Making Processes-The Phase Theorem", *International Studies on Management and Organizations*, 2, 1972, pp. 156-182.

Womack, J.P.; Jones, D.T.; and Roos, D., *The Machine that Changed the World*, New York:Rawson Associates, 1990.

Wood, Robert and Bandura, Albert, "Impact of Conceptions of Ability on Self-Regulatory Mechanisms and Complex Decision Making", *Journal of Personality & Social Psychology*, Vol 56, 1989, 407-415.

Woodruff, David C., "Sequencing and Batching in a Constant Work-in-Process Production System", PhD Dissertation, Northwestern University, 1990.

Woods, B.T., "Observations on the Neurological Basis for Initial Language", in D.Caplan, ed., *Biological Studies of Mental Processes*, Cambridge, MA:MIT Press, 1980.

Woods, W.A., "What's in a Link: Foundations for Semantic Networks", in D.G. Bobrow and A. Collins, eds., *Representation and Understanding: Studies in Cognitive Science*, New York:Academic Press, 1975.

Worrall, Norman, *People and Decisions*, Hong Kong:Longman, 1980.

Wright, Gavin, "The Origins of American Industrial Success, 1879-1940", *American Economic Review*, September 1990, pp. 651-668.

Wyatt, Gary, "Decision Making Under Condition of Risk: Assessing Influential Factors", *The Emporia State Research Studies*, Vol. XXXVII, Winter 1989, pp. 1-48.

Yoon, K. and Kim, G., "Multiple attribute Decision Analysis with Imprecise Information", *IIE Transactions*, Volume 21, Number 1, March 1989, pp. 21-25.

Young, J.A., "Global Competition: What's at Stake, Where We Stand", *The 1985 Benjamin F. Fairless Memorial Lectures*, Pittsburgh:Carnegie-Mellon University Press, 1986.

Young, R.M., "Production Systems as Models of Cognitive Development", *Proceedings of AISB Summer Conference*, University of Sussex., 1974.

Young, T.R., "Chaos and Social Change: Metaphysics of the Postmodern", *The Social Science Journal*, Vol 28, No. 3, 1991, pp. 289-305.

Yuhn, K., "Functional Separability and the Existence of Consistent Aggregates in U.S. Manufacturing", *Management Science*, Vol 32, No 1, February 1991, pp. 229-250.

Zaltman, G. & Duncan, R. & Holbeck, J., *Innovations and Organizations*, New York:Wiley, 1973.

Zimmermann, H.-J., "Modelling Vagueness in Decision Models", *Empirical Research on Organizational Decision-Making*, (E. Witte and H.-J. Zimmermann, eds.), Amsterdam:Elsevier Science Pub., 1986, pp. 113-136.

Zirger, B.J. and Maidique, M.A., "A Model of New Product Development: An Empirical Test", *Management Science*, Vol 36, No 7, July 1990, pp. 867-883.

Zohar, Danah, *The Quantum Self*, New York:William Morrow, 1990.

VITA DAVID MICHAEL REDSZUS

Education:

Ph.D.	Industrial Engineering/Management, McCormick School of Engineering & Applied Science, Northwestern University, 1995.
M.S.	Industrial Engineering, Technological Institute, Northwestern, 1988.
B.S.	Industrial Engineering / Economics (Double Major), Northwestern, 1987.
	Supplemental courses at Kellogg Graduate School of Management.

Extracurricular:

Society of Automotive Engineers (SAE), American Society of Mechanical Engineers (ASME), Institute of Industrial Engineers (IIE); President's Club - Sandler Sales Institute; sports car racing (driving & vehicle development), skiing, sailing, diving, photography, golf, racquetball, travel.

Experience:

1982- present, **Precision Automotive Research**, Bensenville, IL.
Partner - Engineering consulting/testing; chemical distributor; fuel injection systems calibration; engine mgmt. system development, alternator development. Developed performance/safety driving curriculum for major international automobile manufacturer.

1990-1992, **Wizdom Systems, Inc.**, Naperville, IL.
Process Analyst/Systems Engineer/Trainer - Founding member/facilitator of GM-EPIC. Project leader of Next Generation EDM team for US Army Materiel Command (AMC). Documented/analyzed functional architechture for Army Active/Reserve and National Guard Personnel Organizations, Defense Intelligence Agency, several Fortune 500 companies. Trained consultants and clients on IDEF & other methodologies for business process improvement. Refined IDEF0 for non-manufacturing applications.

1989-1990, **Northwestern University**, Evanston, IL.
Teaching Assistant - Graduate and undergraduate courses in organization theory, R&D, and statistics for ~25-70 students. Duties included lecturing two-hour class weekly, administering discussion sessions, consulting students on coursework and industrial conditions, administering & grading examinations, and overseeing group projects.

1988-1989, **Northwestern University**, Evanston, IL.
Research Assistant - Evaluation of a technical information resource center at Chrysler Engineering (Highland Park and Auburn Hills, MI). Duties included records analysis, examination of facilities, staff interviewing, questionnaire development and administration, monthly status reports, and final report preparation.

APPENDICES (Volume II)

The following supplemental items can be found in Volume II of this work:

A) Perspective

B) Field Study Overview/Field Sites

C) Field Study Findings

D) Summary of the IDEF0 modeling methodology

E) Some "Undocumented Features" of New Product Development

F) Time, Cost, & Quality: Observed Measures in Conflict

G) Documentation Techniques for New Product Development

H) Development of a New Product Development Framework

I) Engineering Resource Allocation

J) CPP Model Structures

K) Quality Assessment Strategies

L) Expected Effects of Information Efficiency on DT and TTFR

M) Reconciliation of CPP Structure with Real Development Systems

NORTHWESTERN UNIVERSITY

New Product Development Processes: Creation of a Dynamic Analysis Tool

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Industrial Engineering/Management Sciences

by

David Michael Redszus

EVANSTON, ILLINOIS

June 1995

VOLUME II

APPENDICES

Appendix A: Perspective

The declining capability of our nation to develop, produce, and market manufactured goods is a major current issue of concern. The manufacturing component of this issue has received tremendous attention in the past two decades. Great efforts have been spent on examining localized production efficiencies and developing new manufacturing methods for further improvement of such efficiencies. It is recognized that this is an ongoing process of improvement, as technologies develop, regulations vary, and labor capabilities and costs remain in an uncertain state of flux.

The marketing component of industrial performance is a closely intertwined issue--one which cannot be separated from manufacturing or development. Though this study does not focus on marketing techniques, it is acknowledged that development and manufacturing capabilities can have major influence on the marketability of products. Likewise, the quality of marketing feedback to the development and manufacturing activities may influence development and manufacturing proficiency. This pertains to the ability of organizations to ***effectively produce the right goods***. Contrast this with the ability to ***efficiently produce un-marketable goods***.

Far less rewarding analysis has been conducted on how product development processes behave. Many studies in this area are case-studies which arise from specific managerial experiences. Clouded with personal biases, "war-stories", and often founded upon manufacturing analysis paradigms, such product development studies have offered few tangible, transferable lessons for the next-generation development manager. For many existing development managers, each development project is seen as a unique experience,

with only broad guidelines transferable from previous development experience. For individuals who must partake in *new* product development activities, even fewer, less concrete guidelines are available.

The intent of this study has been to accelerate the understanding of the ***new product development process*** for researchers and managers. We shall see that this process is composed of assorted local decision activities, which incorporate customer wants, company strategies, local policies, personal desires, and, at times, various manufacturing and technological capabilities. We see that new product development incorporates elements of ***innovation, invention***, and what is commonly referred to as ***routine development*** practices. In extreme cases, even some ***research*** tasks are observed to occur. In short, ***new product development may be defined as the totality of efforts necessary to develop a brand new product for the marketplace***. Using this definition, we provide management with some helpful direction in their unenviable task of managing the "process" of new product development; we also provide some direction for future research.

Briefly, consider the significance of this issue for both companies and our country. Typically, new product development tasks annually cost on the order of 6% of a company's sales and can require five or more[1] years to accomplish. The degree to which such operations are conducted satisfactorily may affect corporate sales and profits for decades to come. It is observed that failure rates of truly new products can be as high as

[1] Some defense development projects have been known to take over 14 years to complete.

90%, depending on the nature of the product or industry[2]. Based upon our field studies, we have found opportunities which, if exploited, would significantly cut development times and expenditures. We can also expect increases in product success rates, as new products more closely conform to customer needs and wants.

Yet, our concern is not limited to just reducing cost, time, and failure rates for companies engaging in new product development. These are merely steps towards a more important goal. Delivering appropriate products (whether they be service or manufacturing oriented) to market, ***when those markets need and want them***, has been a major ingredient in the success of nations throughout history. If a country, via its own industries, cannot perform this task adequately in a world market, then we may expect to be subjected to the whims of other countries which can. As trade deficits continue to rise, and domestic manufacturing industries deteriorate, we may eventually find ourselves trapped on an irreversible path of economic destruction. This is an unacceptable consequence, for it directly implies a loss of our most treasured national assets: FREEDOM OF ECONOMIC CHOICE.

That Pesky old question--"Why?"

This thesis is an overview of efforts put forth in search of some fundamental truth regarding new product development. It is expected that numerous questions (and, perhaps, even a little controversy) will be raised by the reader from this overview. By the

---

[2] See, for instance, InfoWorld (1995), Gruenwald (1985), Foxall (1984), Rockwell & Particelli (1982), Juran & Gryna (1980).

end of this work, it is hoped that the reader will have a better understanding of some important issues, a glimpse of the relevant dynamics relating to these issues, and will have developed a more complete, integrative understanding of the overall innovation and product development mosaic.

Before discussing the research, we ask two important questions: ***Why are we here?*** and ***Why is this significant?*** Let us investigate these two questions in a little more detail.

Question #1: Why are we here?

It is contemplated that the activities of innovating and developing new products carry a common seed: ***change***. Dissatisfaction with the status quo, for whatever reason, is a driver for enlisting activity to bring about change. Yet, efforts to satisfy this need are notoriously meet with resistance. By overcoming this resistance--to win this struggle--enterprises separate themselves from their unsuccessful counterparts. Yet, successful change requires understanding needs (as separated from wants) and understanding the struggle which interferes with attaining such needs.

Answer (part 1): We *need* to change

The ability to adapt to changing conditions has been a characteristic of all living organisms. The inability or unwillingness to adapt to conditions is a characteristic of species which become extinct over time. For some organisms, adaptation is an ongoing, straightforward process: stimulus-response, stimulus-response. From organism to organism, and from situation to situation, the required stimulant to induce change may vary. Likewise, responses vary in degree and consciousness. Overall, however, we might

say that adaptation is the natural result of a lack of tolerance for, or perceived un-acceptability of existing conditions.[3]

As with all living organisms, organizations need to adapt to changes in conditions. ***Their long-term survival depends on the ability to change.*** As with organisms, mere change by itself is not a sufficient prescription for survival. Incorrect or insufficient changes could spell consequences as disastrous as not changing at all.

Over 50 years ago, economist Joseph Schumpeter[4] outlined the importance of innovation, so that a company's products were always ahead of the competition. In time, he knew, tenacious competitors could and would surpass even a company's continuous incremental improvements, to leap ahead and capture the market. His so-called "gale of creative destruction" was a reflection of the fate of those firms (or economies) unwilling or unable to innovate. Further, he deliberated that an organization's success blinds its personnel to the important need for innovation.

Few listened to Schumpeter view, however, perhaps because of his assumptions of good market communication and low product switch-over costs to consumers, or perhaps because of the growth of the US and world population during the past half century (which may have temporarily or partially masked underlying phenomena). The demise of numerous powerhouse industries in the US has begun to change the attitudes of some

---

[3] It is useful to reiterate that such natural change is not necessarily a conscious decision. A very simple instance of this may be seen in the response of the ciliary and pupilary sphincter muscles in your eyes to adapt to light conditions, under no conscious control of your own.

[4] See Shumpeter (1927, 1928, 1934, 1939, 1962).

managers. Whether this change of heart is a true philosophical shift or just a convenient response to get through tough times remains to be seen.

In the past decade, we have seen a proliferation of concern over the ability of firms to innovate. Specifically, there has been considerable interest in the ***processes*** by which new ideas are transformed into innovative, user-compatible products and services[5].

Despite the wide range of studies and buzzwords floating around boardrooms, managers' offices, engineering facilities, and research institutions, ***effective*** new product development is still a struggle. The very fact that studies and terminology still flood the workplace is, we believe, indicative of the frustration and inability of managers and researchers to understand and/or communicate underlying drivers of new product development processes. In many cases, second-order struggles (and much personal friction) are evident as managers attempt to convincingly communicate their fragmented understanding among one another. When one hears that product development "has already been studied", it reminds me of the fabled story of Dr. Albert Einstein being introduced to a college student at a dinner party:

When the somewhat naive student asks the doctor what he does, the Nobel Laureate answers that he has "...dedicated his life to furthering the understanding of our existence through investigations within the field of physics."

The student's reply?..."Oh really, I learned that in my A35-2 course last year!"

---

[5] For example, see Wheelwright & Clark (1992), Rosenau (1990), Amendola & Gaffard (1988), Sheth & Ram (1987), Gruenwald (1985), Langdon & Rothwell (1985), and Johne (1985).

The point is that we, as a society, have not learned very much about product development and we have not learned very much of significance from past research on development. If we had done so, and had established reasonable tools to apply such understanding, then we would have a nonexistent problem. Unfortunately, it is a problem for which we may *never* have complete understanding. Fundamentally, this thesis hopes to provide some insight into ***why*** this is so.

Answer (part 2): We *struggle* with change

To ensure survival, both organisms and organizations of all sizes (from the amoeba to the largest of governmental and corporate entities) need to continually and iteratively perform four major tasks:

1) ***Maintain awareness*** of changes in conditions;
2) ***Recognize*** which modifications in behavior are needed;
3) ***Induce*** different behaviors in response to such need(s);
4) ***Check*** that the new state (post change) is appropriate.

Inherent in the above tasks are a few major assumptions. These require that the organism/organization:

- conducts comprehensive, accurate ***environmental scanning***;
- makes the correct ***assessment of necessary changes***;
- has sufficient ability to ***implement*** necessary changes;
- can maintain an ***objective capability to assess*** its new state.

In this study, we recognize, but have not focused on the mechanism by which management in organizations ***recognize their need to change***. Such activities include competitive analysis, market analysis, regulatory assessment, and so on. Though these areas are extremely important, they are outside the domain of this study. It is hoped that more focused research in such areas will be conducted, for it is desperately needed.

We have selected the task of examining how organizations ***go about inducing change***, and offering suggestions for how such adaptive practice can be enhanced over both the immediate and long time frames.

Given the importance--the necessity--of change, too few firms have demonstrated the ability to incorporate change in a satisfactory manner. This study directly addresses this issue. We have encountered evidence of some drivers which can either foster or stifle new product development. The evidence applies to both large and small firms, old and new firms, firms in the US and Germany, and across industries.

Question #2: Why is this significant?

The significance of research is often judged by the degree to which a new finding has furthered human knowledge. This has been consistent with the currently fashionable research paradigm which I call "linear incrementalism." In this view, the scope of human knowledge is continuously increasing. To clarify how this research seems to fit into the ever expanding base of scientific knowledge, we briefly consider the concept of a knowledge tree, and then contemplate how this research differs from more conventional research paradigms.

Answer (part 1): There exists *need* to integrate the "Tree of Knowledge"

A simple framework for considering the dynamic scope and scale of scientific understanding can be dubbed a "tree of knowledge." In this scenario, consider that the "trunk" constitutes the basic knowledge upon which all future knowledge is based. Each main branch of the tree represents a particular domain or field of knowledge. Sub-branches and shoots represent particular specialties within an "established" field of knowledge. Using this visualization, "new" research is any addition to knowledge which permits the tree to grow; research may be considered the "buds" of the tree of knowledge. Clearly, the faster new buds form and germinate into new leaves and, eventually, whole new branches of established knowledge, the faster the tree of knowledge will grow.

But what causes new bud formation? Why do trees grow? What prevents a tree from growing? In an actual tree, there exists a complex system of vascular tissue which provides minerals and moisture from the roots and nutrients, in the form of glucose, from photosynthesis in the leaves. Although the tree responds to the orientation of sunlight, the vascular system provides a certain ***integration*** of the tree, so that nutrients in one branch impact growth rate in another branch. Biologists call this process ***translocation***. How does such translocation occur in the human tree of knowledge? Further, how is the appropriate ***direction*** of growth determined in the human "knowledge tree"? Biologically speaking, does our knowledge tree have an orderly tropistic or nastic response system?

We assert that knowledge growth and resultant direction has been shortsighted during the past few decades. The "translocation" system has been constricted to a variety of localized flows, and has offered little cross-nutrition of knowledge from other branches of

knowledge. Witness a recent estimate that there exist on the order of 20,000-50,000 research journals[6] ; yet, many researchers have difficulty following more than half a dozen journals, usually closely related to their specialized fields of interest.

With such fragmented exposure to vastly growing information bases, we seem to have lost much of our ability to integrate our knowledge. It is as if we actually have a thorny knowledge tree with many fragmented, perhaps broken branches. Some merely consider this fragmentation as the natural process of "specialization". As we become increasingly specialized, however, at what point does new research lose relevance to practical reality? It seems that we have retained the leaves from previous growth cycles, which in effect block the sunlight (real world) from our direct view. Using terms to be described at length later in this work, the process of moving the human knowledge frontier may be both a ***complex*** and ***complicated*** endeavor. Because of misinterpretations and simplifications of the "real world", unrealistic "problems" are formulated for "convenient" investigation. Is it any wonder that the research community is often regarded with a degree of contempt among management?

Answer (part 2): We are *changing* the research agenda

This study was conceived, developed, refined, executed, and reflected upon with the knowledge tree concept in mind. Its significance may be found on a number of methodological fronts, not merely technical discovery (although there are several technical findings that many will find significant). Yet, this is not a methodological essay, in the traditional sense of the word.

---

[6] See Leo (1988) and Melinowski & Richardson (1980).

This work is a foray into fields and issues in product development from which researchers have perpetually and continuously diverged. Just when they appear to get close to some new finding, researchers (including many progressive managers) seem to quit the search, with one of three results:

1) ***Inconclusiveness***: After rummaging one or more organizations, researchers fail to find the cure-all answer they are looking for. They come to the conclusion that there is no "fountain of youth" or concise "meaning of life", and in some cases find no good (i.e., rational) reasons for observed phenomena, and that further study will just be a waste of precious time and resources.

2) ***Eureka!***: After performing a moderate amount of investigation, they feel they have discovered something new which, now found, will make them the guru's of their field. After initial euphoria, they eventually realize that their "discovery" is nothing spectacular, actually just a new spin on a realization discovered eons (or centuries!) ago. Some fail to come to even this latter recognition, resulting in frustration for much of their careers as they have difficulty convincing others of the significance of their findings.

3) ***Puzzling profundity***: Sometimes, investigators find quasi-conclusive answers, but really only enough to recognize that they have almost no understanding...They realize that the problem is so far beyond their mental capability that any finding is likely to be incorrect on a number of fronts. In a sense, they have looked into an abyss, and shied away from climbing inside to investigate its structure.

Recognizing this dismaying trend of research on new product development, we were determined to find an alternative. Perhaps more valid results could be accomplished by taking different approaches to understanding new product development. Perhaps there was some order to be found in the often inconclusive-looking observations. Perhaps we could see why "discoveries" to some are "old knowledge" to others. Perhaps, the "abyss" is merely an overly complicated view of some simpler underlying principles of human interaction. By deciphering these principles, couldn't this mystical, perhaps holographic image be shattered?

This research swings from an overall orientation of practical consideration (new product development as an agent of necessary change), which came from many detailed field analyses, to more abstract analysis methods (CPP structure and dynamic process analysis). Upon further consideration, we see some fundamental truths about human behavior in development organizations. One example of such truths is an observed tendency to mis-prioritize communications among each other, serving to reduce *all* of our productivity. When understood, such truths may be much more important than the detailed nuances which we researchers take such pride in discovering and relaying. Thus, we believe that the abstract portions of this research have helped tell us more about the *real* process of development than many of our treasured, though paradigm-biased, first-hand experiences. This ironic twist of methodology was not realized until this study was nearly complete. It may or may not be a testimony to the validity of this study, but the conduct of this research seems to closely parallel the very convoluted, bi-directional processes which are under scrutiny. Whether and how well the results suggested here will

stand the test of time may depend on the reliability and frequency of additional observations in (and thoughts about) the field of new product development.

Not all of the observations described herein can be considered unique. Rather, much of our task has been to reconcile findings of this work with those of past research works. It is expected that such integrative research will provide new opportunities, previously not considered, to enable the knowledge tree to flourish.

Appendix B: Field Study Overview/Field Sites

B.1. Field Study Overview

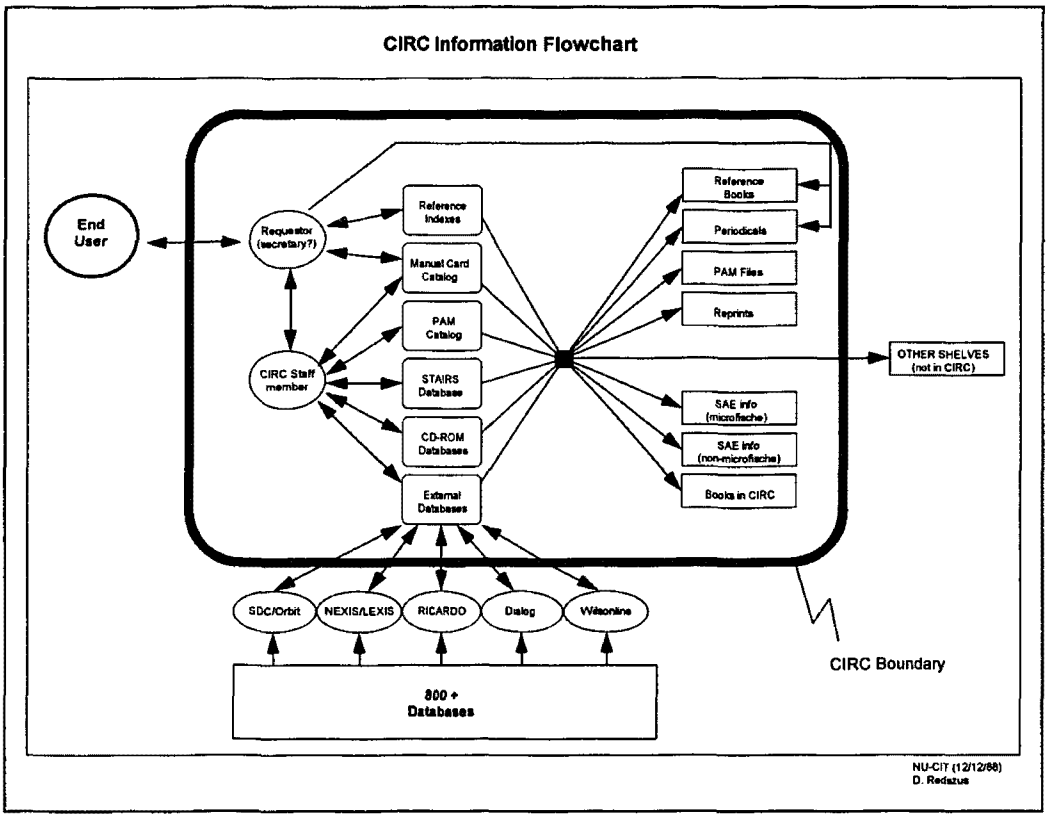
Sample Site 1: Engineering Library

At our first site, we examined the operations of a technical library at a major U.S. automobile manufacturer. A specific objective was to evaluate the technology transfer characteristics among development engineers, between development engineers and the library, and between these parties and the outside world. On-site interviews were performed with a variety of library users. A questionnaire was developed and administered to users throughout the immediate engineering community, asking about the existing capabilities, as well as opportunities for future library capabilities. Records of library requests were reviewed for content and frequency. During on-site records review, we engaged in direct observations of the use of the library[7]. Several document flow models were developed, which illustrated the transfer of library data within the confines of the facility, as well as between sister facilities and "users" (engineers). A sample information flow model developed from this site is demonstrated in Exhibit B.1.

[7] An insightful, if somewhat discouraging, finding from such observation was that the library was utilized primarily as a photocopy center for engineers anf their staff, and as a relaxing leisure reading center. The intended role of the library as a progressive technical information transfer center was of tertiary concern among many engineers. Based upon the small utilization rate of the library, there was a distinct possibility that many engineers were not aware of its existence or the significance of its capabilities.

Exhibit B.1.

An Information Flow Chart at One Site



Sample Site 2: Engineering/Development Center

Nine months were spent on-site at the main engineering facilities of another U.S. automaker. The functions inherent in developing production automobiles were rigorously documented and modeled, using the IDEF0 modeling technique. A core team of 10-14 design engineers, two managers, and an average of three facilitators conducted this monumental task. Over 1200 specific engineering functions were identified and validated. Relations between functions were identified, documented, and validated.

For the remainder of this research, the IDEF methodology[8] was utilized as a basic documentation tool, with specific data collection techniques supplemented on an as-needed basis. The IDEF methodology provides a basic framework, or "skeleton", upon which specific findings and nuances can be hung. This is accomplished through the use of detailed project glossaries and textual descriptions which accompany every visual diagram (per the structured rules of the methodology). Thus, IDEF is a convenient structure to help document large-scale operations in an objective, consistent manner. This was found to be particularly helpful for documentation teams which were made up of several diverse process participants. In such cases, each team member was responsible for developing and verifying major sections of the integrative model of the overall organization under study. Thus, the structure, simplicity, flexibility, universality, and availability of this methodology provided a unifying environment for process documentation.

This thesis is not about IDEF, however. It is about some specific findings which arose through the use of IDEF in documenting the "process" of new product development. However, the characteristics inherent in this methodology facilitated an objective insight to the process which would have been extremely difficult to obtain otherwise. It is important to keep in mind that IDEF0 had not been used in for documenting engineering processes prior to this study.

---

[8] For information systems developers, IDEF1 (and IDEF1x), is a standardized entity-attribute(+ relation for IDEF1x) documentation methodology which can be used to trace the association between fields of databases of varying structures. For more information about the IDEF methodology, refer to Appendix D.

Perhaps the most dramatic aspect of this insight was the acquired capability to ***simultaneously*** "see" development processes from both global (hi-level) and local (low-level) perspectives. Because of the functional decomposition inherent in the modeling methodology, it was (in principal) relatively simple for the trained eye to follow the transfer of ***elements***, information or prototype assemblies, from function to function (or, in some cases, from department to department). Thus, such models indicated movement or, more precisely, the ***channels*** over which movement of elements could occur.

The concept of movement, or "flow", throughout an organization enabled a unifying theme upon which this analysis was based: that ***product development could be characterized as an "accumulation" process***. In this process, a variety of ***inputs*** (broadly categorized as raw materials) are gradually transformed into one or more organized ***outputs*** (which may be composed of the "physical prototype," supporting "prototype documentation," and any remaining materials (discarded or not) which are not encapsulated in the above two). Along the way, there are certain ***controls*** (marketing requirements, machine capabilities, government regulations, company policies, social customs, laws, budget allocations, etc.) which direct and limit the process. There are also certain ***mechanisms*** (property, human beings, machines, funds, etc.) which enable or propel the process. Thus, inputs gradually accumulate with one another, according to the directions outlined in the controls and with the facilitation of the various mechanisms. The end results, the output(s), are a reflection of such efforts. The precise way in which this all happens is the ***process***.

Documentation of the process consisted of three basic steps:

1. ***Identify and categorize the major functions*** which are performed in new product development;
2. ***Identify and classify the critical entities*** (inputs, controls, mechanisms, and outputs) which are utilized and created during new product development;
3. ***Build and verify a model*** which conveys the association of the functions from step (1) with the entities from step (2).

Thus, documentation efforts were not limited to ***identifying*** the entities and functions of new product development, but extended to finding out ***how*** the various entities connected the many functions. As a result of such effort, it could also be possible to see how functions ***affected each other*** during the course of new product development.

Data collection throughout the development organization played a major role in developing an accurate functional model. For instance at site #2, first-hand interviews were conducted with approximately 200 engineers, managers, and administrative personnel over the course of 6 months. A census was developed and administered for response by 500+ engineers in the facility. Direct observation of every phase of the vehicle development process was undertaken over the course of this nine month period.

As for the model, over 1200 distinct functions were identified, documented with over 600 diagrams and textual descriptions. Over 2500 distinct terms from the model were entered into a comprehensive product development glossary. To get a feel for a sample "high-level" diagram of the IDEF0 functional model from this site, refer to Exhibit B.2. Compare this to the "official" functional diagram, derived by many of the same people several years earlier, presented in Exhibit B.3.

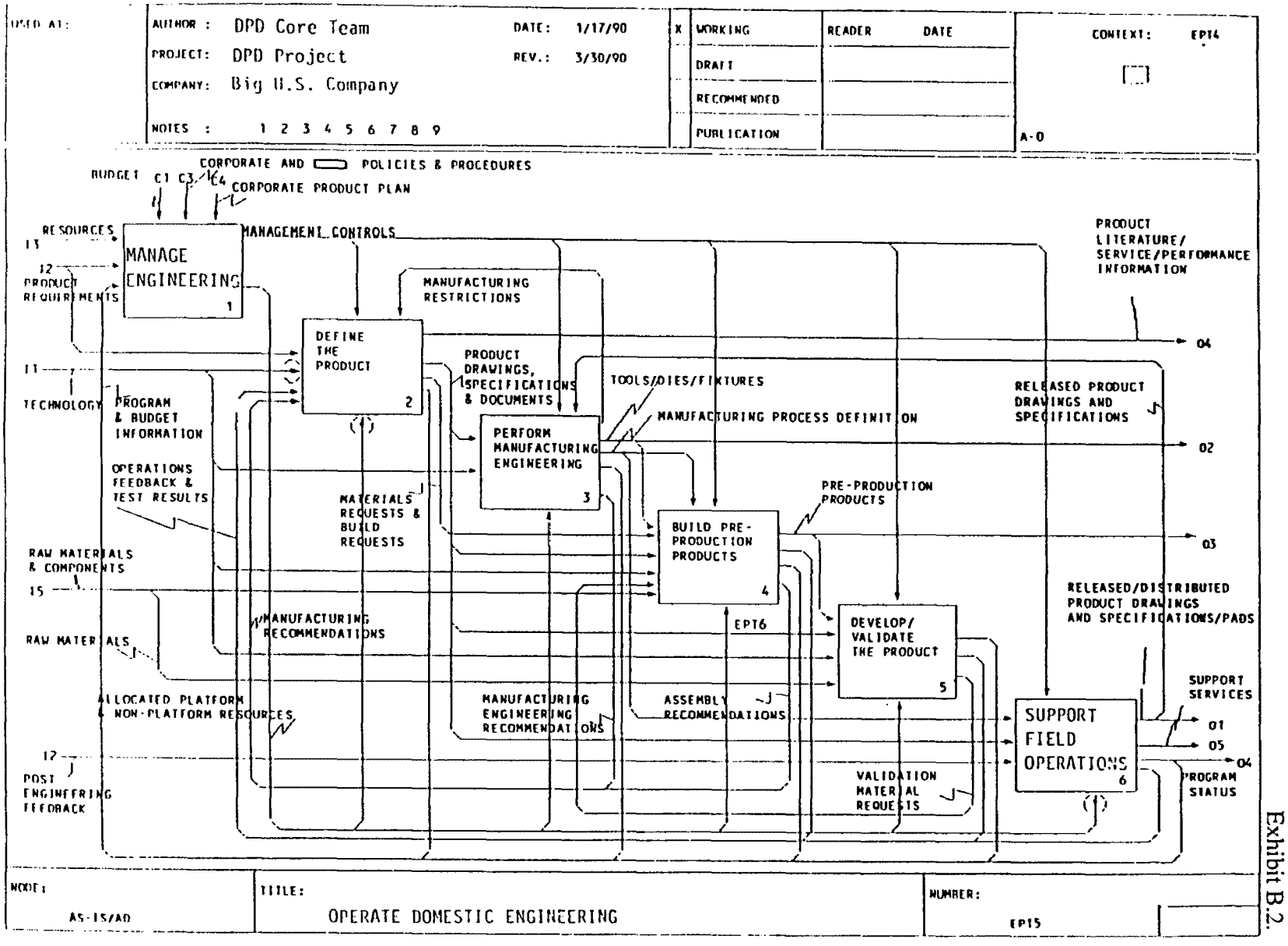
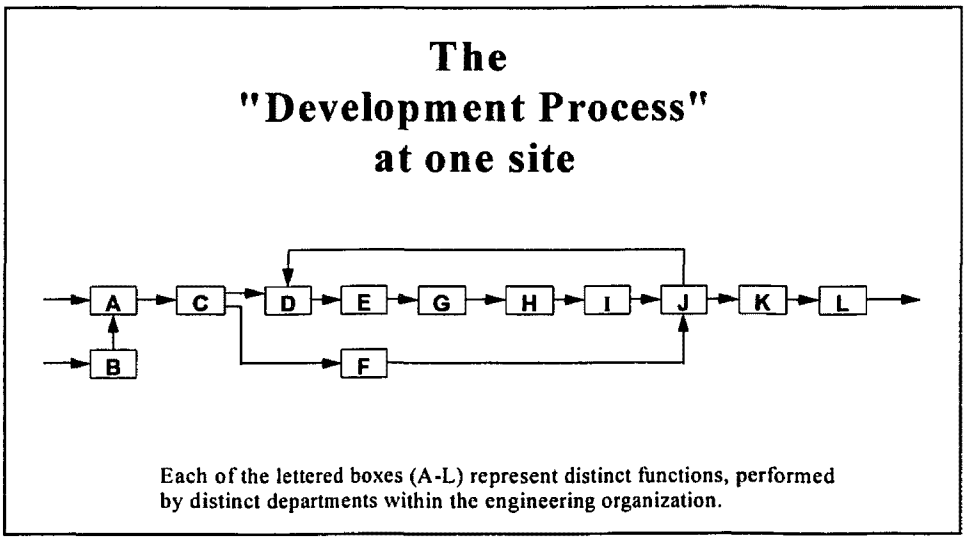

USED AT:
AUTHOR : DPD Core Team
DATE: 1/17/90
PROJECT: DPD Project
REV.: 3/30/90
COMPANY: Big U.S. Company
NOTES : 1 2 3 4 5 6 7 8 9
X WORKING
DRAFT
RECOMMENDED
PUBLICATION
READER DATE
CONTEXT: EPI4
A-0
CORPORATE AND POLICIES & PROCEDURES
BUDGET C1 C3 C4
CORPORATE PRODUCT PLAN
RESOURCES
I3
I2
PRODUCT REQUIREMENTS
MANAGE ENGINEERING
1
MANAGEMENT CONTROLS
MANUFACTURING RESTRICTIONS
PRODUCT LITERATURE/ SERVICE/PERFORMANCE INFORMATION
O4
DEFINE THE PRODUCT
2
I1
TECHNOLOGY
PROGRAM & BUDGET INFORMATION
PRODUCT DRAWINGS, SPECIFICATIONS & DOCUMENTS
TOOLS/DIES/FIXTURES
MANUFACTURING PROCESS DEFINITION
RELEASED PRODUCT DRAWINGS AND SPECIFICATIONS
O2
PERFORM MANUFACTURING ENGINEERING
3
OPERATIONS FEEDBACK & TEST RESULTS
MATERIALS REQUESTS & BUILD REQUESTS
PRE-PRODUCTION PRODUCTS
BUILD PRE-PRODUCTION PRODUCTS
4
O3
RAW MATERIALS & COMPONENTS
I5
RELEASED/DISTRIBUTED PRODUCT DRAWINGS AND SPECIFICATIONS/PADS
MANUFACTURING RECOMMENDATIONS
EPT6
DEVELOP/ VALIDATE THE PRODUCT
5
RAW MATERIALS
SUPPORT SERVICES
ALLOCATED PLATFORM & NON-PLATFORM RESOURCES
MANUFACTURING ENGINEERING RECOMMENDATIONS
ASSEMBLY RECOMMENDATIONS
SUPPORT FIELD OPERATIONS
6
O1
O5
O4
PROGRAM STATUS
I7
POST ENGINEERING FEEDBACK
VALIDATION MATERIAL REQUESTS
MODEL:
AS-IS/AD
TITLE:
OPERATE DOMESTIC ENGINEERING
NUMBER:
EPI5

Exhibit B.2.

Exhibit B.3.



Sites 3-42: From the world's largest development organizations to some of the world's smallest

To avoid site-specific biases, it was recognized that many more observations of development activities would be needed. In the subsequent 21 months, this was accomplished in a number of ways:

- The development operations of the ***US Army Materiel Command*** (AMC) and six of its Major Subordinate Commands (MSC's) were examined in detail, on-site, over a one-year period;
- The communication characteristics between the AMC sites and seven of its ***major contractors*** were examined via site visits. Questionnaires received from 28 contractors were also processed.
- Eight research, development, and production operations of five ***major German industrial companies*** were observed and reviewed;

- The development, test, and production facilities of fifteen ***U.S. manufacturing companies*** were observed.

Analysis of each of these site types involved on-site visits. During each visit, detailed first-hand observations were made about the process and environment of development. Functional modeling, questionnaires and formal (and informal) interviews were conducted with developers, technicians, administrative staff, managers, and executives/proprietors. Comprehensive functional models and a basic IDEF1 data model were developed for the AMC sites. Interfaces between these models were documented via the contractor visits. At two of the other sites, more elementary models were developed each over the course of several weeks. At the other sites, direct observations were noted, albeit no models were developed.

As a result of the field data collection, information was obtained on the use of information technology, engineering data formats and content, Government and corporate standards, training levels, and a host of cultural subtleties.

Throughout the data collection process, it became widely apparent that there existed a high degree of ***interdependency*** between developers. This was true whether one considers interfaces with other developers within or external to a particular organization. Thus, it was observed that *inter*-organizational, as well as *intra*-organizational, communication is essential to the development process. The nature and effects of such communication is of major consideration and played a significant part in the development of a new product development analysis model described later in this report.

B.2. Field Sites

During the course of this study, we were in contact with many different organizations. Some were specifically development oriented; some were not. This provided us with a diverse array of perspectives from which to draw. Some sites we visited first-hand; others provided useful information via questionnaires, phone calls, or personal interviews. We are deeply indebted to the participants at each of these sites. Without their cooperation, this study would not have unveiled many of the dynamics which we address. The following pages include two lists. The first is a list of most of the field sites which we visited first-hand. Following this, we list organizations which we did not visit first-hand, but whose personnel provided us useful second-hand information, based upon their experiences.

B.2.1. A Sampling of Visited Sites

AMG
Development and Production Facilities, Affalterbach, Germany

AMOCO
Research Laboratories, Naperville, IL

Boeing Corp.
Commercial Aircraft Engineering/Production, Seattle, WA
Defense and Space Group, Philadelpia, PA

BMW, AG
FIZ (Engineering Center), Munich, Germany
Munich Manufacturing plant (3-series), Munich, Germany
BMW M GmbH (specialty products), Garching, Germany

Chrysler Corporation
Engineering Center, Highland Park/Auburn Hills, MI
CIRC (Information Resource Center), Highland Park, MI

Computervision, Inc., Bedford, MA

Daimler-Benz (Mercedes-Benz), AG
Engineering Center, Untertürkheim, Germany
S,E class Mfg Plant, Sindelfingen, Germany
Mercedes Sport-Technik R&D Center, Fellbach-Schmiden, Germany
Vehicle Preparation Center, Franklin Park, IL
Parts Distribution Center, Carol Stream, IL

FMC, Ground Systems Division, Santa Clara, CA

Ford Motor Company
Body Engineering, Dearborn, MI
Body Test Labs, Dearborn, MI

General Motors Corp.
Advanced Vehicle Engineering (AVE), Troy, MI
Manufacturing Technology Center, Warren, MI
Chevrolet-Pontiac Canada (CPC) Engineering, Warren and Troy, MI
GM Research Laboratories (GMR), Warren, MI

Linder Rennsport, GmbH
Development and Preparation Facility, Füssen-Weinensee, Germany

MAHLE, AG
Advanced Engineering/Development Center, Stuttgart, Germany
Automotive Piston Manufacturing Plant, Stuttgart, Germany

Matrix Technologies, Inc., Toledo, OH

McKee Engineering, Inc., Lake Zurich, IL

NALCO Chemical
Central Research Laboratories, Naperville, ,IL

National Academy of Engineering, Washington, DC

Penray Company
Administrative Headquarters, Des Plaines, IL
Filling/Distribution Facility, Elk Grove Village, IL

Porsche AG

Weissach Engineering Center, Weissach, Germany
Production Facility, Stuttgart, Germany

United Technologies
Chemical Systems Division, San Jose, CA

Nomura Enterprise Inc., Rock Island, IL

Raytheon Company, Land Based Systems, Andover, MA

US Army
Army Materiel Command (AMC), Alexandria, VA and the following Major Subordinate Commands (MSC):
AMCCOM (Armament, Munitions, and Chemical Command) Rock Island, IL
ARDEC (Research, Development, and Engineering Command), Picatanny Arsenal, Parsipanny, NJ
AVSCOM (Aviation Systems Command), St. Louis, MO
CECOM (Communications and Electronics Command), Ft. Monmouth, NJ
CRDEC (Chemical Research, Development and Engineering Center), Aberdeen Proving Grounds, MD
IEA (Industrial Engineering Activity), Rock Island, IL
MICOM (Army Missile Command), Huntsville, AL
TACOM (Tank Automotive Command), Warren, MI

Rock Island Arsenal, Rock Island, IL
Army Reserve Personnel Center (ARPERCEN), St. Louis, MO
Personnel Command (PERSCOM), Alexandria, VA

US National Guard Bureau (NGB), Alexandria, VA

Wizdom Systems, Inc., Naperville, IL

B.2.2. Samples of Contacted Sites (Secondary Sources)

CACI Products Co., La Jolla, CA

CERC (Concurrent Engineering Research Center), West Virginia University, Morgantown, WV

CSC (Computer Systems Corporation), Moorestown, NJ

EDS, Inc., Troy, MI

OSD CALS (Computer-aided Acquisition & Logistics Support), Washington, D.C.

JCALS (Joint Computer-aided Acquisition & Logistics Support), Washington, D.C.

Hewlett-Packard, Boise, ID

Honeywell, Minneapolis, MN

IBM, Durham, NC

Illmor Engineering, Switzerland

Martin Marietta, Orlando, FL

MathSoft, Inc., Cambridge, MA

MathWorks, Inc., Natick, MA

Matsushita Electric (Panasonic), Franklin Park, IL

Microsoft Corp., Redmond, WA

Milliken Research Associates, Inc., Williamsville, NY

Motorola, Shaumburg, IL

Phillips Petroleum, Bartlesville, OK

Robert Bosch Corp., Broadview, IL

Appendix C: Field Study Findings

In the following pages, we present an overview of some major findings and conclusions from the field studies. Because there exist many field study findings, we have categorized them into three major areas: ***Recognition, Technical Performance,*** and ***Management Paradigms***. Within each of these categories there are several classification areas.

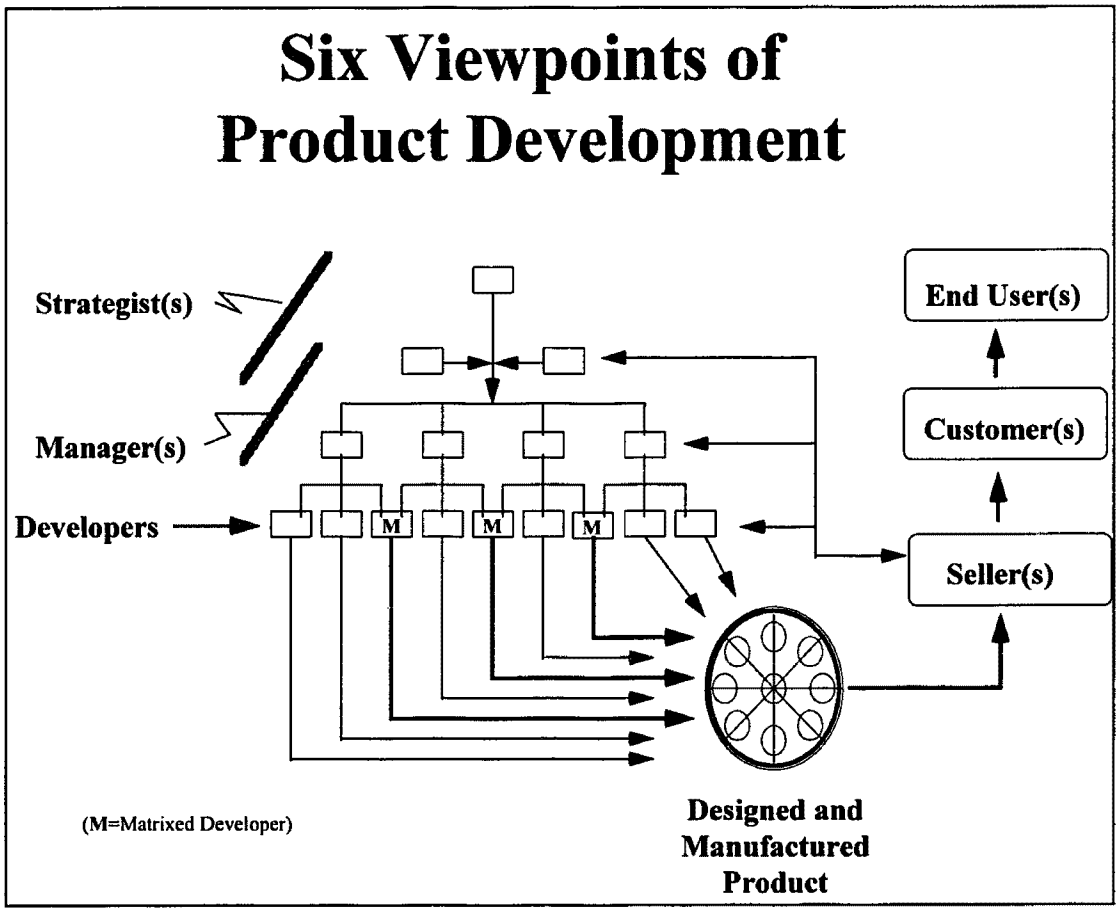
Observed Recognition Issues (Problems of Perception)

The most significant discovery of our field studies has been the over-riding observation that there exist many different perceptions of how new product development proceeds or "should" proceed. As a result, there exist many different opinions and theories about the "best" solutions to the problems of new product development. Recognition issues are classified into major areas: ***Viewpoint, Definition,*** and ***Purpose.***

Viewpoint

Based upon our interviews, there are six types of players in new product development, each of whom has a critical role. These players are the strategist, manager, developer, seller, customer, and user. Refer to Exhibit C.1.

EXHIBIT C.1.



These players are defined as follows:

The ***strategist*** is an individual who is responsible for determining the overall direction of a development program. In addition, the strategist is expected to coordinate various developing and existing programs to best maintain continuity to customer needs.

The ***manager*** is the individual or set of individuals which oversee the activities of developers. The manager is expected to interpret directives from the strategist, to better guide his developers.

The ***developer*** is the individual or team of individuals which actually designs the product, or as is more prevalent, one or more components of the product. In the latter case, the developer is usually responsible, officially or unofficially, for integrating product componentry. This person is involved with the development of the product on a day-to-day basis, under the direction of the manager.

The ***seller*** is the individual or collective organizations which attempt to communicate the features and benefits of the product to customers. One might suspect that the seller should perform a liaison role between customers and developers. In some cases, this role was performed. In others, the developers perform more or less "open-loop." Often the seller does not interface with developers at all, but rather consults with managers or strategists.

The ***customer*** is the individual or organization which buys, leases, barters, or performs some other transaction to obtain the product.

The ***end-user*** is the individual who actually applies the product to his application. Notice that the user is not required to be the purchaser of the item, and thus may be distinct from the customer.

It was observed that distinctions between these players is sometimes far from clear. In some large organizations with a few large customers, these six players were clearly separate. For smaller firms, and increased customer diversity, the separation of these players became less straightforward. In fact, there were several instances where the strategist, manager, and developer were the *same* person. In such cases, it is not unusual for this person to also be a user and seller, in search for other potential users, who might be persuaded to become customers.

Despite observations that there can exist fuzzy boundaries between such players, it was also observed that such players can carry very ***different impressions*** of what development is or should be. A caveat to this observation is that different players carry ***different objectives*** about products and processes.

In this sense, it was found that the oft referred to concept of "product champion" was not limited to the stereotypical senior manager. Rather, it was found that any individual with a ***strong interest in*** and ***adequate resources for*** development could facilitate a new development process. In several noteworthy cases, the interest level and personal resource levels were high enough among developers, and sufficiently low among managers and/or strategists, that certain developers initiated development activities ***outside*** of their employer's domain. As development progressed, a new company was initiated by such developers, who then abandoned their former employers. When personal resource levels were insufficient to perform such autonomous development, some developers (with clandestine support of some managers) initiated bootleg development activities within the company, with mixed results.

Definition

A definition for new product development was briefly discussed in the introduction of this study. Yet, there seem to be as many definitions for this term as there are independent minds defining it. As with other issues, this definition seems to be outwardly dependent on one's viewpoint. Users, customers, sellers, strategists, managers, and developers all see "new" developments in particular ways.

Moreover, people who operate from any one of these viewpoints do not necessarily carry a homogeneous definition of such development. This was an important finding during the field study. Contrary to the simplified hopes of past research, not all developers are alike. Neither do all managers nor all strategists carry the same philosophy towards new product development. In marketing, it has long been realized that customers and users have wide-ranging and highly variable needs, wants, and abilities to obtain such needs and wants. To the abhorrence of many sales managers and customers, so do salesmen!

In our literature review, we discussed some problems of various scientific definitions for basic research, applied research, invention, innovation, and development. It is a finding of this study that ***new product development is composed of a plethora of fields, in varying depths that cover the spectrum from basic research to routine development.*** Attempts to classify new product development under any one of these areas blatantly ignore the observed fact that participants (i.e., players) must "do what it takes" to deliver new products to users. If this means occasionally creating or discovering something new (i.e., invent), then so be it. If it requires incorporating disciplines with which one is currently unfamiliar, then so be it. One might imply that the definition of new product development will probably remain fuzzy, as long as developers engage in a wide range of activities during development.

Purpose

In the introduction to this work, we discussed the purpose of new product development, in the sense that it is a requirement for survival in an environment of continuous change. While observing the process of development, however, it became apparent that there

could exist various interpretations of the changing environment, as well as various prescriptions for what constitutes appropriate "change."

This variation in opinion begs the question of the purpose of a specific development or design change. Curiously, this question was often not asked by me, but by numerous ***participants*** in the development process itself. Consider the following anecdote from a senior engineering manager at a major international automobile manufacturer:

> A design/release engineer was assigned to a product development team (PDT) with the task of designing the rear cargo section on a new passenger sedan. Specifications were initiated, modified, and finalized as to the cargo space height, width, depth, and perimeter dimensions. These were determined in several interfacing meetings with engineers responsible for driveline, interior (including rear seat and stereo system), fuel system, chassis electrical, and suspension systems, as well as styling and marketing departments.
>
> The design/release engineer and his subordinates developed and refined their design over the course of 30 months, from vehicle concept finalization through the prototype build stage. Three and one half months prior to the scheduled start of production, the Vice President of Marketing sent a memorandum to the Chief Engineer, stating that the rear hatch needed redesign. "Insufficient Rear Hatch Access" was discerned as the culprit.

A common customer/user activity was noted: loading a full bag of groceries (in an upright manner). The trunk had adequate space to *hold* the merchandise, but the merchandise could not be satisfactorily *put* in the trunk!

The problem of accessing the rear cargo area was addressed by the design team, and apparently resolved with the assistance of the styling department (modification of trunk lid width and rear window rake) and a sub-contractor's innovative dual-elbow hinge design. Still, the access dimensions were deemed insufficient, given this previously unconsidered, though common, customer activity. Short of major redesign of the rear of the vehicle, no solution to the problem was readily apparent.

The decision was made to lengthen the vertical drop of the trunk lid, nearly down to the bumper level. This necessitated a redesign of the rear fascia, requiring expedited efforts from personnel in the body, chassis electrical, styling, and certification departments. Given the timing, unprecedented tool and die modifications were made to minimize the impact on production release.[9]

From a design engineer's perspective, this example is relatively mundane; it is yet another case of engineering change in response to a change in perceived requirements[10]. Many

---

[9] As an interesting aside, this unique solution, which was created out of desperation in reaction to a pressing last-minute problem circa 1983 has been a feature of this company's products ever since. Moreover, the feature has been incorporated by a host of competitors' designs, as well!

[10] In the preponderance of cases, requirements changes were ***internally*** driven (i.e., came from within the development system). For manufacturing intense firms, manufacturing cost concerns drove many such changes. Unfortunately, fluctuating cost/benefit criteria seems to play a non-trivial role in the dynamic nature of engineering requirements.

engineers live with this "fire-fighting" scenario every day. It is a way of life throughout the development process.

In this example, as well as hosts of others, technical requirements were initiated prior to conceptual development. However, it was impossible to specify *every* nuance of the preliminary requirements early in the development process. Even if it were technically possible, the shear volume of requirements would have been much too large for a small concept development team to process. Thus, simplifications are made for communication's sake. In this case, a senior design engineer was involved at the conceptual stage of development, with the intent that continuity between the design concept and the design execution could be retained.

Unfortunately, product requirements are typically interpreted by different development personnel in different ways and at different rates. Each group of managers extracts their segment(s) of the overall requirements pool, to be satisfied by their specialists, or developers. From the beginning of the process, latent ***gaps*** in development requirements exist. Allocation activities may, at times, have the effect of creating new, internally-driven gaps. Thus, from the beginning, developers are on an extended hunt for missing requirements. Some gaps are discovered early in the process and are ***resolved without major incident***. Others, such as in the above example, are not discovered until late in the development process, and may ***result in turmoil.***

Further, just because a requirements change has been approved and induced does not mean that all the "affected" developers *are aware of the change*. Upon our review of

configuration management (CM) systems, it was apparent that the following two problems exist:

1. Changes to a component are induced by a sub-assembly design team, but they feel that their changes do not significantly affect others, so they ***do not announce them.***

2. Changes are announced, but ***not heeded by other developers***, because of communication imprecision, CM system failures, or politics. Most often, however, this is a problem of delayed communication of engineering changes. By the time other developers are aware of, and understand the significance of the change(s), they have already proceeded using existent pre-change requirements.

There is another, more subtle dimension to the requirements definition problem, however, which was widely apparent in this research. This is the degree to which development personnel understand the ***reason*** for the requirement itself. Note that this entails more than identifying and recording the requirement. It calls for understanding the customer's and, perhaps more importantly, the end user's application of the product. ***In several significant cases, the developer's impression of product use was significantly different from that of the customer or user.*** Such problems, of course, can be magnified when external requirements actually do change over the course of development.

Consider the operation of another automotive development organization, which has not responded to "external" requirements, and has consistently seen its share of market fall over the past decade:

> An executive with little engineering background, nor early input to the design reviewed a nearly complete prototype, only a few months before production was to begin. His remark: "I don't like the front end."

Given the cultural climate of the development organization, it was difficult for developers to precisely identify *what* the executive did not like about the front of the vehicle. Second-guessing (i.e., ignoring) the executive was not an option. Yet, past requests for clarification had been met with insult, a condition which development management did not wish to repeat.

So they guessed. They reviewed past designs which he had championed, and tried to modify the existing design to better conform. This drew strong anti-sentiment from the styling group (which, as is typical, was already at odds with the engineering group). The HVAC and engine groups had to make modifications to accommodate engineering-driven airflow needs. The vehicle went back to the wind tunnel. For better handling, front suspension geometry needed to change as a result of a changed pick-up point, but cost considerations precluded this. Several dozen vendors were affected, many of whom had already begun gearing-up for impending production. Purchasing contracts had to be re-negotiated. Production plant plans had to change. Long-approved crash-tests has to be re-administered. So many ripple effects were manifest that the car was late...by nearly a year.

The executive's consequent approval after these changes were executed never was forthcoming, for he had retired by that time. Yet, the development project continued on the trajectory instilled by development management as a response to his casual remark, some six months earlier. By this time, however, the revised design *had* to be produced[11], for the existing product was getting "old in the tooth."

Such explosion of effects would likely have been avoided by accurate, up-front requirements, and may have been drastically reduced by better clarification of the reason for change, even at the late prototype design-review stage. Perhaps only a piece of

[11] Per the directive of *his* replacement.

chrome trim or subtle change in grille treatment would have sufficed. More importantly, notice the lack of customer/end-user involvement in this scenario. Furthermore, despite the seemingly extraordinary nature of this situation, such scenarios are commonplace in developments today. Developer after developer have incredible accounts of such problems in their current (and previous) workplaces.

Observed Technical Performance Issues (Process behavior)

Technical performance issues are findings which reveal some characteristics about the ***behavior*** of development organizations. Note that none of these findings are a reflection of the ***attitudes or beliefs*** of management about their development organizations; that is reserved for the next category of findings: observed management paradigms.

Findings within this category have been, at once, the most clear and direct, yet often the most controversial and most fiercely contested among participants in this study. The principle findings with regard to the performance characteristics of development organizations are classified into six areas:

- Time
- Information
- Priorities
- Feedback
- Predictability
- Redundancy

Time

Among the development organizations visited, the vast majority placed time-to-market as their most critical issue. There are many justifiable reasons for this; several of these will become self-evident as we overview other classes of findings. There are synergistic effects, for instance, between time (as expressed in days, weeks, months, etc.) and customer requirements; as time shortens, the stability of requirements is seen to increase; as requirements stabilize, development time shortens. Time (as expressed in man-hours) has been found to closely parallel total development costs, because development is still a labor-intense process, unlike many manufacturing processes, for instance. Moreover, shortening development time is often regarded as a method for developing products which are more closely aligned to customer wants.[12]

Given that there is acknowledged need (by most) for reducing development time, we found it remarkable that development time typically takes 300% to 400% *longer* than necessary. We found engineering centers which only permit their engineers to perform engineering tasks 15% of the time. We found that engineers spend a great amount of time working "backwards" through their processes, trying to resolve prototype problems. We found that "idle" time dominates the *de facto* development schedule, even though engineers and their assistants are overworked and put in many long (read "overtime") hours. We found, at times, that the faster (more *efficiently*) some engineers worked, the slower (less *effective*) the development organizations became.

---

[12] At a few sites, this view was not shared. Rather, management-derived customer needs take precedent over customer wants. The reasoning: "the customer does not *currently* know what he will want by the time our development is complete."

We found *no* organizations that practice "concurrent" engineering in the pure sense; nor do any of them engage in purely serial processing. Rather, all organizations already engage in a mixture of *some serial, some concurrent* processing. The ratio of this mix can *change daily*, depending on current development situations. We also found that some "unsophisticated", small development organizations were able to develop products in far less time than their sophisticated counterparts. Yet, speed does not seem to be a unique feature of size; some large organizations can do it, too.

We found that budget allocation specialists and developers notoriously disagree about the appropriate *timing* for cash flow--and that the overall process pays that price. We found that there are different *time metrics* within and between development organizations; precise (and widely published) "development times" are subject to gross leaps of faith (and, sometimes, revisionist history). In conjunction with various time metrics, we found there often exist "local" process time optimization strategies. Perhaps, by this point, it should not be a surprise that such localized optimization strategies were found not to be integrated into a cohesive system-wide time-reduction strategy. Nonetheless, "time-reduction" was a pressing need among development managers. We shall discuss management tools later in this report.

Information

Information creation, retrieval, transfer, and dissemination are major parts of the "process" of new product development[13]. We observed that there exist an enormous

[13] After just several interviews at the first site, we could see that "taking care" of administrative burdens was a major part of engineers' responsibilities. We did not expect some of the severe impacts that information processing could have on the overall process, however.

number of formal and informal interfaces among development personnel, very few of which are ever documented, much less channeled via "official" information pipelines. This observation was fully expected. What was not expected, however, was some of the characteristic ***effects*** of information on the development process. These characteristics suggest that information can be both helpful *and* detrimental to successful development.

For our study, ***information*** is defined as any elements of correspondence or attempted correspondence, *excluding* actual prototype assemblies or specialized engineering-specific data about such assemblies. Some samples of information include management memoranda, financial reports, conference meetings (or notes about such meetings), policies, regulations, work orders, request forms, schedules, proposals, and evaluations.

As to its helpfulness, information is regarded as necessary for appropriate synchronization of developers. As products become more integrated in their multi-functionality, the need for information transfer between developers grows. Developers have remarked that the information/integration issues rise faster than the number of new features[14]. We call this characteristic ***interface complexity***. The more efficiently information transfer and dissemination tasks can be accomplished, the more effectively developers should integrate the product. In this vein, strong ***information transfer capability*** was expected to be associated with improved decision-making among developers. As long as interface complexity does not rise faster[15], we have seen some

[14] As we discuss in Appendix D, interfacing needs can be expected to rise as a function of the square of the number of features or nodes of the system.

[15] We have found this to be a big "if".The usual situation is for interface complexity to far outpace the increased sophistication of information transfer capability. Question: Has the FAX machine increased or decreased our productivity? Unoquivocal answer from this research: It depends.

indications of this. We suspect this to be the result of more complete understanding of the problem(s) at hand. With improved decision-making, prototype (and final product) quality is also expected to improve. Though this latter aspect was not documented, there is a "warm and fuzzy" feeling in the field that this is the case.

Unfortunately, we identified a number or "cold and prickly" aspects of information, which undermine many of its advantages. First and foremost among these negative aspects is the time associated with processing information. Recall our observation that engineers spend only 15% of their time performing "engineering" tasks. Much of the remaining 85% can be attributed to processing information in one form or another. As far as developers are concerned, such tasks are ***non-value added activities***. Yet, many such activities are helpful for *other* developers to get their engineering tasks done. Thus, there exists a high degree of inter-dependency between developers. ***These dependencies often take the form of information and may extend beyond the confines of the development organization.*** Consider, for example, the multitude of cases in which sub-contractors are utilized: Nothing happens (often as a course of policy) until the work order(s) have been finalized. Delayed processing of work-orders notoriously delays development work.

We observed two interesting characteristics of information transfer between contractors: ***stratification*** and ***unilateralization***.

- ***Stratification*** is the condition where classes of information are screened and segregated for use by specific organizations with an established "need to know." For security reasons, as well as contractually-based financial concerns, this had the effect of limiting the sharing of development data.

- ***Unilateralization*** is the condition where an interface channel is open in one direction, but not the other. It is like having a diode installed on a communication channel between person **A** and person **B**: **A** may communicate to **B**, but **B** cannot communicate to **A**:

A o---------- |----------o **B**

Such behavior was a reflection of concerns over propriety and contractual obligations of companies among each other and with Government. Naturally, with the multi-tier structure of prime contractors, subcontractors, and sub-subcontractors, such issues were more predominant than normally observed within a single development organization.

We witnessed that ***developers are sometimes forced to "wade through" information, just to get back to their "engineering" tasks***. This relates to the findings about priorities, which we shall discuss in the next class of findings. Unfortunately, it is common for developers to have difficulty performing triage on the information presented to them. There is always the "possibility" that a "golden nugget" of information would be found among the babble. Such deciphering of information takes valuable time; it also could lead developers astray. If information assimilation is flawed (or incomplete), then

the developer is forced to make decisions based upon flawed or incomplete premises. If not reconciled quickly, developers could begin working in the wrong development direction.

Given such problems, one would expect developers to interface on a nearly continuous basis (i.e., ***high frequency, low amplitude communication***). Yet, in large development organizations we found that developers met on an "as-needed" basis--when one of them felt that their local project sufficiently impinged upon the "problem space" of another. Aside from the cultural implications of looking like a worry-prone individual in a testosterone-rich environment (the cultural dynamics are very relevant and very interesting components of development which we did not focus upon, but which would make a fascinating study), there is a resistance among some developers to continually ask about their "impinging" effects on others. One contributing reason for this is the earlier stated finding that ***participants carry little understanding of the requirements of other developers' processes***. This is an extremely controversial, yet important point. It was so unbelievable that, upon this discovery, we went back to confirm it over and over again. The result: because developers do not understand fellow developers' processes, they do not know when their actions adversely affect their colleagues. When their conflicting actions do manifest themselves, the parties end up engaging in ***low frequency, high amplitude communication***, with inevitably high error rates. Reconciliation of problems by this point, however, often involves management. Thus, the timing of information may be every bit as important as its content.

A very popular attempt at solving this interface problem has been the advent of product development teams (PDT's). In Appendix E, we outline some interesting dynamics with

respect to PDT formation and growth. A propos information transfer, however, we found that the establishment of PDT's is not a cure-all. There were observed situations where they succeeded, yet there were many situations where the PDT concept was in existence in name only, not really supported by management. At several sites, PDT's were rejected by developers, either because their interfaces were too obscure or hypothetical (i.e., "we're a PDT just in case") or because the team members shared too few personal or professional interests. The most apparent reason for PDT breakdowns, however, was when PDT members felt that their empowerment was a false promise.

Given such findings, it is apparent that information sometimes carries some very heavy baggage. Nonetheless, there are situations where information proves to be extremely valuable. This is especially true in the area of customer requirements. Though there exists a popularized view that product developers perform their tasks in response to well-defined requirements, it is observed that this is the rarity, not the rule. In many large firms (where the distinctions between user and developer are more clear), developers do not necessarily have good understanding of the actual use of the product, nor an adequate sense of all customer requirements. This was most acute in the defense-related developments (because of the sheer number of requirements) and in certain sub-sectors of the automotive industry. At one site, ***developers were actually prohibited from speaking with real users!*** In this particular case, the marketing functions served as filters of such information; by the time requirements were passed to developers, they had been manipulated by administrators to such a degree that there was real doubt about their validity. At most sites, management needs took precedent over customer needs or desires. As related earlier, there exists a very real (and, perhaps, often valid) reasoning that customers do not know their product needs well enough to develop products by. Major

exceptions to these findings were found in the industrial intense development organizations, where products are developed jointly with the customer[16].

One additional set of information-related findings that bear attention regard the use of "information-systems" in development. By and large, developers are well aquatinted with computers[17]. Yet, many sophisticated information systems installed at sites blatantly ignore the needs of developers. It was widely apparent that system administrators were more interested in the efficiencies of their computer systems than the effectiveness of the engineers. Depending upon the relative power between administrators and engineers, one of two scenarios resulted from this condition:

1. Engineers ignored the "useless" systems when they wasted their time, and bypassed them with their own personalized local procedures. Naturally, this threw the "master" computerized indices out of kilter with reality, and frustrated system administrators.

2. The computer systems were mandated by management, with strict guidelines for "appropriate" behavior of engineers, thus frustrating those engineers whose tasks necessitated better performance than the system could deliver.

---

[16] For example, we observed an exceptionally close relationship between a piston design/manufacturing company and engine manufacturer clients. Such close relationships, which in one case had been on-going for nearly one hundred years, often resulted in designs that far exceeded current design requirements on the first iteration of development.

[17] The reader should not that we have deliberately avoided discussing our many observations on the use of automated "engineering tools" (e.g., CAD/CAM/CASE, CMM, FEA, CFD, etc.). This is because the findings reveal that other problems dwarf the observed problems of automated engineering inefficiencies. There are still many gains to be had, but these are being resolved more and more by specialists in this area.

In some cases, alternative practices (i.e., local databases or specialized software) were banned. At one of the largest development organizations, whose parent had just purchased a huge systems integration company, an undercover enforcement strategy was employed to try and force engineers into compliance. Given such practices, it was not surprising to find that many developers in the trenches perceive information technology as an administrative "club" to reduce their autonomy. Engineers divulged a number of schemes they have employed to circumvent such obstacles; some of them would be considered outright innovative in nature. Yet, it cannot be considered funny to see a very expensive, sophisticated technical information center (library, but don't use that word around the site manager, please!) whose primary uses by engineers were the periodical browsing room and the photocopy machine. Worse, in some (other) companies, there were many engineers who did not even know that a "engineer-friendly" technical information library (or any library, for that matter) existed at their facility.

Priorities

In our discussion of information characteristics in development, we alluded to priorities which exist to resolve competition for engineers' time. Though a very simple concept, the establishment of appropriate priorities has been found to be very difficult. Should a developer ignore a memorandum to save his own time? Or should he read it, for its content may help save his (or somebody's) time? Should engineers go to more or fewer meetings? How many should go? Which ones? Who should get copies of a newly revised drawing...who should just be told about it...who should not be told anything? Why should developers fill out this or that form? Who needs it?..... the number of local decisions and actions made by developers and managers may seen endless. Yet, they must, for better or worse, all finally result in something resembling a producible product.

There exists a noticeable problem of setting compatible priorities among participants: ***"Insignificant" information to some developers or managers were perceived as critical for others.*** Because they must assign some priorities (there never seems to be enough time to process everything!) to their information processing, they have to determine how important certain information is to them *and* others.

It was observed, however, that no universally acceptable, coordinated priority strategies exist in development organizations. Rather, judgements are made on a decentralized basis, based upon developers individual knowledge or common sense understanding of the needs of other developers. As we described earlier, though, developers typically do not understand the processes of most of their fellow developers. Thus, their abilities to make the right decision (and use the *appropriate* level of urgency) are compromised.

There was an observed tendency for information's priority level to change as a function of its source and destination in the product life cycle. Fundamentally, information regarding "downstream" activities had higher priority than "upstream" information. The reasoning for this was that ***products closer to production had more urgency***, for production start dates were impending--"better to delay future products, than compromise immediate products." Otherwise, one could conclude that the "squeaky wheel" strategy rules--the loudest, most prevalent voice gets the most attention.

Process Feedback

During the functional modeling, and subsequent isolated process modeling[18] efforts, it was documented that the process of development is not conveniently linear in nature, but rather contains innumerable process feedback loops, at times ***causing the process itself to change with time***. This is in stark contrast to the convenient characterization of product development as a stable, manufacturing-like (unidirectional) process. We observed that *every* development process is non-linear in nature. Such feedback can be characterized as a rework process. Moreover, it was observed that managers much prefer to think[19] of the process as a linear process, for that characterization suits their tools better. Rework processes were treated by some managers as special case situations. Yet, we found that ***such rework processes dominate development time***. Furthermore, since rework can completely change the criticality of impending "downstream" functions, the process bottleneck can change. Utilizing CPM-type techniques is difficult enough for management of large, complicated systems. Now, we have evidence that the process is contingent and non-linear, resulting in ***changing critical paths***. Given these conditions, we conclude that many long-trusted management tools being used are inappropriate for new product development.

Non-linear processes are also known as complex processes. Thus, we distinguish between the terms ***complex*** and ***complicated***. Complicated merely implies that there are many

---

[18] A more detailed description of functional (architecture) modeling and process (flow) modeling is given in Appendix G.

[19] When modeling conducted by developers showed highly ***complicated*** and ***complex*** processes, some managers asserted that this was merely a demonstration of many linear processes overlaid upon one another. Regardless of the level of explanation of the diagram by developers, managers often continue to retain this convenient view.

components to the process. For instance, a small development firm we examined only had eight functions. This would not be considered a complicated system. Contrast this with the development systems of a major defense project, where there are thousands of functions. Yet, both of these systems could be complex, through the mere existence of one or more feedback loops.

Process Predictability

Experts of non-linear dynamics know that non-linear systems can behave in very strange, unpredictable ways, depending upon specific structures and parameters of the system. The field modeling efforts and interviews revealed that process feedback loops could occur at just about *any* time in the development timetable. Since the effective severity of these loops was dependent upon the amount of unresolved latent technical or communication problems between developers (and sometimes their bosses[20]), which was itself unpredictable, we have identified new product development as ***a non-linear system with unpredictably variable feedback***. Unfortunately, a priori predictability of such systems is well beyond current technical feasibility.

Yet, some companies repeatedly are able to develop products in approximately the same (long) time frame. Why is this? In our interviews, it became apparent that firms do not perform the same tasks during each development. Rather, there is a self-leveling effect with regards to the number and nature of new product features, given the amount of time

[20] We have focused here on the problems of ***internally generated*** feedback loops. Just as severe, however, can be those loops which are induced by ***external*** forces. One obvious example of this is a change in government regulations. Another prevalent example is when an executive reviews a prototype and declares that "he doesn't like it", but doesn't specify what or why he doesn't like. This latter example has been observed to send development organizations into multiple simultaneous loops, as developers go through "guessing games" as to what they must change.

available. Thus, product sophistication or content (and, thus, the efforts necessary to incorporate such) is expanded or reduced to accommodate pre-defined production start schedules. In development programs which do not have strict (closed) time-frames, but rather a pre-defined set of requirements, developers have shown us that their development time is, in fact, highly unpredictable. For industries in which consistent model release dates are not the norm (currently, the computer software industry comes to mind), product release intervals from a single company may vary widely.

Such unpredictability can play havoc on the marketability of the product being developed. As we discuss in Appendix F, variation in market response time can be a triple edge sword, two edges of which are turned *toward* you:

- ***Late product releases*** can result in lost market share, in the presence of quick, competent competitors;
- ***Early product releases*** can result in very costly efforts to educate the market on your new product. Once you've spent your money, "follower" competitors can jump in and capture much of your market;
- If the ***product release date matches market timing needs***, and your marketing efforts are sufficient, then you can exploit the emerging market. In the absence of strong competition, acceptable profitability can be sustained to more than recover total development costs.

In the current environment of shortening product life cycles (and thus shorter windows of opportunity), release date volatility can be expected to be less acceptable. Thus, reduced

absolute development time *and* more precision in product release date accuracy must be considered performance attributes of development enterprises of the future.

We also observed problems of resource allocation throughout the development process. Excess resources could be found in non-critical parts of the process and inadequate resources were available in critical, "crisis" parts of the process. From an observer's point of view, it was common to see management chasing this problem. From engineers' perspectives, workloads were intermittent--periods of sheer chaos followed by periods of calm[21]. Since the process was always in some degree of flux, this is not surprising.

Downstream functions (those considered "closer" to production) had more consistent workloads, for their conformance to production schedules was more firm. This gave the effect of increasing the pressure on upstream functions to expedite, so that downstream functions (also considered higher priority functions) would not be delayed. Coupled with the fact that upstream functions were less routine in their tasks, this forced workload variance on upstream functions. This can make accurate resource requirement forecasting even more difficult for management. For largely this reason, ***contract labor is widely evident*** in modern development organizations.

---

[21] We observed some engineering tactics to smooth this varying workload. One such tactic was to increase one's apparant activity level (i.e., "look busy"), by teaming with others on some "important" task.. By seeming busy, an engineer would (he expected) not be assigned some additional task. On the other hand, if an appealing task was being assigned, it was remarkable how quickly some "busy" engineers could make themselves available!

Redundancy

It is common for participants to conduct similar tasks independently of one another. Such duplication of effort is often bemoaned by management as an efficiency problem, yet is treated by many developers as necessary to expedite their local processes.

Based upon our investigations, we find both viewpoints to be true. Many functional redundancies are, *ex post facto,* easily noticeable areas for potential improvement. This was apparent when we documented development tasks and categorized them by functional similarity. There are problems, however, with merely concluding that all redundant tasks should be eliminated. We observed that many redundant tasks were performed more efficiently (i.e., quickly or inexpensively) *and* effectively (i.e., properly) when left decentralized. There can exist major cost, time, and quality deficiencies when one must transfer all relevant information to some single specialist or "department" for processing, particularly if local developers are already adept at the task.

Even if developers are aware of duplicate efforts, it is not necessarily in their best interest to eliminate such duplication. One reason for this is that ***tools for synchronization*** between such efforts do not currently exist. A developer's need to perform a task may be immediate; he cannot afford to wait and see if others are also in current need of the same task. Also, it was observed that ***professional and friendly competition between developers*** could necessitate isolated, albeit duplicate activities. For instance, it is not unusual to see two or more developers working in the same room on competing facets of design, only one of which will be approved by management. In some installations, "joining" of duplicate tasks is considered a ***security*** risk: it is considered better to suffer

some inefficiency than potentially sacrifice the veil of secrecy of emerging projects[22]. This has been traditionally true within the defense industry. For the AMC, for example, the need for redundancy between nationally distributed facilities is especially apparent. If one facility were to become incapacitated, other sites must be able to pick-up the slack. Lately, product security issues have become very pronounced in commercial development organizations, as well.

Perhaps the most prevalent reason for functional redundancy, however, relates to the ***need for process autonomy***. There is an undeniable sense of pride and self-reliance which exist in many engineers. In fact, such characteristics are part of the basis for the establishment of elite development teams and product platforms within larger development organizations.

One additional redundancy-related finding is that there exists a high degree of ***technical memory loss*** in development organizations. This forces subsequent development projects to "re-invent the wheel"--to perform redundant tasks *over time*. We found that problems of technical memory were not limited to product attributes, but could also extend to process attributes and methodological recall. Unfortunately, the latter two areas are not generally well documented. We found the number one retrieval source for this type of information to be "old-timer" engineers, who had seen a wide variety of situations and solutions over the years. Unfortunately, we have observed that such extensive knowledge

[22] Issues of security are rampant in development organization. Quite often, developers are forbidden from entering high-security areas of their own development facilities.

bases are fading from the development scene[23]. Largely, the scope of this study could not have been conducted without the insight of such experienced engineers.

Observed Management Paradigms

This third category of findings reveals some characteristics of the management practices utilized to control product development organizations. Overall, we observed that management's tasks and responsibilities are neither enviable nor terribly rewarding. Frustration levels run high in new product development, as great expectations are rarely met. Managers of new development projects resolve to learn from past projects, to keep from making the same mistakes. Nevertheless, many projects and their managers degenerate into modes similar to their predecessors. As a result of this research, we are beginning to understand why this is so. Underlying these observations is the realization that managers are often paradigm-based in their decision-making. These findings are classified into five areas:

- Scope
- Efficiency vs. Effectiveness
- Decision Authority and Responsibility
- Tools and Measures
- Change Methods

[23] During recent cost-cutting measures of some organizations we visited, experienced engineers were considered high-cost resources, who could be adequately replaced by younger, lower-wage engineers.

Scope (where they look)

Scope may be defined as a weighted spectrum of vision. In the vein of development management, this is an indication of how much of the process one sees at any given time. As we observed earlier, it is typical for developers to be focused on their aspect of the process, with little knowledge of the effects of their processes on others. It was also discovered that many managers engage in similar behavior. Not only did they not understand the structure and interdependence of other parts of the process, they often carried delusions about the processes of ***their own*** developers. This was initially discovered in our process documentation efforts, for which we enlisted the efforts of developers. When the developers' final characterization of their own processes[24] were presented to some managers, the managers could not accept them. The reason: their views differed from those of their own developers. No amount of persuasion by developers could convince these managers of the true nature of their own processes! Though far less vehement in most situations, it was typically true for managers to carry outdated or ill-founded understanding of their processes.

When developers began discussing their findings and insights from examining ***other processes***, managers could be observed to be both angry and perplexed. Their anger stemmed from the notion that there was no need to "worry about other peoples' problems" and doing so was just a waste of their time. Their curiosity, however, usually got the best of them; though not an official inquiry, they would want to know (off the record) what their developers had found.

---

[24] Such characterizations were typically arrived at by consensus of the team, not majority election.

Such scenarios seem to stem from two sources: the ***need for recognition*** and ***insufficient environmental scanning***. We observed that development managers tend to be very interested in streamlining the processes of their developers (and, of course, getting credit for such improvement). Naturally, this "streamlining" has proven very difficult to do in a significant way. Thus, many managers have developed an attitude that every improvement, no matter how small, will result in a positive contribution to system performance. This philosophy is virtually the definition of the ***reductionist approach*** to system improvement: improve each part of the system, and the whole system will improve. As stated earlier, we have seen scenarios where local process "improvements" actually undermine the effectiveness of other processes. If a manager does not know (or care) about this, however, he could be systematically disabling the organization, just by making his local processes look good.

We observed many cases where managers (and developers) lamented about the need for the organization to engage in some new method or paradigm which they had just learned about. Upon conducting a state-of-the-art review, it was not unusual to find local areas of the organization which ***had already been utilizing*** such methods. Because the original manager(s) did not know this beforehand, and much ado had already been made about "their" new methods, such scenarios could be sources of embarrassment.

Most importantly, however, ***few managers seem willing to scan the entire organization*** for system-wide effects of their developers' actions. Those who did so had local processes which better complemented other processes within the organization. Interestingly, we found that ***many executives acknowledged the problems of feedback*** among local

processes. It is hypothesized that executives have a vision spectrum more weighted towards global orientation than local orientation.

Efficiency vs. Effectiveness (how they manage)

The problems of insufficient scope lead directly into the findings of this section. Succinctly, management had a readily observed bias towards improving the ***efficiency*** of their operations, and gave inadequate consideration about the ***effectiveness*** of their organization. Given the incentive structures and existing paradigms of how the process worked, this should not, in retrospect, come as a surprise.

What was surprising, however, was that there was an observed ***reluctance to change from this efficiency orientation***, even after more realistic views of the process became available. For instance, rework processes were considered to be the exception to the process, not the rule. Interfacing processes between functions were assumed to be part of the local function's responsibility, not conducive to consideration by some management. It became apparent that many managers live by their tools. Take away their tools, and they are unfamiliar territory. As we shall discuss later, their tools are often manufacturing-derived, not necessarily appropriate for development processes.

Decision Authority and Responsibility (who gets to "manage" crystallization?)

We remarked earlier that experienced developers posses a wealth of information and insight not generally observed in younger developers. Their insight comes from experience, to be sure. Their variety of experiences, however, seems to dominate in

importance over any particular individual experience. Perhaps in no areas are such "old-timers" as useful as in times of development crisis. When developers are forced to "bridge" gaps in requirements, good judgement about the process and about the customer or end-user are important. Experienced developers were useful for this task. When budget negotiations with one's own management were an issue, experienced developers could offer salient advice. Surprisingly, when new paradigms were required to solve a technical problem, the experienced developers were often better suited to the task than their younger colleagues. Perhaps there is truth, after all, to the old adage that "the young know everything, the middle age reject everything, and the elderly accept anything!"

We have concluded that such favorable attributes are not so much a function of experience, per se, but rather an ***attitude*** which often is associated with experience. This is the no-holds barred, "war-dog" attitude, which says that anything can be accomplished, if one can get into "war-time" mode. Other characterizations of this concept have been illustrated with slogans such as "Give engineering back to the engineers", "Death to Administravia", and "Rambo Engineering". ***When in this mode, which many projects go through near the end of the development process, administrative (and most non-value added information) activities must step aside or be sidestepped.***

There are few managerial tools for managing war-time mode, which often removes management from the development equation[25]. In fact, many managers have stated that they much prefer smooth, gradual development, so they can better manage it. This,

---

[25] At one of the US Army projects we oversaw, two very different IDEF functional representations of the same organization were developed--the "war-time" model and the "peace-time" model. In the "war-time" model, administrative functions were all but eliminated altogether.

however, is an anathema to highly integrative engineers, who see development as a situationally-driven process.

What we have observed in "war-time" situations is the systematic elimination of excess, non-helpful, information. This forces developers to engage in decentralized decision-making, sometimes thinking "off the page" to get problems resolved. Underlying such processes, however, is what we believe is a very important shift in orientation and priority of objectives. There is far less waffling over minute aspects of objectives and more emphasis on a core set of development needs. Supplementing such a clear core, often, are budget allocations for engineering tasks that would be unthinkable in "peace-time" mode. When budget restrictions are removed (i.e., "do what's needed, whatever the cost"), developers are more free to perform engineering, not waded down by administration or politicking. They are free to self-allocate funds, according to *their* priorities, not an administrator who doesn't understand their processes. In short, developers gain both authority and responsibility for development, traits which are rare in "peace-time" mode.

Significantly, out of a melee of prior confusion, disjointed processing, and perhaps "over-management", the **development tasks suddenly get done--promptly.** I call this phenomenon ***development crystallization***.

Tools and Measures

Several times in this report, we have commented on the inappropriate nature of many managerial tools used for controlling new product development. This is an important, though controversial point among many managers. And why not? If managers depend on their tools, then taking such instruments away leaves them with an empty toolbox. They

would often be aimless in an unstructured environment. *De facto,* many managers felt that their tools were important enablers for the process to continue. They would not like to hear that the development process often proceeds ***despite*** their tools.

A sampling of ***tools*** used by development managers includes Gantt charts, PERT/CPM network analysis, Resource Utilization Histograms, Work Breakdown Structures, various Materials Requirements Planning methods, basic Economic Order Quantity models, Activity-based cost accounting, organizationally-based cost accounting, ad hoc local process flow models, high-level linear process-flow models, and element-attribute-relation data modeling. We have witnessed proposals for JIT (KANBAN) and "Lean Production" methods to be tried in development. To better facilitate such tools, managers have a variety of commercial and locally-developed software tools at their disposal, which run on various computer platforms. These platforms range from portable and desktop personal computers, to workstations, to mainframes[26].

In addition, we have observed a slew of locally derived time, cost, and quality ***metrics*** among development management. The integration of such metrics to one another, and across locales, however, often left something to be desired. There commonly exist multiple objectives among development management. Yet, a specific metric which satisfies one local objective is not generally compatible with those of other objectives. For example, development costs for a specific department are routinely gauged using a simplified surrogate: headcount. If managers can reduce headcount and still get their tasks accomplished, they look considerably more efficient. One method for

[26] I have yet to observe Supercomputers being used for/by management, though they are utilized for a variety of specific design tasks by engineers and technicians.

accomplishing this, particularly when downsizing is prevalent within the organization, is to replace employees with contract labor, who are not counted as part of the official headcount. Thus, managers get the same tasks done at lower "departmental cost." Yet, one view of the development costs (in dollars) shows that contracted expenses have increased. Depending upon the need for resource level flexibility and employee fringe costs, this transfer of cost could result in higher overall organizational costs, for contracted labor required considerably higher compensation[27]. We consider other examples of convenient, though inappropriate use of time, cost, and quality measures in Appendix F.

Many of the management tools and measures utilized by managers are ***manufacturing-based***. This can be expected, given that many engineers and managers are trained on manufacturing-based methodologies, and manufacturing gets the most emphasis in many organizations. We have observed that manufacturing-based tools have been on a recent upswing in development organizations, particularly as the need for manufacturing integration in development has become more prevalent. Much discussion is made about such topics as bottlenecks in development, moving bottlenecks, critical paths of development processes, new automated engineering tools, better information control systems (often called business systems), new cost accounting methods, more simultaneous engineering (i.e., parallel processing) and more. Fundamentally, however, it has become acutely apparent that ***development processes do not resemble most manufacturing processes***. The process is probably better though of as a non-linear

---

[27] In several example which we witnessed, the contractor was the former employee, hired back at higher rates (including fringe). In several cases, we encountered such contracters who had retained that status for over five years.

electronics circuit, with both forward and reverse feedback loops. There is little or no discussion about the ***dynamically changing***, sometimes ***instable*** nature of the development process. This is a serious deficiency, which is not overcome with more vehement declarations of development as being like manufacturing. It is not.

Change Methods

We observed that many managers are more comfortable with evolutionary approaches to change than revolutionary approaches. We have also observed that such evolutionary, incremental approaches work well in some manufacturing and other linear, reductionist systems. Since new product development does not fall into that category, however, it is now understandable why incremental improvements can push the process into localized valleys of optimization, resulting in sub-optimization of the overall process.

It was also observed that personal knowledge bases and beliefs could interfere with execution of even well-thought changes. If a manager does engage in a more holistic view of the overall organizational processes, he may find that appropriate changes do not look appropriate to locally-oriented individuals. Thus, there is a very real problem with ***low commonalty of focus within the development organization***.

Commonalty of focus was a feature of successful product development teams (PDT's). If they engaged in mutual reinforcement of the importance of their common goal, then local optimizations could be seen as having lesser importance. It is rare for such a dedicated team to be established and hold together throughout the development process, however. Personal temptations grow as the operations of PDT's become routine and administrative

in nature. As engineers at one site often remarked, "We don't build products, we build careers." As an engineering manager who carries the same orientation, how can you be expected to manage and change this?

Per the account of several developers in a large-scale, multi-contractor military aircraft project, the integration problem was so bad that a joint meeting of *all* the developers was conducted on one weekend, away from all of the "home" development bases:

> Located in a large aircraft hangar, the specifications and progress status of the various sub-assemblies were reviewed by all. By seeing how each of these previously isolated sub-projects related to one another, developers generated many new ideas and solutions to problems, many of which they were previously unaware of. In the course of one weekend, some development problems which had been ongoing for several years were rectified. Developers from different sub-contractor organizations, from opposite ends of the country, met each other for the first time. The meeting was so successful that the temporary site, with all its process-progress charts and component prototype assemblies became a permanent, albeit high-security, fixture of the overall development project, as an integration tool for decentralized developers.

We discuss the integration problem at more length in Appendix E.

Appendix D: Summary of the IDEF0 modeling methodology

The IDEF0 (pronounced "I-deaf-zero") methodology was utilized early in our field studies to help document, verify, and understand the operating environment of several engineering organizations, particularly as they engage in new product development. As a rigorous function modeling structure, IDEF0 is an excellent modeling framework with which to document large-scale systems. It encompasses functional identification, acquiring information on the relations between all functions, graphical representation, textual descriptions, establishment of a terminology glossary, and a formal review cycle. Because it is unlike many other diagramming techniques, it may prove useful to consider the background, structure, development, and use of IDEF0 models. Once one has a grasp of the fundamentals of the methodology, future encounters with such models can be insightful, rather than intimidating experiences.

Background

IDEF0 is a subset of the IDEF methodologies, which were developed[28] as part of the ICAM (**I**ntegrated **C**omputer-**A**ided **M**anufacturing) Program at Wright-Patterson Air Force Base during the late-1970's, in an effort to help modularize and standardize US Aerospace manufacturing processes. The other major IDEF methodology[29] in

---

[28] Technically, IDEF0 (properly printed as $IDEF_0$) is a derivative of the Structured Analysis and Design Technique (SADT), which itself was derived from Dr. Hori's "Human-directed Activity Cell Model" work at IITRI. Reference USAF $IDEF_0$ Function Modeling Manual, Report # UM 110231100, June 1981.

[29] For acronym enthusiasts, which military programs seem to be filled with, IDEF is a two-level acronym standing for **ICAM Definition**.

widespread use today is IDEF1/IDEF1x modeling, which is used to document and define data structures for database and large-scale information system design. Because we do not directly use IDEF1/IDEF1x models in this report, they are not addressed here. In addition, there was an IDEF2 methodology, which was intended to demonstrate the dynamic behavior of functions, information, and resources in the manufacturing environment. With the rise of commercial manufacturing simulation tools, this method is not in current use.

Though IDEF is a public domain methodology; there are a variety of proprietary IDEF-based tools and associated methodologies in use today in both government and industry, to facilitate its use in large organizations. For much of this study, I used the "IDEFine" semi-automated tools from Wizdom Systems, Naperville, IL. As a facilitator of several new applications of the methodology (of which new product development was one example), I helped refine the capabilities of such tools, which enabled us to see previously undocumented phenomena. Due to the sheer number of elements, terminologies, and process structures typically identified during any serious IDEF project, I strongly suggest using some automated tools such as these to assist the process[30].

As we outlined in our research, however, current manufacturing methods (of which simulation methods are one class of examples) do not capture some of the more interesting ***dynamics*** which occur in product development. Given that we have

---

[30] One may well consider that IDEF methods have been around for nearly twenty years, but have really only seen major use in the past 6-7 years. This may be attributed to the rapid development and utilization of the personal computer. Prior to the availability of computerized documentation, IDEF-oriented projects were manually driven, which severely limited the scope, scale, and speed of such projects.

documented development with IDEF0 is several organizations, it should only be a matter of time until special non-manufacturing oriented dynamic tools become available. The current research is intended to forge a step in this direction.

Structure

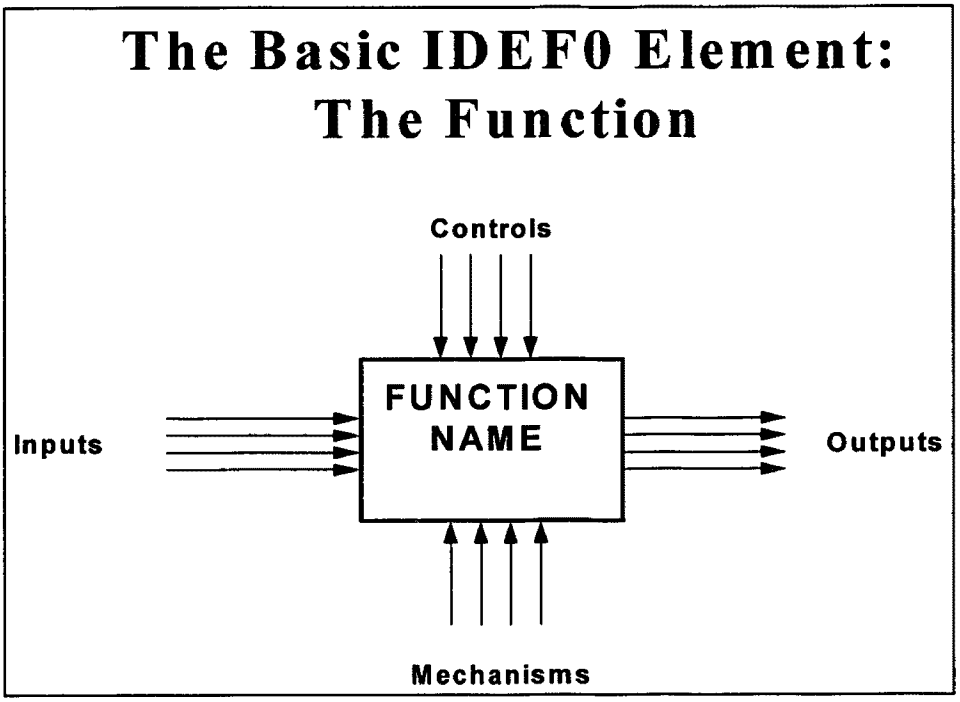
There are several structural components which are useful for any IDEF0 model reader to understand[31]. We address the following concepts: ***functions, interfaces, diagrams, decomposition*** and complete ***model kits***.

The fundamental element to the IDEF0 modeling methodology is the ***function***. Functions are representations of *activities* performed within a system, whether by manual or automated means. Functions are distinguished from organizational units within an organization. The primary difference is that functions, in and of themselves, only represent activity, not the organizing body which supports such activity, nor even the people who conduct functions. In IDEF, such supporting characteristics belong to a very important component of the IDEF model, called mechanisms, which we shall discuss shortly. Functions are demonstrated in an IDEF0 model as a rectangular box. Refer to Exhibit C.1.

---

[31] There are many components and nuances of the IDEF0 modeling structure which we shall ignore in our brief overview, for they are more important to professional developers of models than to general readers.

EXHIBIT C.1



A unique verb phrase is contained within the box, and is referred to as the ***function name***. The choice of function name is a very important, precise exercise, for the name represents all the activities which are contained *within* this box. Some valid function names include: DEFINE PRODUCT REQUIREMENTS, CONDUCT RELIABILITY TESTS, REVIEW WORK ORDERS, DEVELOP PRODUCTION PLANS, etc.

Notice the four sets of arrows that enter and exit the function. Each of these classes of arrows are quite distinct in nature from one another. ***Inputs*** (coming in from the left) are those interfaces which are changed as a result of the function. Examples include raw materials, WIP, previous design data, etc. ***Controls*** (entering from the top) are those factors which constrain the function's performance. Examples include things like policies, laws/regulations, budget limits, schedules, etc. ***Outputs*** (exiting to the right) are the

immediate results or products of the function. Examples may include such wide ranging items as finished goods, inspected raw materials (if one's function was the inspection function), modified schedules, etc. ***Mechanisms*** (which enter a function from the bottom) are those entities or facilities which enable the performance of the function. Typically, mechanisms include facilities, equipment, personnel, money, and so forth. Collectively, all arrows are known as ***interfaces***, for their role is to connect a particular function with other functions and with the "outside world." By rule, every arrow in the model is marked by name, delineated with the noun which such an interface represents.

To look at one function all by itself is usually less than interesting. Thus, the IDEF0 methodology permits the display of several functions at once. When a small collection of functions, as well as their interfaces, have been depicted on a single page, one has the basic formation of an IDEF0 ***diagram***. To keep diagrams from being too simple to gain insight, the IDEF0 methodology requires *at least three* functions to be displayed per diagram. To keep from getting unwieldy, a *maximum of six* functions per diagram are permitted. This is in accordance with research such as George Miller's Magical Number Seven Plus or Minus Two" (See Miller (1956)), which reveals that the human mind typically has difficulty juggling more than several items at once. For similar reasons, interfaces are generally held (in practice) to an upper limit of six per box side, though this is not a strict requirement of the methodology. Functions, of equal size, are drawn along a diagonal (from upper-left to lower-right) path, to permit adequate room for interfaces between functions.

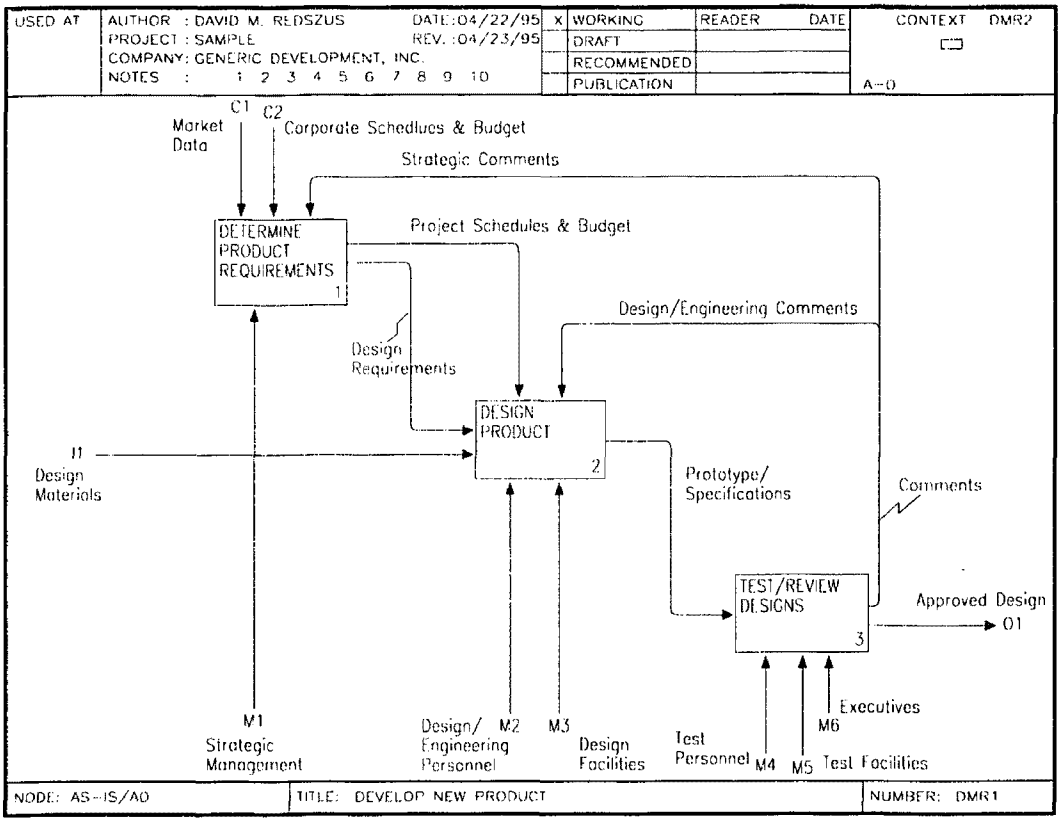
Neither time nor sequence dictate the structure of an IDEF0 diagram. Rather, the diagram is designed as an illustration of the relationship between functions. If there is a general

sequence to a functional relationship, however, it is standard practice to demonstrate this as a left-to-right process.

Interfaces between functions on a diagram are known as ***internal arrows***. Due to the specific relationship which may exist between functions, the output of one function may be considered as an input, control, or mechanism of another function. In fact, functions to the right of its colleague functions may affect these other functions. When this occurs, the resulting interface is known as ***feedback***. Refer to Exhibit C.2.

EXHIBIT C.2.

A Typical IDEF0 diagram



Here, "Comments" (leaving from the TEST/REVIEW DESIGNS function) are fed back to the other functions. Such comments may be characterized as "Design/Engineering Comments" (those going to the DESIGN PRODUCT function) and/or as "Strategic Comments" (which are sent to the DETERMINE PRODUCT REQUIREMENTS function). Depending upon the nature of such comments, functional cycling may continue for several iterations, until the design is adequate for approval.

Functions are not performed in isolation. They are usually associated with other similar-types of functions. Thus, in a model, functions are ***grouped according to their functional similarity***. In assessing this similarity, one may find that similar functions can be found *across* the organization. As a result, functional structures are, not typically congruent with classical organization charts. This attribute can be very helpful in determining the degree of functional *redundancy* in an organization: an accurate model will display the various parties which perform that function as separate mechanisms to the same function.

Further, in an IDEF0 model, ***each function is only displayed once***. This requires a model developer to carefully think about the overall structure of the model before engaging in detailed modeling. To help define this structure, an IDEF0 model progresses *from the more general to the more specific*. This is accomplished by developing multiple diagrams. each a more detailed, more focused subset of its predecessor. The first diagram is generally a simple single box with the general name of the activity being considered as the totality of the system of concern, with general external relations (illustrated with straight arrows) entering and leaving this box. As this overall function is gradually decomposed into its constituent sub-functions (to be found on subsequent diagrams), the terminology of both activities and relations become more specific. Further decompositions become more and more specific, until an adequate level of detail has been reached to obtain data and begin analysis. In this way both ***breadth*** and ***depth*** of a system may be obtained in an organized fashion. It is useful to think of an IDEF0 hierarchy of functions as a family-like structure. Higher level diagrams are referred to as "parent" diagrams; lower level diagrams are "child" diagrams. In this scheme, all diagrams (except for the very highest and very lowest) are *both* parent and child diagrams. Refer to Exhibit C.3.

EXHIBIT C.3.

IDEF0 Decomposition

A-0 Diagram

Parent of A0 diagram

A0 Diagram

Child of A-0 diagram

Parent of A1, A2, A3, and A4 diagrams

A3 Diagram

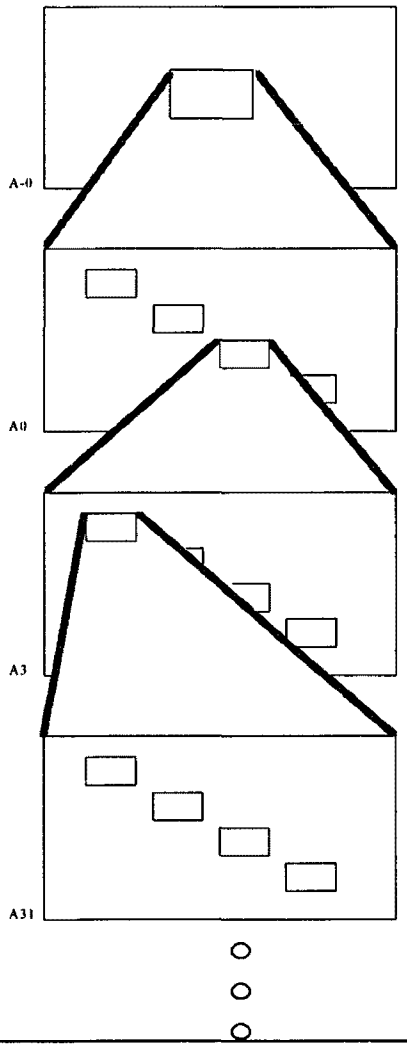
Child of A0

Parent of A31, A32, A33, and A34

A31 Diagram

Child of A3

Parent of A311, A312, A313, and A314



More General

More Specific

A ***standardized numbering system*** has been established to organize IDEF0 diagrams. Each diagram is assigned its own unique serial number. These serial numbers begin with the letter 'A' (activity), and are followed by one or more numeric digits. The *number of digits* and the *value of each digit* tell the reader precisely where in the functional hierarchy one is looking. The number of digits is an indication of how many steps into the hierarchy one is peering. The value of each digit indicates the path of branches one had to traverse to get to this diagram. Often, the 'A0' ("A Zero") diagram is considered the starting diagram of interest, for it is the only diagram that at once shows the full breadth of the system under consideration, and yet had enough resolution to reveal some goings-on (relations) between the major sub-functions. The 'Ax' diagram (where x is a whole number greater than 0) demonstrates the major workings within the x-th function, just "one level below" the A0 level.

Given this, it is natural to see that a single step into the third function on the A0 diagram would result in an 'A3' view...looking into the second function would result in an 'A2' perspective. Once one is on a functional branch, the left hand digits in the serial number remain constant; deeper levels result in new numbers being stacked to the right of the existing number. Thus, the A31 diagram in a model is a second tier diagram, within the 1st sub-function of the third function of the A0 diagram (refer to Exhibit C.3.).

After some experience, it became apparent that it was also useful to look **up** by one level from the A0, to establish some context for the system. But what should such a diagram be called? Presumably, a mathematical limit-oriented individual realized that this level was too similar to be given a whole new number, so they named it the A-0 ("A minus zero")

to reflect the ***context diagram***. This diagram is the only diagram in a model that contains only one function[32].

On a given diagram, arrows may pass *between* functions, emanate from outside the diagram, and/or leave the diagram. Those arrows which originate or are destined outside a particular diagram are called ***boundary arrows***. To help trace such arrows such arrows, ***ICOM codes*** are utilized. These codes are two-digit codes (a letter, I, C, O, or M, followed by a number) which tell the reader which parent arrow (the arrow as it enters or leaves the diagram, when viewed from one higher level) is the parent of the one in question. For instance, if a boundary arrow carries an M3 code, then that arrow is all or part of the third arrow entering the parent box on its mechanism side. ICOM numbering is conducted on a per-side basis; input and output arrows are counted from the top down, while control and mechanism arrows are counted from left to right. In this way, each diagram is related to its immediate parent diagram[33].

Individual functions and the predominant relations between such functions are the key elements which are documented in an IDEF0 model. A complete model (usually referred to as a ***model kit***) is a highly-structured hierarchical set of ***diagrams***, with ***textual descriptions*** for each diagram, and a model-wide ***glossary***. Each diagram provides visual stimulus for the relationship among functions depicted on that diagram, as well as a clue

---

[32] For some analyses, we have taken this direction on step further, by creating an A-1 diagram, to see the interfaces of outside influences, as well as their functions. Such upward model integration can be a convenient way to see the relationships (if any) between models of two separate organizations. With such a view, one can make some judgement about how isolated various systems really are.

[33] ICOM codes only relate a particular diagrams boundary arrows to its *immediate* parent diagram. Thus, an arrow which traverses from diagram to diagram, being bundled or unbundled, may very likely change its ICOM designations on different diagrams.

to the degree with which each function relates to functions *not* on the diagram. The textual descriptions highlight and clarify important details of the functional relationships which are not immediately clear on the diagram; they are not mere substitutes for the visual diagram. The model-wide glossary contains definitions for *every* term in the model, including functions, relations between functions, inputs, controls, outputs, mechanisms, and other terms which help put the functions and their relations in better perspective.

Development of IDEF0 models

For all sites in which IDEF0 models were developed in this study, a rigorous methodology was employed. It was important to adhere to a set of modeling conventions, so that development could proceed without excessive rework, models could be consistent in detail and context, and (most importantly) the models could be accurate facsimiles of the real systems under analysis.

Major stages of this documenting methodology included ***team-formation, training, context formulation, preliminary modeling, data collection, detailed modeling,*** and ***validation***. We shall briefly review these steps here.

> ***Team formation***: Before collecting any data about the system of study, it is important to obtain some rudimentary knowledge from a number of participants *within* the system. Particularly in large-scale systems, such as in our study, this is done by establishing an AS-IS documentation team, consisting primarily of such participants. During this stage, preliminary system "borders" are established, to bound the analysis problem. This is typically done by a few strategic managers.

As such bounds become more refined, the team size grows in response to the need for certain system specialists. ***The establishment of a diverse, objective team is critical to the success of the modeling task.***

Training: Before any modeling can be conducted, each member of the team must be familiar with the IDEF0 modeling methodology. This "training" may be formal or informal (largely dependent upon the size and background of the evolving team), but must be done. Such orientation is not limited to the particular rules of IDEF, but also includes some consensus decision-making about nomenclature, system definition, and some modeling conventions[34]. During this orientation, agreement is made about people assignments and meeting conventions (e.g., meet every Mon-Wed-Fri at 8:00 AM). Though this may seem highly rigorous, it is important, for the modeling task can become very complicated very quickly, particularly when decentralized activities are being modeled[35].

Context/Viewpoint/Purpose formulation: Before any modeling begins, the team must agree upon the ***context***, ***viewpoint*** , and ***purpose*** of the model being created. This means refining the bounds of the system, determining what activities are/are not going to be covered, as well as the perspective from which the model is being

---

[34] Such conventions may be as mundane as "Functions will be designated in UPPER CASE, Interfaces will be denoted with Mixed Case", but also establishes some important team member roles. For instance, there should normally be only one librarian (collator of all collected data), one glossary manager (hopefully, someone with good linguistic skills!), and so forth.

[35] For instance, when we examined the configuration management functions of the Army Materiel Command using IDEF0, the effort comprised the simultaneous integration of findings from eight different sites distributed around the country. Collectively, these the functions modeled at these sites were conducted by approximately 100,000 individuals. Without rigorous modeling standards, it is very unlikely that such a task could have been undertaken satisfactorily.

created. A model from the CEO's perspective is likely to be different than a model from an assembler's perspective. They may have different purposes for the model, and thus focus on different slants of the process. This must be determined early. Otherwise, modeling objectives can vacillate during the modeling process, creating a model with mixed perspectives and insight on different diagrams. As a memory aid, the Context, Viewpoint and Purpose are clearly stated on the first diagram, so there can be no confusion.

Preliminary modeling: Once the perspective of the model has been established, the team begins the modeling effort. Note that this is begun without any detailed data collection. Rather, this preliminary modeling is a "mental download" of team-members' existing perceptions. Though most of what will be created in this step will likely be purged by the end of the modeling process (this can be very humbling for some!), this step does permit some basic functional structuring. The functional names for the A-0 and A0 diagram should be established, followed by any and all known interfaces between these functions. It is not uncommon for this preliminary step to take over one week[36].

Data collection: Once the team is satisfied with its preliminary A-0 and A0 diagrams, data collection can begin. This usually entails going out to the organization(s) under study and observing and asking questions. We have utilized questionnaires; though they can provide very interesting findings, they have not

[36] My experiences are that more team members at this stage actually slow the process down, as there are more opinions to be "wrecktified". At later stages, there is notoriously a shortage of team members, as the model grows faster than members can keep up. Recall that interfaces rise on the order of the square of the number of functions...

been helpful from the perspective of modeling[37]. Interviews with actual system participants seems to be the most effective technique. One must be careful, however, not to look like managerial analysts. This is one reason that the establishment of an objective team of participants is so important. Let the participants interview other participants. It can be surprising to see the level of astonishment of interviewers, once they see the world through the eyes of their colleagues.

Detailed modeling: This step is conducted simultaneously with data collection. In fact, they can drive each other (as long as one remains within the agreed-upon context of the model). New data provides new model ideas; model creation uncovers new areas to get/refine data. The basic routine here is to begin decomposition of the A0 diagram. It may prove useful to sub-divide the team into functional groups, though this seems to depend upon the substance of the system under study.

Notoriously, the team will need to change the A0 (and often the A-0) diagram(s), based upon their newfound understandings. This is OK, for these are the diagrams which the entire model hinges upon. In this regard, model development is similar to product development, ***slowing the system down early*** (while requirements and major interfaces are being established) ***may be necessary for the overall process to finish faster***. One does not want to make major changes late in the model

[37] Questionnaires can be very useful for gathering detailed quantitative data, however.

development process: ripple effects of changes can be disastrous. Get each level right before proceeding to the next decomposition level!

To maintain rigor during IDEF modeling projects, no graphical drawings are to be developed without a companion textual description and complete definitions for *every* term on the diagram. Normally, a team is responsible for developing the model; in such cases, definitions are determined by consensus of the team. The reasoning behind this is clear: commonalty of meaning is critical if the same item affects functions across the model. By consolidating and agreeing to definitions, it is also easier to manage the glossary, for fewer distinct terms need to be defined.

Often, the glossary and text diagrams are postponed until "we get things gelled". THIS IS A MAJOR MISTAKE!!! Glossary terms are critical to establishment of a coherent model, not just to model readers, but for model developers as well. As interfaces branch apart and then recombine, loose terms become the norm. If developers are not quite sure about the content or definition of a term developed by another, how is s/he expected to utilize it accurately? Develop definitions for terms early, as they are created. If changes need to be made to a definition, this is acceptable, as long as all affected parties (other developers) are aware of (and can accommodate) the change.

Text diagrams are not usually as volatile as term definitions, but can be problematic as their generation is postponed. Specifically, text diagram details seem to fade as their writing gets postponed. The very best textual descriptions are

those which were written immediately after[38] the visual diagram, and then revised regularly. For many modelers, textual descriptions are low priority, because they have seen the process first-hand, and don't appreciate that readers of the model never may have such privilege.

Validation: Other than the first interview session, there is probably no more troublesome time for the modeling team than during validation. This is supposed to be a time for modelers to confirm their visual and textual descriptions with their field sources, and make sure that the parent-child relationships across the model have been executed correctly. The former objective is usually trivial; modifications tend to be minor or exception-type cases, easily rectified with adjustment of a textual description or terminology modification.

The latter, parent-child relationship alignment, is much more difficult. As modifications are made during the detailed modeling stages, parental relationship are often ignored. This would not be a problem for an isolated case. In reality, however, multiple "partial" cases exist. Wreaking havoc on parent interface and function definition. Both redundancy and gaps in functions and interfaces emerge as a result. In extreme cases, the validation step may take as long as the detailed modeling step.

---

[38] I have witnessed that some individuals are better at creating textual descriptions prior to the visual diagram. There is nothing wrong with this. In such cases, it may prove useful to ask such a person to also review the text of other diagrams. As Winston Churchill once purportedly said "We all speak a common language, and that's what separates us."

A better solution is to engage in validation during the detailed modeling processes. In this manner, the validation and data collection stages become virtually synonymous. By the time the model nears completion, only some formalities are necessary; the validation step becomes more like a grammatical check, rather than a major editorial review. In such cases, it is common for team-members to review the model in its entirety, one-by-one, as part of the reader-cycle. This gives all members a better outlook on the overall content of the model which they each contributed toward.

Next, it is customary for an "outsider" to examine the model for readability. Such an individual is typically familiar with the system under study, but has not had involvement with the modeling effort. Based upon his/her comments, team members may make final revisions. Subsequent to such final revisions, the model is ready for distribution.

After modeling is complete, analysis of findings obtained during this documentation process can be reviewed. Collectively, both functional modeling and analysis may be known as the ***AS-IS documentation*** stage of enterprise analysis. Future, prescriptive models and other visualizations are construed as ***TO-BE documentation***.

Using IDEF0 models

Upon completing an IDEF0 model, one has a fundamental structure in place, from which a variety of analysis techniques may be employed. These include process (flow) diagramming, activity-based cost accounting, cause-effect (fishbone) diagrams, and

interface analysis. Automated translation programs have been developed to assist in such processes.

Moreover, the IDEF0 model can act as a common reference device for managers and other system participants. This can be useful in clarifying semantics, illustrating process problems, and assessing/tracking change in the structure of the system.

However IDEF0 models are used, one must recall that IDEF0 is a static view of the functional relationships. It documents the existence of past functions and traces the interfaces which have existed between such functions. The real world, however, is a dynamic environment, filled with contingencies and changing structures. It was with this in mind, albeit with the fundamental useful perspectives elicited from IDEF, that we developed the CPP Structure in this study.

Appendix E: Some "Undocumented Features" of New Product Development

Product development studies by academia, management, and consultants are often immersed in trying to develop simple, understandable views of "the process" of development. Often, such studies offer an explanation of some ***subset*** of a development process, in hopes that it, along with results of other similar studies/projects will "bundle" into a more complete system which will behave in much the same way as each subsystem.

During the field studies, an over-riding theme became apparent, which conflicts with such hopes. The "process" of new product development is a complex, ill-understood phenomena, which takes on characteristics which are highly dependent on the surrounding environment. Perhaps needless to say, such characteristics are not anticipated by managers and analysts of a specific process[39]. Furthermore, when this theme applies, documentation of innovative development processes stifles managers and researchers who attempt to use convenient, traditional project management paradigms.

Some aspects of the new product development process have been agreed upon by several independent sources within this study, however. They are overviewed in the remainder of this appendix, with the following characterizations:

1) Interdisciplinary
2) Parallel
3) Unknown

---

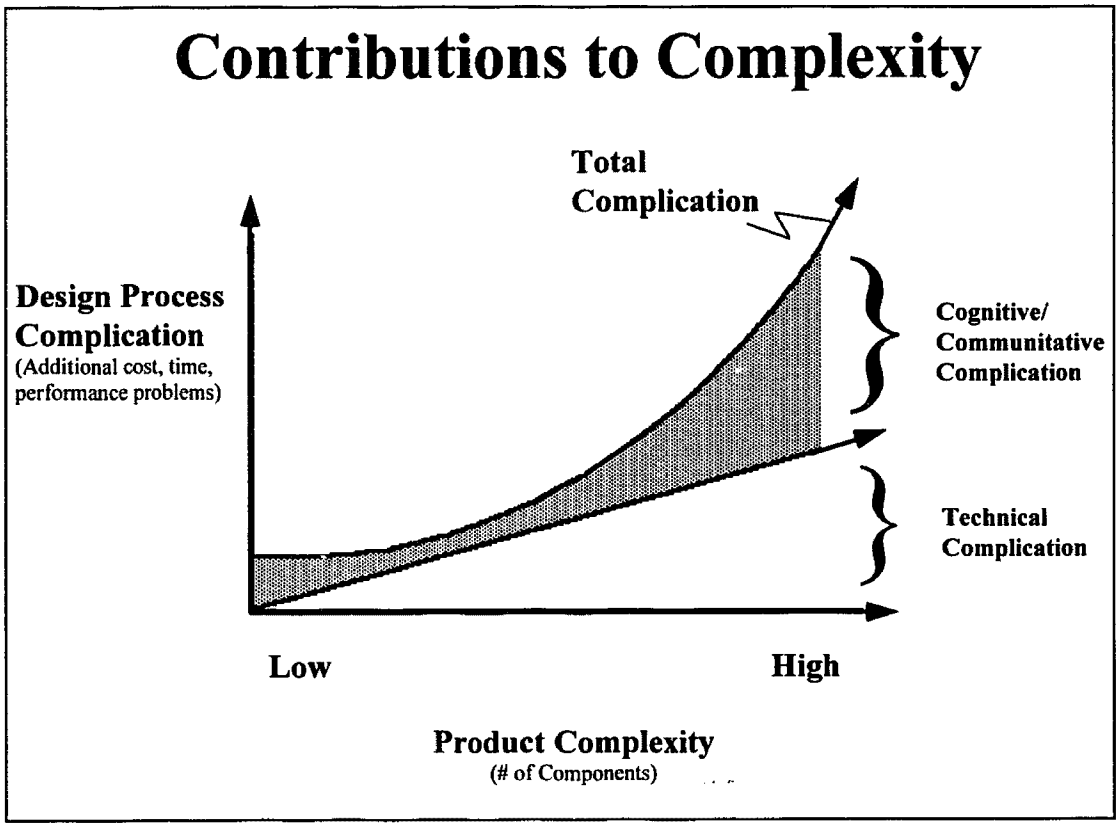
[39] A colleague of mine during parts of the field research, Dr. Allen Batteau, has a very insightful term for unanticipated characteristics, particularly when such characteristics apply to the lackluster performance of software programs. He calls them "undocumented features." Hence, the title of this Appendix.

Descriptions of these characteristics follow.

The Interdisciplinary nature of New Product Development

As product complexity increases, some developers have found that their troubles with integrating the various product features increase at a faster rate than the number of features themselves. Qualitatively, this is illustrated in Exhibit E.1.. In this exhibit, there are two components to the complication issue among developers: ***technical complication*** and ***cognitive/communicative complication.***

EXHIBIT E.1.



In Exhibit E.1., ***technical complication*** is shown merely as a linear function of the number of components which developers incorporate into a design. Thus, a product which has 5 components is only one tenth as complicated as a product with 50 components[40].

In addition to this technical complication, however, managers have been struggling with other factors which result in what we call ***cognitive/communication complication***. Its effects are demonstrated by the distance between the technical complication line and the total complication line. ***Total complication*** may be interpreted as the additional cost or time spent during design, as a result of more components in a product.

Based upon our field work, we assert that a major contributing factor to this form of complication is insufficient familiarity with the different disciplines which are applied during a complex design. If the developer is only one person, or a limited number of people, it may be possible to limit this problem with effective communication. Clearly, with a single developer "system," the communication channels are as short as can be conceived. However, as development teams are increased in size and specialties become more focused, it has been observed that individual team members communicate ***less*** effectively. Why?

We can attribute these interdisciplinary communication problems to two factors: ***technical competence of non-specialists*** and ***separate vocabularies***. The first is a natural

---

[40] Of course, some products may have some significant interfacing to be designed among components, which has the effect of increasing the slope of this line. If all n components have two-way interfaces with each of its counterparts, then the technical contribution to process complication would be on the order of n^2, significantly steeper than the linear (proportional) line demonstrated.

outgrowth of the human knowledge tree, as visualized in the introductory chapter of this report. By definition, as individuals become more specialized, they have less and less time to dedicate to other specialized disciplines. Even if they have the time, there may be cognitive limits to simultaneous, detailed understanding of multiple specialties.

In fact, it was established long ago that there do exist very real limits to human cognitive awareness. Four decades ago, George Miller asserted, with considerable evidence, that an individual's ability to make absolute distinctions among stimuli, or to remember a number of discrete items, all seemed to fall appreciably at about seven items (Miller, 1956). He suggested this is due to a physiological limitation of human nervous systems and that this limit is apparently true for all of our senses[41]. Thus, it may be postulated that we are inherently limited in our abilities to bridge the disciplines. Since product development, particularly innovative product development, requires developers to identify, choose, and apply many different disciplines, it would be apparent that very complex products are outside the realistic realm of a single developer. But for a few exceptional cases, this appears to be true.

The natural solution to this problem is to enlist the assistance of specialists. The innovative, and often entrepreneurial, individual then changes his role from developer to integrator --a role which requires cursory, though wide-ranging knowledge of many disciplines, to enable pooling of appropriate specialists. This is where the second factor

---

[41] As an aside, it is interesting to note that the telephone company determined, after much study, that the average person could adequately remember a seven-digit phone number, particularly if grouped into groups of three and four. Whether this was related to Miller's findings or not is not known. However, Miller was apparently looking to bridge the fields of computer science and information theory, of which Claude Shannon was a principle researcher... at Bell Laboratories!

(vocabulary) is observed to bear more significance in the cognitive/communication complication. When specialists are put in a room together, there appears to be a limit to how much significant (design-related) information can be transferred. This limit may have nothing to do with the intelligence or knowledge base of the various parties. Rather, we suppose that it arises from limited overlap in vocabulary.

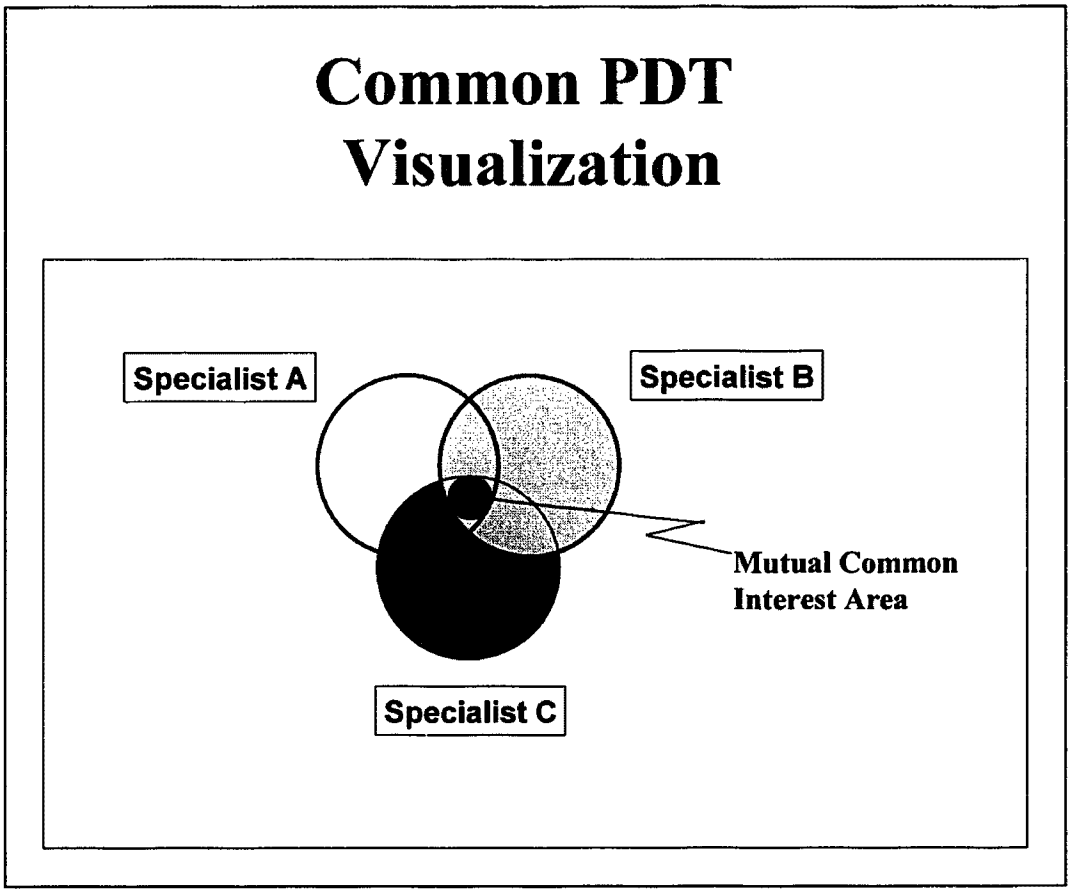
At times, listening to engineers from different disciplines attempt to converse is like having an Italian, a Frenchman, a Swede, and a German in the same room: they may all have valid messages, and all speak Centum languages, but have such specific vocabularies (with very specific meanings) that only similar roots of words get transferred as interpretable information signals. Such broken signal transmission can have the unwanted effects of neutralizing or, worse, ***inverting*** the intended information signal. To paraphrase Winston Churchill, "We speak a common language...and that's what separates us!"

Despite development of a large variety of linguistic theories, including Miller's catalytic development of the field of psycholinguistics (Miller, 1962), research contributions to understanding communication between specialists of different background have been far from practical. Yet, this problem seems to plague developers and non-developers in all organizations visited in this study.

In many firms visited, diverse product development teams (PDT's) have been established to alleviate communication gaps among relevant disciplines. Of course, these teams still fall prey to the vocabulary factor mentioned above. In fact, observation and interaction with members of such PDT's demonstrated this phenomenon innumerable times.

Nonetheless, PDT formation does at least offer a forum to enable interfacing among specialists who would otherwise have no contact with one another. This concept is often illustrated with overlapping Venn diagrams, which temp one to think of PDT's as carrying some significant common knowledge among members. Refer to Exhibit E.2.

EXHIBIT E.2.



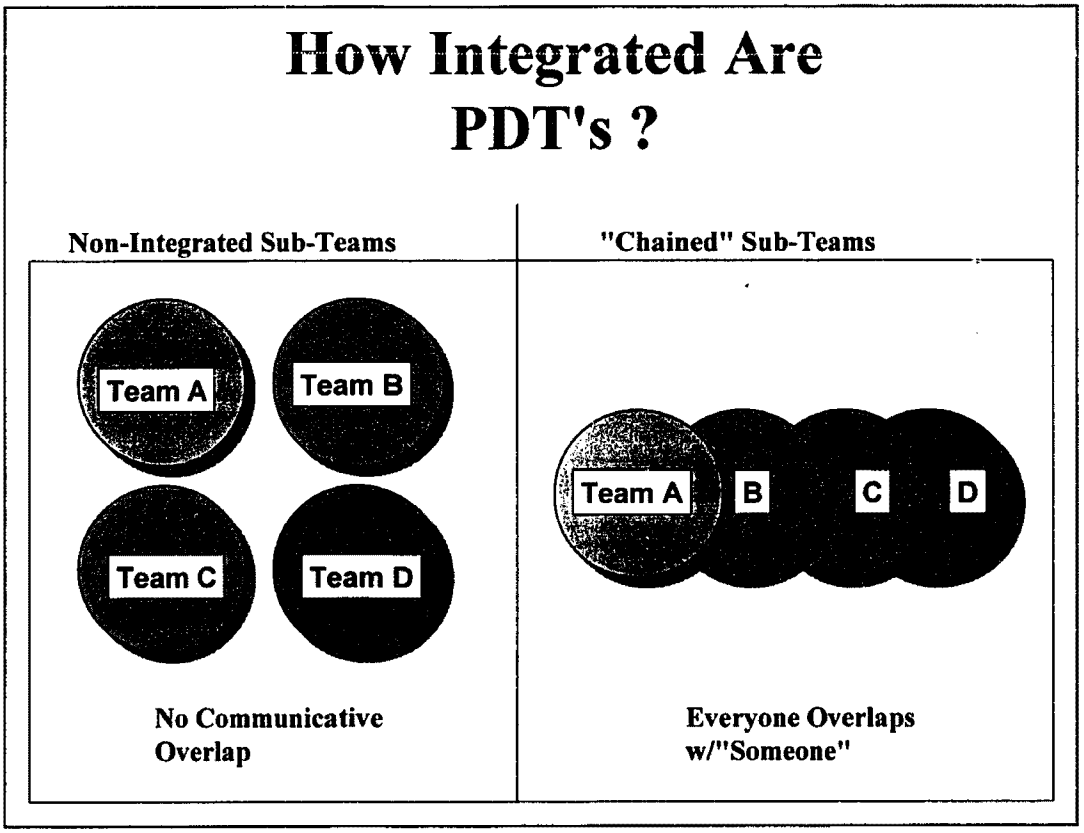
The "cross-fertilization" strategies employed by management were observed to be far from consistent in their treatment of PDT's. In some organizations, interdisciplinary product development teams existed in name only. Under this condition, many developers

were assigned to a product "platform" on which they are given a specialized responsibility, but little or no authority to make meaningful design decisions. When such developers realized the apparent fruitlessness of offering their expertise, their interest waned. Membership on such a PDT became an obligation of employment, rather than an exciting, meaningful experience. Thankfully, this condition seems less prevalent than years past. However, in some design environments, such stoic PDT's are still established, merely because management feels that "PDT's are a good thing to do."

By establishing a PDT with diverse personnel from diverse specialties, it was not unusual to witness development of "strange" personal associations. In such cases, individuals who normally would never converse with one another, based upon occupational interest or outward appearance, became very good associates. At times, they found a common interest area which had no immediate relevance to the development project. In fact, their established friendship often lasted long after development was complete. As their interpersonal communications increased, the information channels between such individuals were observed to become less obscured. It was also observed that some specialists would offer to investigate some new idea, not for the team *per se*, but rather as a favor for a fellow team member. In fact, it was common practice for a PDT to naturally break-up into smaller "sub-teams" composed of only 2-3 members each. This enabled friendships to flourish and for breakthrough, "off-the-page" ideas to be considered, without the cumbersome formality of the entire PDT. After consideration by sub-team associates, many such "off-the-page" ideas became reality on the final design!

A few "PDT buddies" does not constitute an integrated design team, however. It is observed that there needs to be some thread of continuity from sub-team to sub-team. Refer to Exhibit E.3.

EXHIBIT E.3.



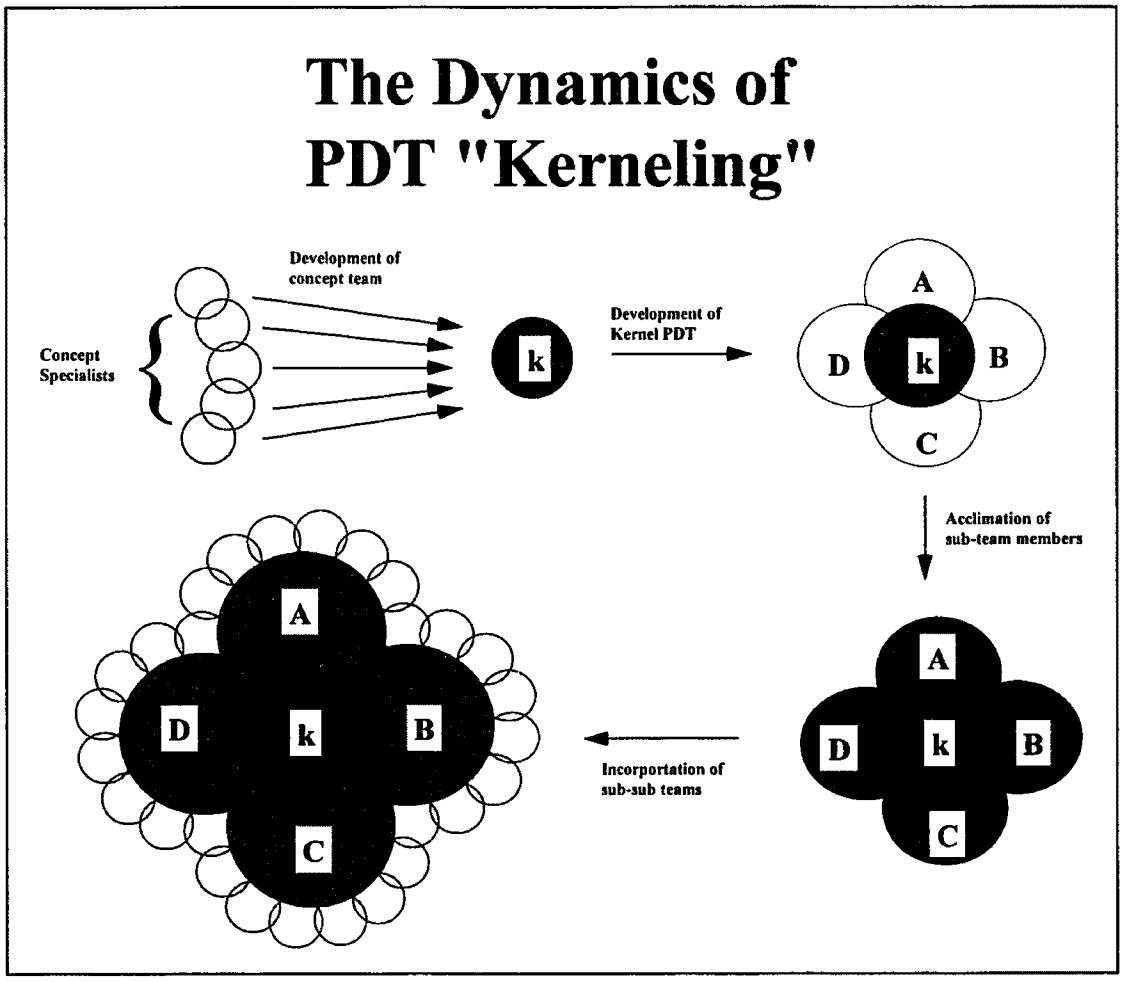
The issue of integrating team member expertise is often addressed through the practice of matrixing sub-team members within a PDT. In this scenario, members of a sub-team are also members of one or more other sub-teams. By overlapping team membership on sub-teams, it is expected that each sub-team is better capable of directly or indirectly communicating with every other team, via common membership.

It is observed that this "chaining" strategy works when overlapping team members are capable of convincingly communicating design interface problems with members of the connected sub-team. If this member is not persuasive enough to communicate interface problems, then he/she becomes a "weak link" in this chained PDT. Either one of the sub-teams make appropriate changes at the appropriate time, or both sub-teams work in a non-integrated manner with results to be decided at a later date. Often, this postponing of problem resolution causes major problems at a later date.

At a few sites, a more structured PDT development strategy was employed. I call this the ***kerneling*** strategy of PDT development. For this scenario, a core team is developed. Initially composed of senior conceptual design personnel, who originally started the project with the board of management, this PDT starts off very small. As the conceptual design "gels," more members are added to the team. Generally, these are composed of younger managers and developers who are on a continuous learning program of design activities. Each of these team members become sub-team leaders as the development progresses. Yet, they are still bound to the core (kernel) team of conceptual strategists/managers, who are responsible for overall shifts in direction. This is contrasted with many phased PDT's, which dissipate personnel when their chronological "phase" of the project is completed. In such phased development, many managers shift from project to project, repeatedly instilling havoc on downstream process activities. By the time feedback can point out such difficulties, however, the manager (or team of managers) has moved to another development project, and no longer bears responsibility to his "old" (previous) team.

With the kerneling strategy, as the product design detail increases further, additional "sub-sub" teams are developed, which use the original sub-team membership as their leaders. More bifurcations of cores and resulting sub-teams are initiated as the product design becomes more complicated. The process steps to this iteration are demonstrated in Exhibit E.4.

EXHIBIT E.4.



The important point of the kerneling strategy has been that the "flowering" size of active team membership expands and contracts according to the design ***need***, not some preconceived notion of what "phase" the process is in. Members of the PDT remained accountable to the development project, even if they were not involved in daily project "crisis management." This diminished lags and rework due to learning curve effects of "new" team members and knowledge loss from the exodus of previous team members. As one manager expressed, "We are like family --once a team member, always a team member."

In effect, resulting "family-pride" means that there is no distinct "end" of a development project. Follow-up incremental product refinements are conducted by various "original" team members throughout the product life cycle. This provides a training ground for the younger engineers and managers, as they experience the "downstream" (customer) effects of their engineering activities. It is not uncommon for members of "design families" to perform service training education for field technicians. Such activities presumably enhance the experience and diversity of engineers in a manner that will help them, as future managers, to better guide future engineering projects. For many, this seems to remain a lifelong interest: We have encountered several such former engineers who, ten years after retirement, still return for occasional consulting engagements for the company.

Nevertheless, this strategy does not seem conducive to all organization cultures, for it necessitates acceptance of rigid corporate hierarchy and confidence that the initial design concept will be robust throughout the design process, and thus confidence in marketing-derived customer requirements (used as a basis for the initial design concept). Moreover, and perhaps even more significant than generally believed, this approach

demands a dedicated workforce, where long-term employment is expected and cherished among all levels of the organizational hierarchy[42]. For a European development organization which most vividly demonstrated this PDT kerneling, mutual dedication between the firm and personnel was widely recognized. For this organization, the product development time and expense were not particularly good (according to U.S. managerial expectations), but the performance and thorough integration of components of the resultant product has tended to set formidable world-wide design and production standards for their entire industry, upon nearly every product release. Once again, we return to the perennial trade-offs between design cost, development time, and design/product quality.

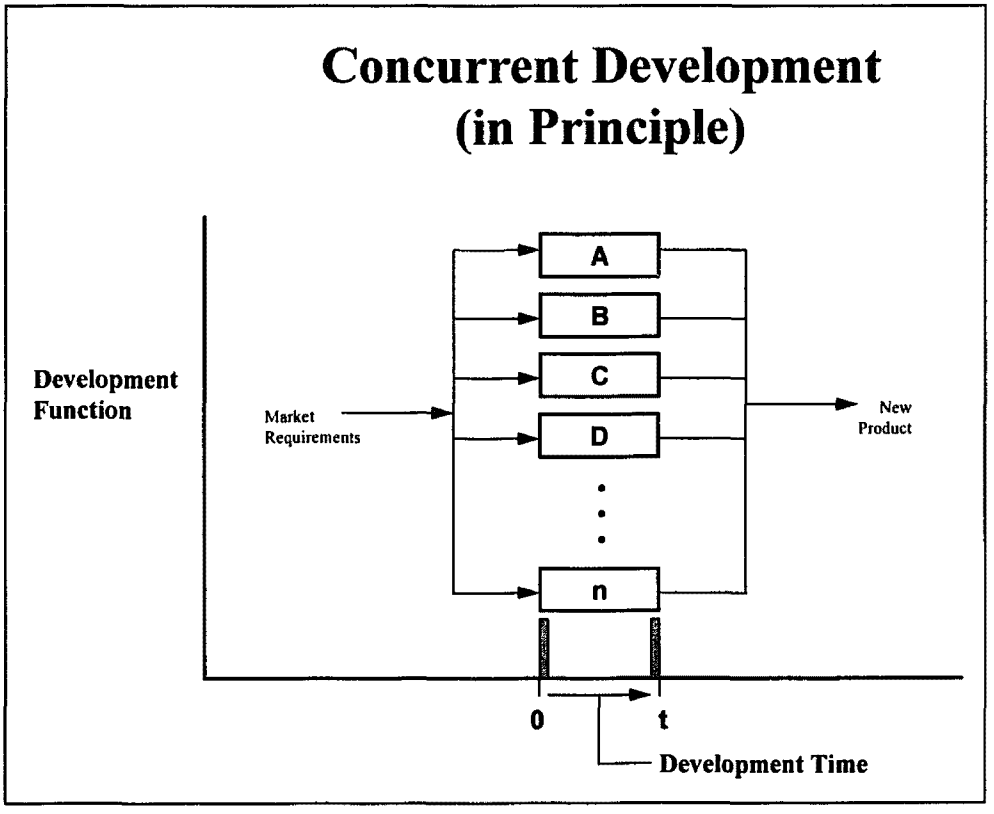
[42] This points to the important distinctions among cultures within organizations, between organizations, and between major societies. Though outside the central focus of this study, cultural considerations may be critical to the rules and structure by which individuals operate in the "system" of development.

Parallelism of New Product Development

The Concept of Concurrency

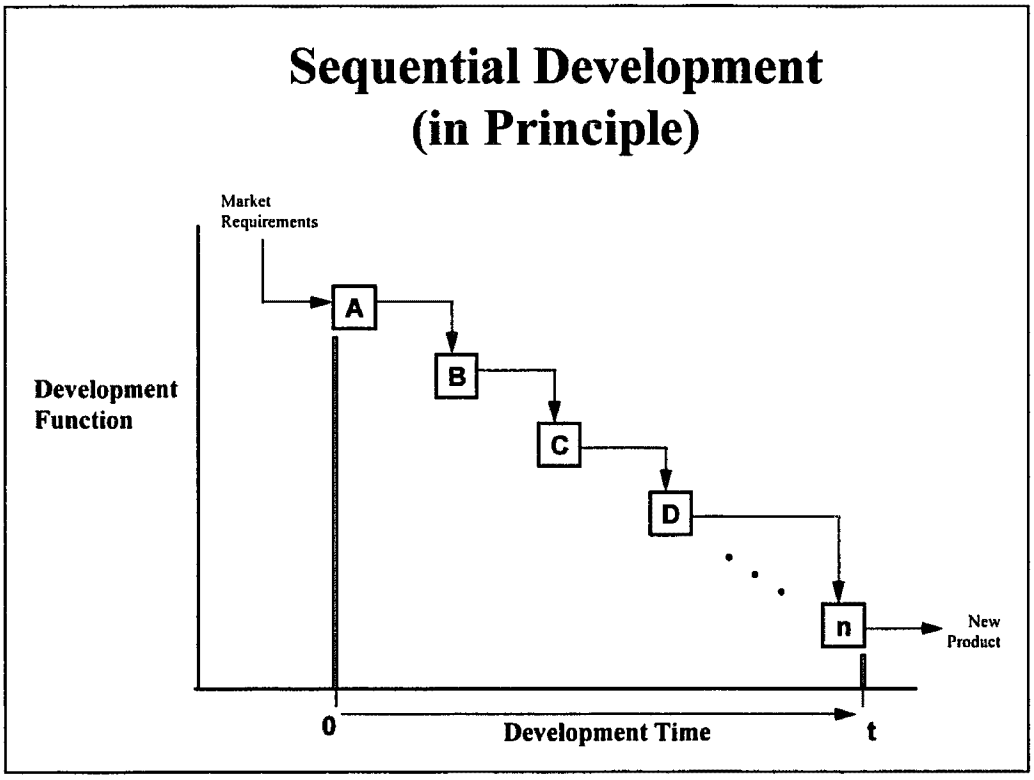
When one considers the formation and dynamics of product development teams (PDT's), it is important to consider the utilization of resources at a given time. Just because 250 developers participate in the development process of a particular product, and the development takes 2 years to complete, it does not follow that 500 man-years were spent on the development process. This would only be true if ***all*** the developers worked on the project, simultaneously, on ***all*** working days. Refer to Exhibit E.5. In this study, not a single site has demonstrated this characteristic of 100% parallel track development.

EXHIBIT E.5.



On the other hand, if *no* parallel processing of development occurred, a 500 man-year project would take 500 years to complete! The result of this (refer to Exhibit E.6.) would be that each of 250 engineers would work an average of 2 years out of every 500 years -- he would be idle 99.6% of his time! As poor as some engineers feel their process is, this condition of 0% concurrent development (100% sequential development) was not apparent in the field studies either.

EXHIBIT E.6.



Every organization in this study conducted some degree of concurrent development. Even single-person development firms conducted parallel development, through extensive use

of out-sourcing, for example. Thus, this concept (also known in the field as ***concurrent engineering, simultaneous engineering, synchronous development,*** or ***parallel development***) is well known and practiced...more or less. Perpetual questions of concern include the following:

1. ***Extent:*** To what level is concurrent development being practiced now?
2. ***Appropriateness:*** How simultaneous can/should development really be...is there an upper limit to its usefulness?
3. ***Transition:*** Given a difference between (1) and (2), how can a firm position itself at the appropriate level?

These questions are much more difficult to answer than they are to postulate. In fact, research centers such as CERC at University of Virginia and DICE within the U.S. Department of Defense have spent a great deal of time attempting to formulate and answer appropriate subsets of these three simple questions. Major systems-engineering departments exist in some large development organizations, with this issue as one of their focal points. Vendors of CAD/CAM systems and systems-integration software are continually searching for (and asserting that they have found!) answers to these questions, in efforts to enhance sales of their products.

To enable objective analysis of concurrent engineering in actual organizations (rather than just abstract models), it is useful to review a few frequent assumptions about this concept. Subsequently, we can provide some interesting considerations which may help development managers answer the above questions for ***their specific organizations***.

Common Assumptions in Concurrent Engineering Analyses

The concept of concurrent development is probably one of the oldest forms of systems analysis in existence. One could speculate that past and current project management methods have their roots in the philosophy of concurrency. Yet, management awareness of terms such as "concurrent engineering" is arguably higher today than ever. We speculate that this may be a result of increased global competition over the past several decades. Certainly, many development managers are beginning to analyze their processes much as manufacturing engineers analyze production processes. In some cases, this may be a natural result of the ascension of manufacturing managers in corporate hierarchies: frequently, engineering managers in this study were found to have backgrounds in manufacturing management prior to entering the engineering/new product development ranks.

Though this migration may be an explanation of the recent focus of manufacturing analysis methods on development activities, it does not mean that their methods are necessarily appropriate. Some of the common concurrent development analysis methods in use today make assumptions about the nature of development. It is questioned here whether such assumptions are congruent with development, be it routine development (continuous product improvement) or seminal innovation. Three major assumptions include ***predictability***, ***non-recurrence***, and ***structural consistency*** of the "system" under study. Direct and secondary observations reveal that one or more of these assumptions are violated during every development process. Depending on the degree of this violation, this may render many detailed systematic designs for concurrency to little more than futile exercises.

Predictability

This is considered an indicator of how consistently an activity is performed, and how well we can forecast its execution. In a manufacturing process, this is determined by experimentation and observation of assembly personnel, and by calibration of machine rates, for example. When numeric values are unknown exactly, but the nature of the process is known, it is customary to estimate process activities with statistically convenient distributions. Such luxuries are not necessarily available to the development manager, however. Development is a diverse collection of continuously changing problem-solving activities, not a structured routine of operations anchored to well-established physical laws. Thus, the completion time for activities in one development may be of little or no use to predict the completion time of similar looking activities in another development project.

Non-recurrence

This is a feature of unidirectional, non-reversible systems. Most manufacturing processes are unidirectional. They rarely work on principles of iteration. Development, however, is composed of many integration activities which require two-way communication ability among participants. Some of these activities have been observed to occur multiple times over the course of a development project. When operating at very low levels of concurrency, communication between activities seems to be far less frequent. Using nomenclature from the sequential development chart (Exhibit E.6.), the executor of function "A" sends information (via hard documents, physical hardware, verbal, or electronic means) to the executor of function "B" When the "B" activity is complete, information is passed

to "C." As long as the upstream development activities provide a large enough "feasibility window" for the next activity to be performed, the process will move forward. However, if the development feasibility window closes or is perceived to be closed, process progress ceases. The development activities to date need to be reviewed, and resolutions to the closed feasibility window need to be "resolved." This resolving activity may require the process to restart at an upstream activity, and rework major elements of the process again. The subsequent process ordering may or may not change as a result of this "resolution"[43]. By working in this sequential ordering process, it is easy to see that development time can lengthen dramatically as a result of such cycling activity. In fact, developers at nearly every site estimated that they design their product ***at least 3-5 times over*** during a single product development project.

Exhibit E.7. demonstrates an "official" process flow chart for development at one site visited. Even this simple diagram exhibits concepts of concurrency and iteration which occur at the highest, abstract levels of functional (and in this case, organizational) hierarchy[44]. When interviewing several individuals who were

---

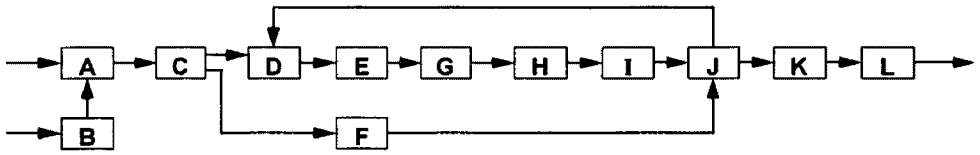
[43] Actually, it inevitably changes the process ordering, providing one is examining the most detailed process activities. As one looks at the system from a more global view, process ordering *looks* more stable. This becomes a factor of the number of cycles and size (and scope) of cycles which the process goes through. It is postulated that a "sunk cost" mentality reigns among reviewers, which psychologically prevents designs from performing many large "painful" cycles. For the same reasons, "politically-correct" process flow charts have similar characteristics.

[44] Although this simple diagram reveals these concepts, and supporting documentation to this particular process announces their existence, actual analysis methods used subsequently to this work (it was originally used to help describe the computer systems which support this process) completely ignored the iteration component. Rather, traditional PERT and SPC-derived methodologies were used! Later in this chapter, we shall demonstrate another look at the ***same*** organization, and gather a very different impression about the nature of its development process.

involved in the preparation of this process view, it was acknowledged that this is a simplistic view for high-level management. Specifically, one major intent was to demonstrate to executives how engineering departments related to one another. Though they realized that more recurrence existed, this was thought (hoped!) to be a special, rare condition which would only serve to confuse executives. Unfortunately, recurrence seems to dominate the actual development process.

EXHIBIT E.7.

The "Development Process" at one site



Each of the lettered boxes (A-L) represent distinct functions, performed by distinct departments within the engineering organization.

Structural consistency

This is the degree to which a system being analyzed retains its structure. Thus, a fixed manufacturing process can be analyzed the same today as it can tomorrow: it has the same physical layout, the same inputs and the same output(s) over the time period in which it is being analyzed. Even a flexible manufacturing system (FMS) has limited domains of problem space over which to analyze the process.

> For a given FMS setup, the process of manufacturing is known and is stable. A system of development, on the other hand, is continually in flux. Studies to pictorially document even routine development processes reveal so many contingent changes that the descriptive document is obsolete by the time it is published. One might say that the analysis data gets "cold" long before anybody even realizes that it can be considered "old."

Looking Back to See Forward?

Reflective of these assumptions, concurrent engineering analyses tend to be focused on examining past development activities, rather than predicting and managing future development projects. This *ex post* analysis approach reveals what ***should have*** been done to shorten development time, reduce cost and increase design quality. When prescribing "fixes", however, analysts must consider how robust activities are from one project to another. Despite the efforts of some managers to become "closer" to their developers (via breakfast meetings, open-session forums, "MBWA"[45], departmental social outings, and so forth), little progress has been made in forecasting activity characteristics. It seems the classical problem of communications between engineers and their managers is not fully responsible for disappointing managerial forecasting.

Rather, a key problem is that the sources of such communication (the engineers) do not necessarily have adequate knowledge of the process themselves! This is an important and likely controversial finding, which bears repeating: ***Many developers and managers do not understand the nature of the very development processes which they create and***

[45] MBWA = "Management By Walking Around"

foster. When this oft observed characteristic (discussed more in the next few pages) occurs, communications between engineers and managers become more conceptual and hypothetically-oriented than pragmatic or directly related to process characteristics.

Although one cannot admonish managers for analyzing past performance and trying to gain some "lessons" from such analysis, it is tempting to over-react to previous activity performance as an indicator of the performance ***potential*** of that activity. When attempting to operate in a concurrent development atmosphere, "knee-jerk" reactions can serve to throw the system into more plentiful and more complicated recursions, perhaps so complicated that accurately tracking development status exceeds the capability of a manager or his management team.

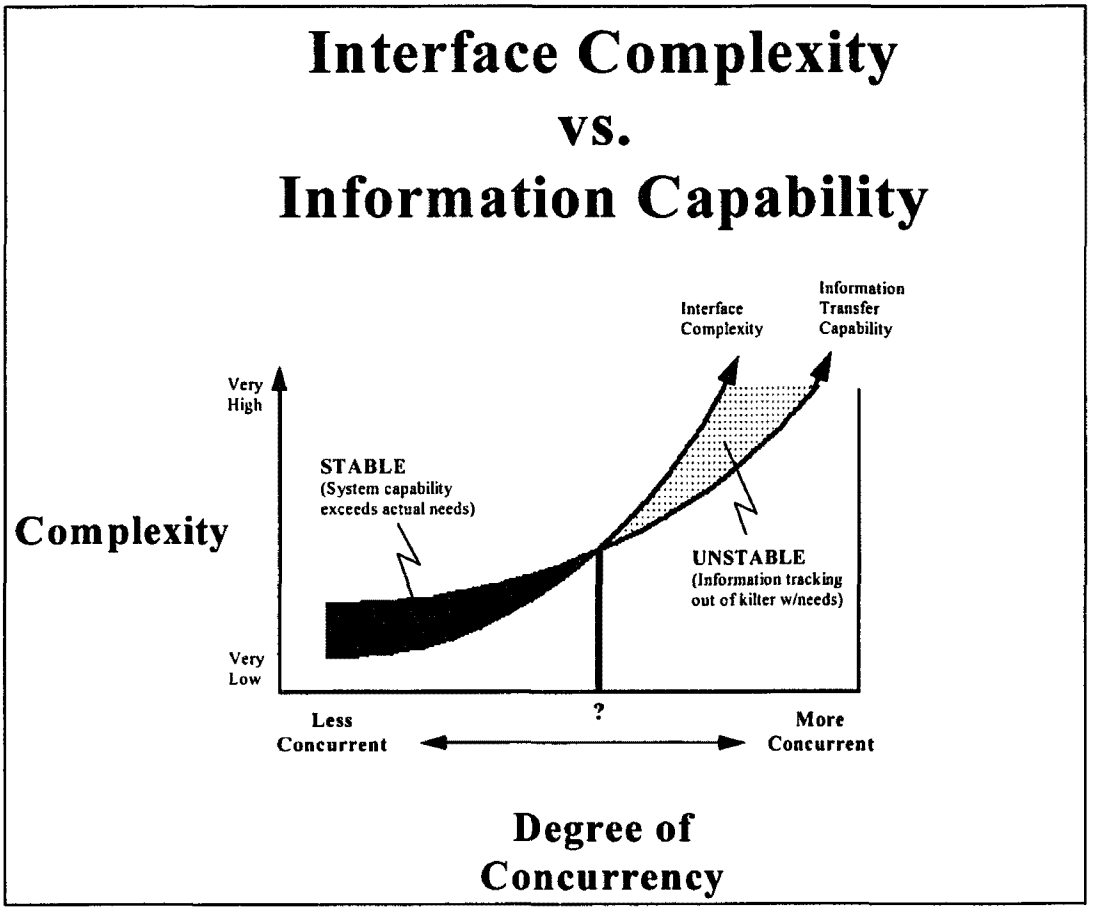
The Computational Paradox in Business Systems

Even the advent of MIS (or ***business systems***, as they are often referred to) into development organizations may offer too little help to managers wanting to know "what's happening." In many cases, automated systems have served to *hurt* managerial capability. This is readily understood when one considers that developers' ability to change status may increase faster than the systems' ability to track such changes[46]. Thus, even though tracking systems have indisputably improved technologically, they are soon asked to track much more complicated interfacing than they are capable of, and thus fall short on a practical front. This computational paradox is demonstrated as a function of concurrency level in Exhibit E.8.

---

[46] This problem was particularly acute in the configuration management processes of the Army Materiel Command.

EXHIBIT E.8.



If interfacing complexity is low, then simple tracking systems, even of a manual nature may work just fine. Of course, limited complexity may be associated with a lower degree of concurrency, as there is less need to interface with other functions. Thus, the capability of even simple manual systems may exceed the natural complexity of everyday activities. As long as this condition holds, the manual, semi-automated, or fully automated system works fine. Such a scenario is associated with the left shaded region of Exhibit E.8.

As concurrency efforts increase, however, the need for simultaneous information sharing intensifies. From a strict channel-number perspective, this increases with at least the square of the number of concurrent activities. If the information systems are "improved" at a slower rate (but still positive slope) than the interface complexity, however, there may exist a crossover point at which information systems capability and development complexity due to concurrency are exactly equal. Beyond this point, the interface complexity begins to dwarf even impressive gains in information systems capability. This will consistently induce an information transfer rate which is out of kilter with the speed (usually, but not always, too slow) of the more concurrent process. When operating in this region, developers fall out of synchronization with other developers and, of course, managers trying to control the system of development. Thus, some personnel are acting according to old information, and systematically adding invalid information to the system, causing instable designs to become even less stable. Of course, trying to increase concurrency and activity interfacing with the existing system can increase this instability "gap" even more[47].

Can Confusion Help Innovation?

Just because a system may operate in this unstable range does not mean that the development process will become self-arresting, however. In fact, it is postulated that some development projects work "better," albeit slightly slower, as a result of a little information system inadequacy. After all, innovative development is supposed to be a creative process. If the system can flawlessly perform every conceivable task asked of it,

---

[47]The Complex Process Path (CPP) system described later in this report was developed to help explore this right-hand region in greater depth, to see what happens when the developers take control of their process with no active management intervention.

then one might ponder the driver for creativity. What triggers an individual to become so dissatisfied with the status quo that a whole different perspective or paradigm is contemplated?

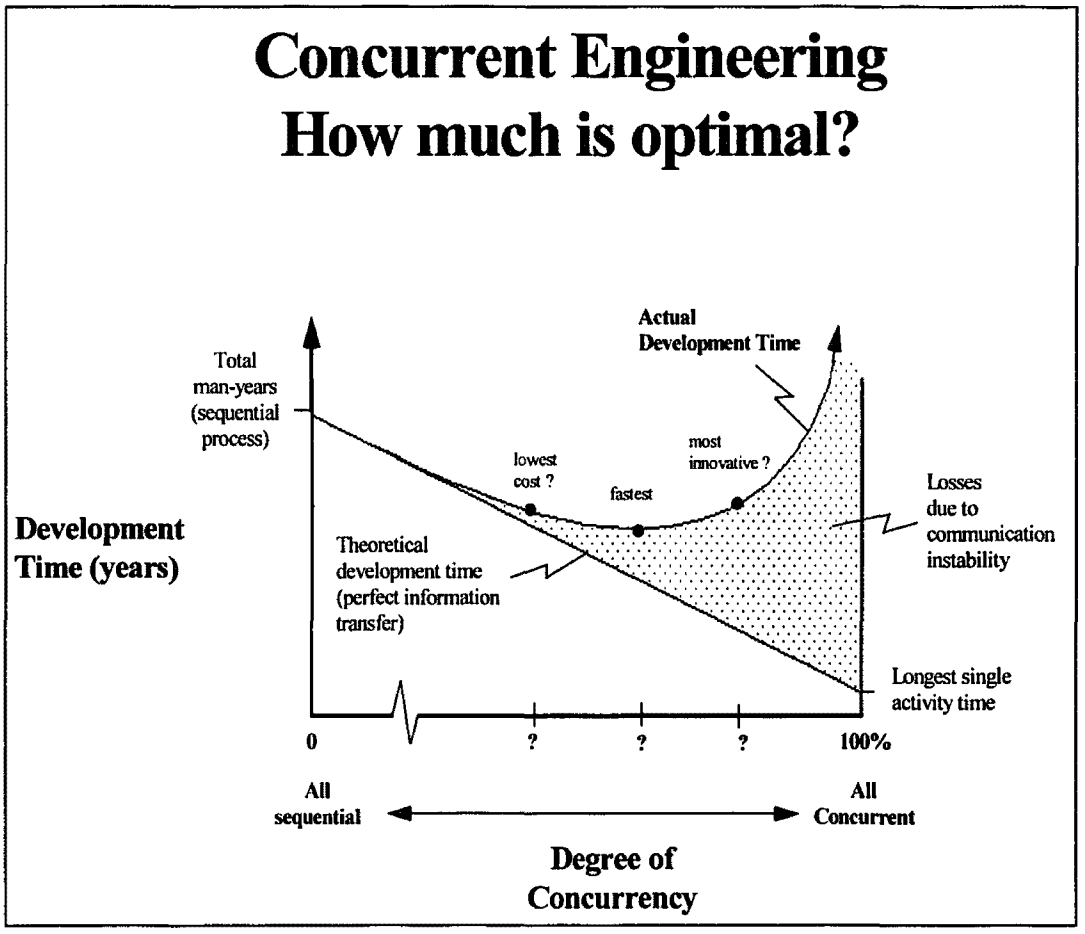
Considering the serendipitous nature of some discoveries (see, for example, Roberts, 1989) it is proposed that some development ideas have their genesis in the developer trying to solve a different, seemingly unrelated problem. For instance, suppose developer A is encountering difficulty in communicating a new self-defined concept to developer B. Recognizing his difficulty in the current description, he pauses and re-thinks of ways to communicate his concept to developer B. During or after this re-think process, developer A realizes that he actually had not considered another critical aspect of the concept. This may enhance or diminish the concept's validity to developer A, the very originator of the concept in the first place! In the process of trying to solve a communication problem, the information source discovers new ideas which might never materialize otherwise.

Depending upon the degree of this inverted communication-creativity relationship, the development process may improve or deteriorate in strange ways with respect to concurrency. Refer to Exhibit E.9., which compares product development time to concurrency of operation. If no communication problems occurred with increased concurrency, then we expect that improvement in development is merely proportional to the degree of concurrency. With no concurrency, the development time is just the sum of the man-years spent on the development project[48]. With 100% concurrency, the

[48] This assumes that the completion time of each function coincides with the start time of the "next" function. By attributing such idle or "dead" time to either of the corresponding functions, we can make this assumption without excessive loss of generality. With zero concurrency, of course, there exists no slack or overlap time.

development time is equal to the single longest activity time. When real communication problems are introduced, however, development time breaks away from this simple straight line. The difference, represented by the shaded region, is "lost time" as a result of communication instability.

EXHIBIT E.9.



As demonstrated in this exhibit, such loss is a continuously increasing function of concurrency level. At any level of concurrency, some of this loss is bound to occur. Given a management's desire to simultaneously increase product quality, decrease cost,

and minimize development time, how much of this communication loss coincides with "best" system-wide performance? Put another way, how much communication loss is acceptable to adequately meet all of the above objectives?

In a perfect communication world, where communication costs are negligible, cost and time objectives could best be met at 100% concurrency. We do not live in this type of environment, however, and efforts to achieve such communication ability can be frightfully expensive, perhaps technologically impossible. Thus, best performance must occur at some point less than 100% concurrency. It is contemplated that the point of lowest development cost does not coincide with fastest development time or most innovative performance[49]. Thus, management which uses simultaneous maximization or minimization objectives must be willing to accept that one or more objectives will not be met. Perhaps the best overall solution to such a multi-objective problem is one where ***none*** of the individual components are "optimal."

The Problem of Transition

This brings us to a fundamental problem of concurrency, particularly for large development organizations: ***transition***. Along with lags due to diffusion of information, large organizations contain various "pockets" of constituents with varying beliefs about the "best" solution to the concurrency problem. This makes a manager's task of aligning developers to a particular concurrency structure more difficult.

---

[49] Quality is such a subjective and loosely-used concept that it is difficult to track as a function of concurrency. Thus, it has been omitted from this discussion. It is returned to in Chapter VI, and is the focal point of Appendix D.

In some organizations, the design and implementation of certain information systems was not perceived as a support mechanism for developers, but rather as a structure to *force* developers into conformity. Though automation was widely acknowledged as a beneficial tool in development, many developers expressed their resentment of the use of technology as a "club" by non-engineering types to reduce the developer's workplace autonomy. This internal struggle, in a few notable instances, escalated into personal battles between engineers and information system personnel to such a degree that opposing sides either rejected or ignored the needs of the other. This makes transition even more difficult, as the communication gap incorporates more and more obtrusive socio-psychological components.

Two different approaches to transitioning from sequential to concurrent environments have been observed: the ***greenfield approach*** and the ***pilot approach***.

The Greenfield Approach to Transition

In this scenario, a brand new corporate entity is established, with little or no ties to the existing organization. For many intents and purposes, this new entity is considered a different company[50]. The new organization selectively recruits employees with the same philosophies as the managerial body. Thus, there is less "baggage" to handle with regard to existing corporate culture.

---

[50] In one widely-publicized organization currently undertaking this approach, the new entity has distanced itself, in the consumers' eye, from its parent company --largely because of growing negative customer impressions of the parent company's products.

There is, however, an investment in time required to establish a systematic administrative "gelling" of personnel who have not worked together before. After this learning curve effect is complete, the new organization is expected to become much more responsive, more cost efficient, and more innovative than the existing organization. If the learning curve is too shallow, however, one can reasonably expect increasing impatience among high-level management. If returns to this new structure are not soon enough, then the evolving entity may be deemed a failure before it ever reaches its potential. Thus, development managers can fall into a mode of looking for a short-term, high-payback effort to demonstrate the worth of their new structure.

Naturally, some such payback demonstrations are not representative of the new organization's true behavior. Rather, design cobbling becomes an intense activity shortly before interim executive reviews. Once a review has passed, the design organization may go back to business-as-usual... until just before the next review. After enough iterations of this activity, it is plausible to suppose that the "green" organization will gradually transform into an administrative structure which vaguely resembles the "old" organization.

The Pilot Approach to Transition

The second approach to transitioning to a concurrent environment is the "pilot" approach. In this scenario, a small sub-set of the development organization experiments with new methods of development. Their small size enables the communication channels to be shorter and more plentiful, even reducing the requirement for sophisticated information systems among team personnel. As

with the greenfield approach, development managers are often expected to "report" their progress/status to their senior management, and thus go through similar scrambles to "look good." With the pilot approach, it was often apparent that the sub-organizations actually do perform better than their parent organizations, though perhaps for different reasons than publicized. If this enhanced performance is recognized by management as well, then similar approaches are established in more locations in the organization. Each of these secondary pilot programs go through similar gyrations as the first, and gradually "blossom" throughout the organization. Of course, a little success in each program breeds a little more administration (and more than a few inflated egos!) until the organization becomes a collection of self-interested sub-organizations, just as it was prior to the initial pilot programs.

Thus, it may be asserted that the pilot and the greenfield approaches to transition ultimately give the same results: an organizational and functional structure which (in practice) closely resembles its predecessor. We have observed that executives of several successful companies understand that such circular organizational dynamics are natural. To counteract this problem, many have operated on a basis of continuous change, incorporating several simultaneous pilot programs, staggered in time and intent. When successful, these strategy of pseudo-competitive programs has kept each development sub-organization from resting on its past success, and kept organizing systems and structures from becoming "too" stable.

The underlying dynamics of such self-organizing structures are not well understood by researchers or managers. For researchers, objective data acquisition is extremely difficult

because the organization is so different and so fluid from one department to another. For managers, the period of the above cycling may be well beyond an individual's career interest. Further, on a number of fronts, the Heisenberg uncertainty principle seems to apply to this form of analysis: detailed analysis may affect the current structure, which may unknowingly and permanently affect the dynamics of future structures.

The unknown process of New Product Development

It was stated earlier that many important participants in the development process do not have an objective, up-to-date understanding of their own development process. This was observed directly at large organizations in the private and public sector. Specifically, this was discovered when documenting development processes, with the assistance of strategists, managers and developers. On the surface, inquiries to each individual were simple; they related to an individual's specialty:

1: ***What is your title?***
2: ***What would you consider to be the top five functions you (and/or your department) perform, regardless of official title?***
3: ***What "inputs" do you have to work with in performing each of your major functions?***
4: ***What do you consider to be your product(s), upon completion of your major functions?***

Next, interfacing characteristics were discussed:

5: ***What other person or organization do you work with in accomplishing your major functions?***
6: ***Who/where do your "inputs" come from?***
7: ***Who/where do you send your "products" to?***

Once these innocuous questions had been satisfied, and the respondents realized that these were not inquiries to evaluate their performance, but rather system-wide performance, the respondent was asked to "sketch" their operations. Specifically, they were asked to diagram the flow of paperwork and materiel from their sources, through their own functions and facilities, onto their respective destinations. Very few developers or managers had difficulty sketching this out in a matter of minutes. Upon finishing sketches, however, many individuals were dissatisfied with their initial sketches, and modified or re-sketched them as they thought about their activities and interfaces in more detail[51]. Inevitably, the sketches began as simple sequential diagrams and evolved into complex, detailed diagrams which reflected the recursiveness and conditional/contingent structure described earlier in this chapter. To a newcomer, these diagrams looked like a jumbled melee of boxes and arrows, reflecting the complexities that developers deal with on a daily basis. When described as a conditionally recursive process, however, it was clear to see how this patchwork of activity fit together within the daily/weekly scope of that individual.

When asked the next series of questions, however, the level of detail and familiarity in their answers fell appreciably:

8: How are the "inputs" to your process generated?

---

[51] As a brief aside, it was interesting to note how pleased many individuals were that somebody was ***asking*** them how they worked and interfaced with others, rather than ***being told*** what they should be doing. It is difficult to objectively assess how much more accurate, if at all, the statements of such respondents were. However, when they took pleasure in discussing their processes, their frankness and openness to introduce me to others in the process did seem to increase.

9: How are your "products" processed, after they leave your station/office/department?

In fact, after seeing the detail in which they had reviewed their operations[52] , the most common response was something to the effect of, "You know, you really should talk to [name of another specialist] about that --he/she'd know better than I." It was also at about this time that such individuals turned to official organizational reference materials to "look-up" the process or, at minimum, find the appropriate contact for clarification.

This phenomenon was so consistent among strategists, managers, and developers that the interviewing protocol was tailored to take advantage of this behavior. By having multi-colored pens (to distinguish multiple iterations), diagramming templates, official organization charts, and previously developed process charts for easy reference and documenting, later data gathering sessions provided fewer surprises, but more detailed, accurate information.

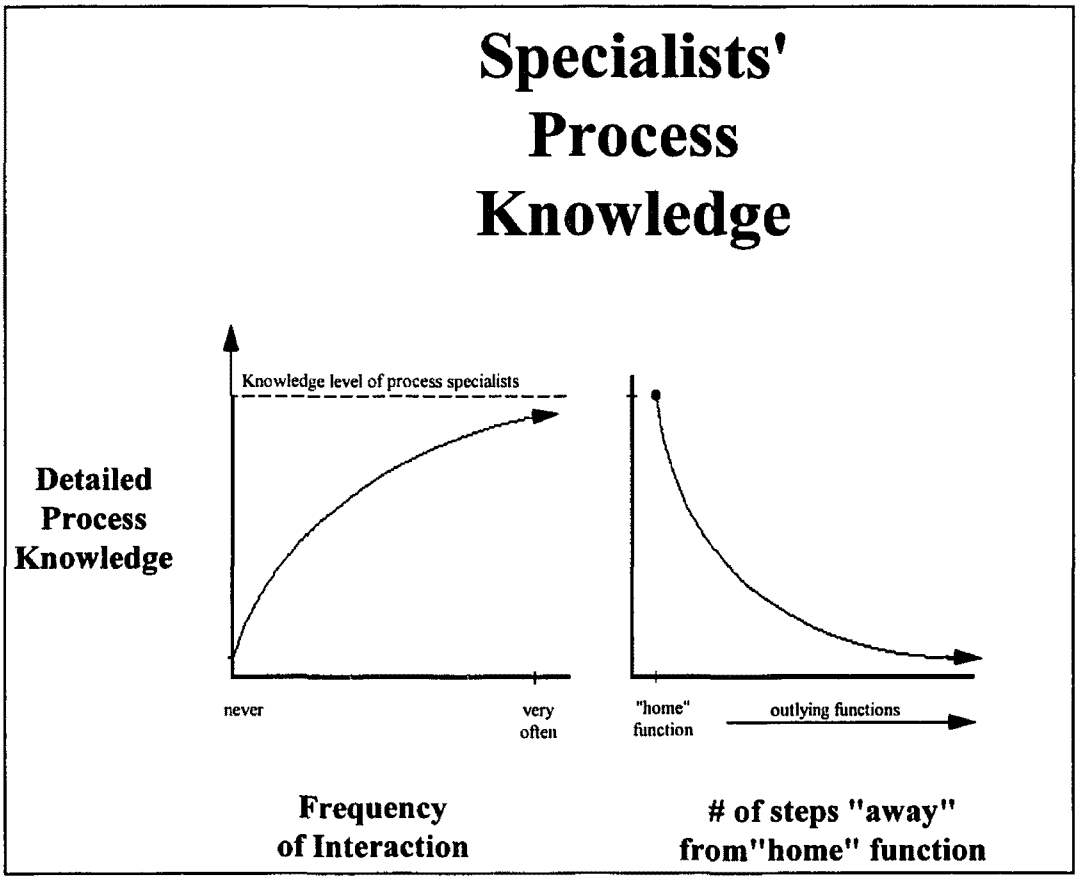
Significantly, it was discovered that the atrophy of knowledge about the process fell even further as one looked more than one activity beyond their specialty. This was particularly evident when asking ***who*** performed such "outlying" activities. Often, respondents did not even know the name of the departments which performed these activities, much less the individuals or detailed processes. Generally speaking, respondents were more familiar

---

[52] In a few cases, respondents indicated that they had never diagrammed their activity before, and made photocopies of their sketches before giving me the originals! It was not unusual for those same people to send "updated" diagrams to me several days later, as they reflected upon their operations even more.

with activities with which they interfaced more frequently. These features of their process knowledge are qualitatively illustrated in Exhibit E.10.

EXHIBIT E.10.



There were differences among strategists, managers, and developers with regard to the scope and detail of their knowledge. As might have been expected, strategists and high-level managers had the most broad-based understanding of process activities. They had a good appreciation for the entire spectrum of the organization's departments, for their everyday dealings often related to resolving major interfacing issues (problems)

among departments. However, they did not know, nor seem to care, much about detailed processes. The information they did have about detailed activities was largely second-hand, or based upon their memories from when they were participants in development (and thus assumed that the process hadn't changed significantly since then). Several high-level managers verbalized their knowledge with a high-degree of confidence. Even when such confidence was demonstrated, their statements were not necessarily supported during independent discussions with managers and developers. This is demonstrative of the well researched phenomena of ***mental imagery***: the human mind's ability to unconsciously fabricate representations of reality, with little or no substantial basis. It is reasonable to expect that executive opinions are based upon many instantaneous glimpses of the process in action, not integrative complete views.

In fairness to executives, this latter cognitive observation applies to the judgment of managers and developers in estimating how strategists/executives operate, as well. Even outside the arena of development, mutual misunderstanding and resulting mental fabrication of actions at other levels of the hierarchy seems to apply. In the military organizations visited, as well, it was observed that generals, colonels, majors,... down to the lieutenant ranks had prevalent misinterpretation of the reasons behind the orders and actions of others. Thus, even in a rigid top-down hierarchical structure, mis-projection of intents and actions is a regular occurrence.

Among developers and many "hands-on" managers, the scope of knowledge is far more limited. Rarely was a developer involved in actions which cover the entire spectrum of the development process. As described earlier, their detailed understanding of ***their*** portions of the process was more widely apparent.

This early observation barely prepares us for the next observation, however:

With little or no knowledge about the processes of others, many developers assume that the "other" processes have enough capability and latitude to accommodate their own variance of operation.

It is almost as if each developer feels that *his* is the only time-resource-cost or technology-limited operation in the organization. Having seen only short glimpses of the processes of others, some developers have a (natural?) tendency to be "process optimistic": they view others to have enough slack time, money, or technology to consistently perform at their maximum potential. Obviously, this assumption is ill-founded. One need look no further than one's own activities to see that there are perpetual complications which serve to temporarily queue up incoming requests, and which delay one's own response time.

This optimistic notion was shattered when developers were congregated in efforts to define the overall process of development in their organization. Having been primed by management on the virtues of "phased development" processes[53], this gathering of specialists was expected to be a semi-straightforward task of visually assembling or

---

[53] Known by a variety of names in the field, a "phased" development program is one in which sets of detailed development activities proceed in orchestrated stages or phases. The next phase does not begin until all the activities of the prior phase have been complete and design status/direction has been agreed upon by all, during a "gate" review. It is currently the "target" approach among many large development organizations today, because it promises to offer product design stability upon completion of each phase. At no sites, regardless of "official" statements, was the phased approach fully implemented.

connecting the processes[54] of each specialist's area. The result would be a complete "picture" of how the entire development organization worked. Even if individual processes were recurrent in nature, a high-level view was expected to be a simple, sequential ordering of such detailed processes. The "development process definition team" (DPDT)[55] began with this approach, even enlisting the knowledge and experience of members of the team who helped create the process chart shown earlier in Exhibit E.7. In fact, the *lead manager* of this DPDT was a member of the team which produced and published this perspective of the process.

Three months later (roughly 9500 man-hours later) detailed activities had been defined and documented, and one could assert that local processes were understood even better by the specialists who performed them. Understanding of ***integration*** among specialists (i.e., the "complete picture"), however was very much still a mystery.

When attention was focused on piecing this integration puzzle together, however, the phased development paradigm (as shown in Exhibit E.7.) was soon dismissed by developers as irrelevant and, at times, obtrusive to establishing true development behavior. Exhibit E.11. is demonstrative of an early integrative (and high-level) view of this organization's development activities. Supporting documentation reveals approximately 1200 individual functions, with 600 high and mid-level diagrams (decomposition of these basic functions) to demonstrate their interaction.

---

[54] During one such exercise, this was jokingly referred to as playing a professional/adult version of "connect the dots"!

[55] Not its real name. The exercise of team name formation in itself was an interesting dynamic, however, for which there must be some social or psychological explanation. On five separate occasions of examining development, team name development became a non-trivial activity for the first few days, sometimes weeks of the project.

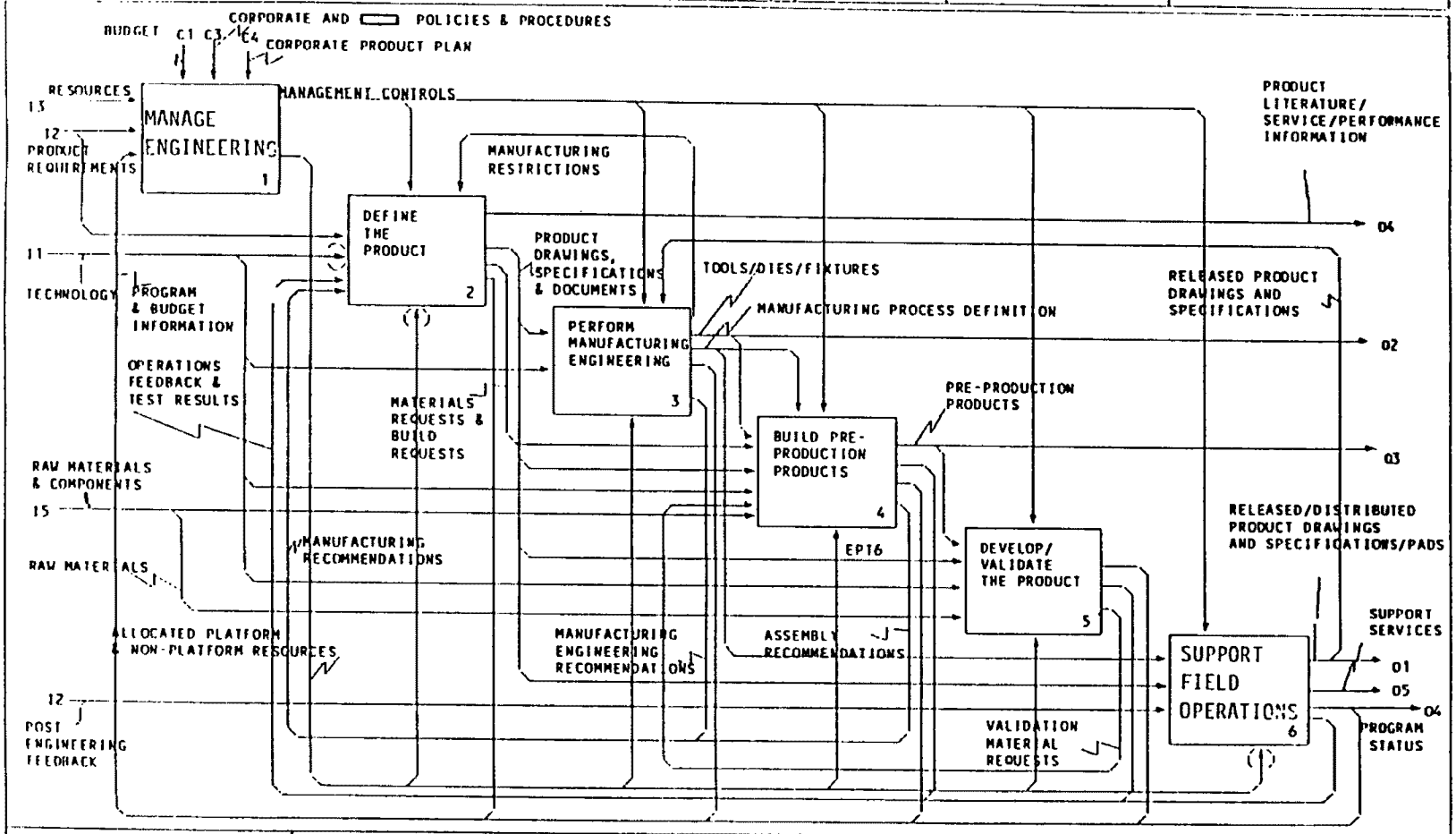
USED AT:

AUTHOR : DPD Core Team
PROJECT: DPD Project
COMPANY: Big U.S. Company
DATE: 1/17/90
REV.: 3/30/90
NOTES : 1 2 3 4 5 6 7 8 9

X	WORKING	READER	DATE
	DRAFT		
	RECOMMENDED		
	PUBLICATION		

CONTEXT: EPI4

A-0



NODE: AS-IS/A0

TITLE: OPERATE DOMESTIC ENGINEERING

NUMBER: EPI5

EXHIBIT E.11

And what about the picture of an integrative, organization-wide "process" of development? After six more months, this picture "grew" to fill 2 large walls of the project "war-room", with continuous haggling over which activities were performed by whom, under what conditions, and with what sequence(s). The picture became so large that ***no individual could carry an up-to-date, comprehensive understanding of the entire "picture"***. Moreover, the "real world" was proving to change faster that team members could incorporate into their visual model.

Within several more months, the original exercise of determining the overall process flow was considered too large a task for a single team. Thus, the team was broken into smaller, more focused analysis teams with the intent to independently improve local processes. Traditional, comfortable techniques would be used to unravel local process puzzles. Thus, reductionist approaches to reducing cost and time (and local, measurable "performance") would be used. ***The overall process had proven too complicated, too complex, and too dynamic to understand.***

It is important to realize that such a description was not unique to a single, specific organization. In every project where formalized modeling was conducted, very similar results were obtained. Of specific concern, participants regularly resorted back to traditional, reductionist techniques to analyze complicated, non-linear systems. Largely, this move was due to a lack of an adequate alternative. Development of non-linear techniques to analyze such a complex system as new product development has heretofore been ignored. The remainder of this report describes the development of a new methodology which could help set a better foundation for analyzing such complex systems.

Appendix F: Time, Cost, and Quality: Observed Measures in Conflict

During our field studies, we encountered three classes of performance measures which are utilized by product development managers, developers, and executives. Though there are various company-specific terms for these, we characterize them as time, cost, and quality. Regularly, improvements in against all these measures are expected. On a practical basis, we found that application of such measures can be inconsistent. This could render the development process a "measure chasing" contest, reflective of the currently fashionable measure in use. At one site, a company which prided itself on ***establishing goals*** (but not necessarily *reaching* them), corporate measures were so fickle that developers mockingly referred to the latest corporate policies as "objectives of the week."

In the remainder of this appendix, we consider some problems with time, cost, and quality measurement.

Time-frame: When and for how long is new product development conducted?
Perusal of the popular as well as scientific press indicates that time is a critical issue in the management of product development, innovative or not. According to participants in this study, time-to-market is not just a critical issue, but *the* critical issue in product development. When examining the steps being taken to shorten product development time, however, a variety of time measures were found to be employed.

It was indicated by several managers that each organization has its own definition of "product development time" (or "time to market"). This has introduced systematic errors in the valuation of product development processes ***from organization to organization***. This fact seems to have impacted the popularized perceptions of development times, particularly in the automobile industry. Today, there is considerable misinformation about the actual product development times found in many companies. Further, this research has found that variations in perceptions about development time occur ***within*** many organizations.

Time-frame *efficiency* measures

A brief sampling of measures used to measure product development time efficiency include the following:

> ***Product Life Cycle (PLC) duration***: The dwell time from release of a product to the market until the product's successor is released to the market. Seemingly straightforward, this measure is popular among researchers and some strategists. As this dwell time shrinks in a number of industries, however, accurate prediction of its length becomes more and more difficult. PLC duration may also have no relation to the other time measures discussed here.
>
> ***Concept Initiation to Start of Production (SOP) time***: The elapsed time from the germination of a concept until the first unit of full-scale production is produced. For some products, where there are very small lot sizes, the production time is included in this measure, as it may consume a major portion of time-to-market release time.

Concept Finalization to SOP time (also known as ***specification-to-production time***): Once the overall concept has been "approved", the meter starts. It runs until the final prototype is compatible with the production-ready tooling. The start point and finish point of this measure are continually in flux and difficult to determine. Why? The concept (in the form of specifications) keeps changing, and the tooling requirements keep changing, even after official SOP dates.

Approved Prototype to SOP time: The time elapsed from "approved prototype release" (itself ambiguous, because prototypes notoriously change after this release date) until production begins. This is supposed to be a measure of production tooling fabrication time, but still involves a significant number of activities which clearly involve product development.

At the day-to-day operations level, numerous measures are used to define the time-related performance of individual activities. These range from an individual's response time for a memorandum, to the order-to-delivery time of a sub-contractor for a major prototype sub-assembly. Numerous types of "micro-measures" were observed in the largest engineering facilities; each manager appears to have his own preferred performance measure(s). In smaller firms, fewer explicit (formal) measures were observed, likely because less administrative time was spent formalizing them. However, managers at all companies do operate with some "custom" paradigms in mind; such paradigms seem to have developed from a subjective mix of their experience, the second-hand experience of

others, and other outside sources, which usually include management policy "how-to" books, of which there are countless variants in existence today[56].

The importance of time-frame as an issue, however, is not limited to the specific time measures utilized by managers. Rather, the ***compatibility*** of such different performance measures is a more alarming concern which was addressed by a significant number of study participants. Specifically, the issue of ***sub-optimization*** of efforts is a theme which needs consideration. In large measure, this issue is the central focus of this study.

A Time-frame *effectiveness* measure

For some firms, the race for efficiency has camouflaged a major ***effectiveness*** measure of development time: ***Market Response Time.*** This is the calendar time which expires from the true market ***need*** for a product to a firm's ***delivery*** of a product which meets these needs. If one considers that the process of accurately identifying a market's needs has a certain lag time, then it might seem intuitive that market response time would be greater than any of the efficiency measures discussed here.

Reality, however, is a cruel judge of this intuition. We all know that customers and end users generally do not wait for their needs to be completely satisfied. They buy products which currently come closest to satisfying their current needs and, more importantly, their

---

[56] An additional interesting observation was the degree of fascination that so many managers possessed with particular authors of such policy books. It was not unusual to speak with a manager who professed to be a disciple of one author ("guru"), and systematically disdain or be ignorant of other equally renowned authors, who in turn were the gurus to other managers. It might prove an interesting study to investigate the dynamics of how managers come to formulate their alliance with certain authors. How much of this association is emotionally driven, how much is peer driven, and how much is due to intelligent, objective analysis is very much open to question.

wants[57]. Imagine waiting 3-8 years for an automobile manufacturer to specifically develop a car to satisfy your currently expressed needs; by the time the development process was complete, your needs (in terms of styling, price, economy, seating capacity, performance, etc.) will likely have changed. Upon delivery, that car may have far less utility and value to you.

In practice, particularly for consumer products and services[58], market response time can be exceptionally short, much shorter than the Concept Initiation to SOP measure; at times, the appropriate product (from some firm, if not "our" firm) is launched just as the market need emerges.

> ***Case in point***: Immediately after (and sometimes during) a championship sporting event, vendors can be found *within the arena* selling paraphernalia and clothing which exhibit the winner's feat. Such items were in development for weeks prior to the actual event. Yet, the fan can purchase the product immediately after the event, while emotions and "needs" are still high.

This points to the firm's requirement to accurately ***anticipate*** market needs well before the market has such needs. If the firm's anticipation is correct (or close to correct), and far

---

[57] The issue of customer needs vs. wants points to a whole cornucopia of research which is beyond the scope of this study. Demographics, psychographics, and a slew of other market analysis techniques have been and still are employed to attempt to determine what and why a customer will buy. For our purposes, we assume that customers know what they need and pick the product which best serves those current needs. Thus, for us, needs and wants are considered equivilant.

[58] For many custom developments, Market Response Time is indeed quite long, often significantly longer than any of the efficiency measures indicated above. This is particularly evident, for example, in weapon system development, where product development projects can extend for more than 15 years. Over these long developments, perceived needs are notoriously in flux; this characteristic is often cited as the major reason for extended development time.

enough in advance (at least as long as it takes to develop and produce the product), then the firm's product and the market needs are well-matched, a condition well-suited to favorable sales and profitability. If the product is less well-matched to market needs (in features and/or timeliness), then the firm can expect far less success in its "new" product.

In a few cases, it is observed that firms run into sales difficulty because their market response time is, *de facto*, ***negative***: the product is released before the market is "ready" to purchase. In these rare, but significant cases, the firm is forced to place an additional burden upon itself: it must perform extraordinary marketing which explains the product and its benefits to an unwitting market. In the business literature, this is the role of a product ***leader***. Once a leader has paved the way, however, numerous followers (other firms) can often easily jump into the market, due to far lower entry barriers (of development and market education). Naturally, this sets the tone for the traditional leader-follower strategy analysis which many firms still subscribe to.

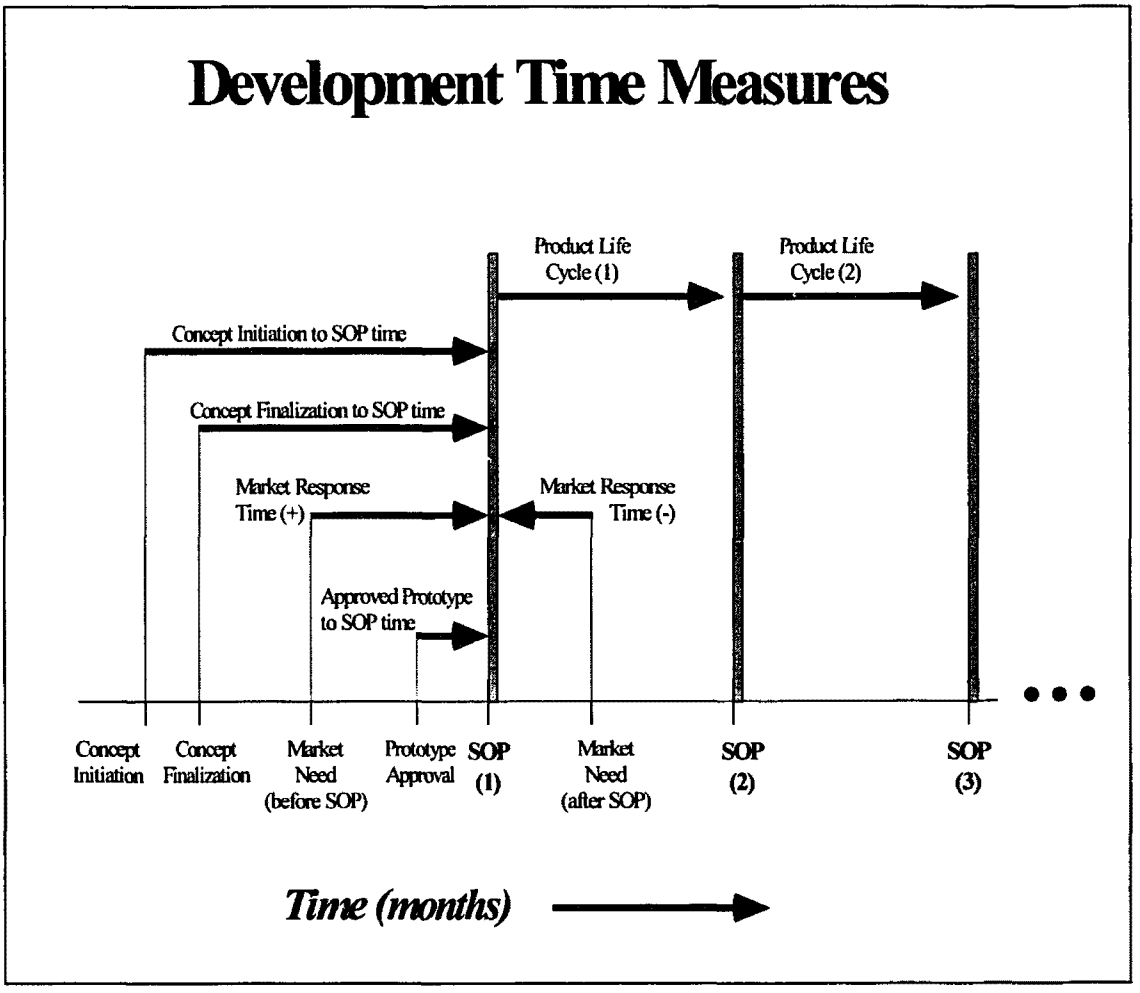
Given the alternatives, it seems logical that best long-term profits should come from matching the timing of market needs to product release date. This implies targeting a market response time of zero. Releasing product "too soon" requires higher market cultivation cost, while releasing it too late makes one just a follower. If one must err, it may be best to "buy a little marketing insurance" and become a leader with slightly negative response time. This may also prove to be appropriate from a market defense perspective; a leader which can behave as a "benevolent monopolist"[59] can prevent

[59] A "benevolent monopolist" is a term developed by Joseph Schumpeter to describe a leading firm that cuts its prices and/or develops more new products before follower firms can enter the market. Thus, such a firm defends its market position by offering ever increasing advantages to its customer base. Peter Drucker considers this one form of "Entreprenerial Judo". Reference Drucker (1985).

followers from invading his turf (the market), nullifying the advantages of others who wait and try to be close followers.

In Exhibit F.1., the time measures discussed here are illustrated. It should be understood that this illustration is not necessarily proportional to the time measures which a given development manager may actually observe. In practice, these measures vary considerably. In this vein, also recall that PLC duration or Market Response Times may have no correlation with the other measures discussed here.

EXHIBIT F.1.



It should be noted that performance measures relating to time-frame are but a subset of the total performance measures used for analysis of the development process. Two other major issues indicated were "cost" and "quality". As with time measures, the metrics used for these two issues also contain some ambiguity and viewpoint biases, as we shall discuss next.

Cost: What is a reasonable expenditure on new product development and how is such expense measured?

As with time, the cost of development is a perennial issue among development managers. It was observed that development engineers view budgets as a necessary evil which constrain their ability to develop products in a "complete" manner. Much of the frustration among engineers, however, does not stem from the existence of budgetary constraints. Rather, it is the fashion in which such constraints are developed and, ultimately, allocated. Some specific sub-issues observed include budget ***allocation procedures, timing, integration, accuracy,*** and ***predictability*** of costs. We briefly introduce each of these issues in the following pages.

Budget Allocation Procedures

When engineering budget allocations are developed, there is traditionally some discretionary pool of anticipated funds against which they are drawn. Based upon such considerations as previous year's engineering expenditures, anticipated corporate revenues (and cash flow), and rough-cut estimates of changing market requirements, some aggregate development budget is established. Lacking detailed activity requirements, allocations are made to specific engineering departments on a best-guess

basis. These guesses are based upon a variety of factors, such as the size of the department's internal requests for authorization of funds, the persuasiveness of departmental management, previous years' departmental performance, and the estimated criticality of certain development activities, given the anticipated financial and market environment.

It is not unusual for participants to feel that allocation biases exist towards activities which are linked to production, or for projects which have been underway for a significant time period. It is clearly perceived that sunk costs are a real, major consideration in allocation procedures. In very few cases in this study did participants (specifically, developers and managers) feel that their allocations were appropriate for the activities expected of them. Of course, most felt their allocations were insufficient. There were a notable few managers who recognized (or at least admitted) that they had more resources than they really needed to perform their tasks. They felt that they were better off quietly spending what had been allocated, however, or risk being under-allocated in the next allocation period.

As a related outgrowth, it is not uncommon for developers and managers to budget capital expenses and personnel expenses separately. Since payroll expenditure tends to be well structured according to the number and "rank" of employees, and since payroll makes up such a large segment of development cost, it becomes very convenient to consider "headcount" as a primary surrogate for departmental expenditure. Now consider that managerial efficiency is derived as a function of departmental performance and headcount (more performance per unit headcount means better efficiency): Managers are compelled to reduce headcount wherever reasonably possible. Yet, certain activities still

need to be performed. The result: development managers hire contract labor, which comes out of the capital budget, not the personnel budget. In some cases, such contract labor cost upwards of twice that of a "normal" employee[60], even though they were performing the same functions! Yet, efficiency, as measured as a function of headcount could actually increase.

It is apparent that budget allocation procedures are "top-down" in nature, while development activities are performed at the lowest levels to produce "bottom-up" results. Though this control architecture appears to be workable in manufacturing operations, it can perpetrate problems for innovative (and not-so-innovative) development operations.

A recurrent source of the problems between "budgeteers" and engineers has been the unpredictability of new product development needs. Neither engineers nor accountants, regardless of experience, can accurately assess all the needs of a development, *a priori*. Individual crisis resolutions are often being made on the fly, without time for long financial deliberations. Thus, financial analyses and resultant budgets are made on gross simplifications of the totality of activities (usually approximated by headcount) which compose development. Some other distinguishing factors between "top-down" and "bottom-up" philosophies observed in the field studies include those in Table F.1.

---

[60] Actually, such contract labor was often merely a former employee, who knew the company, its people, and its projects very well already, didn't require training, and sometimes even retained the same office as before!

Table F.1.

Comparison of Viewpoints

"Budgeteers vs. Engineers"

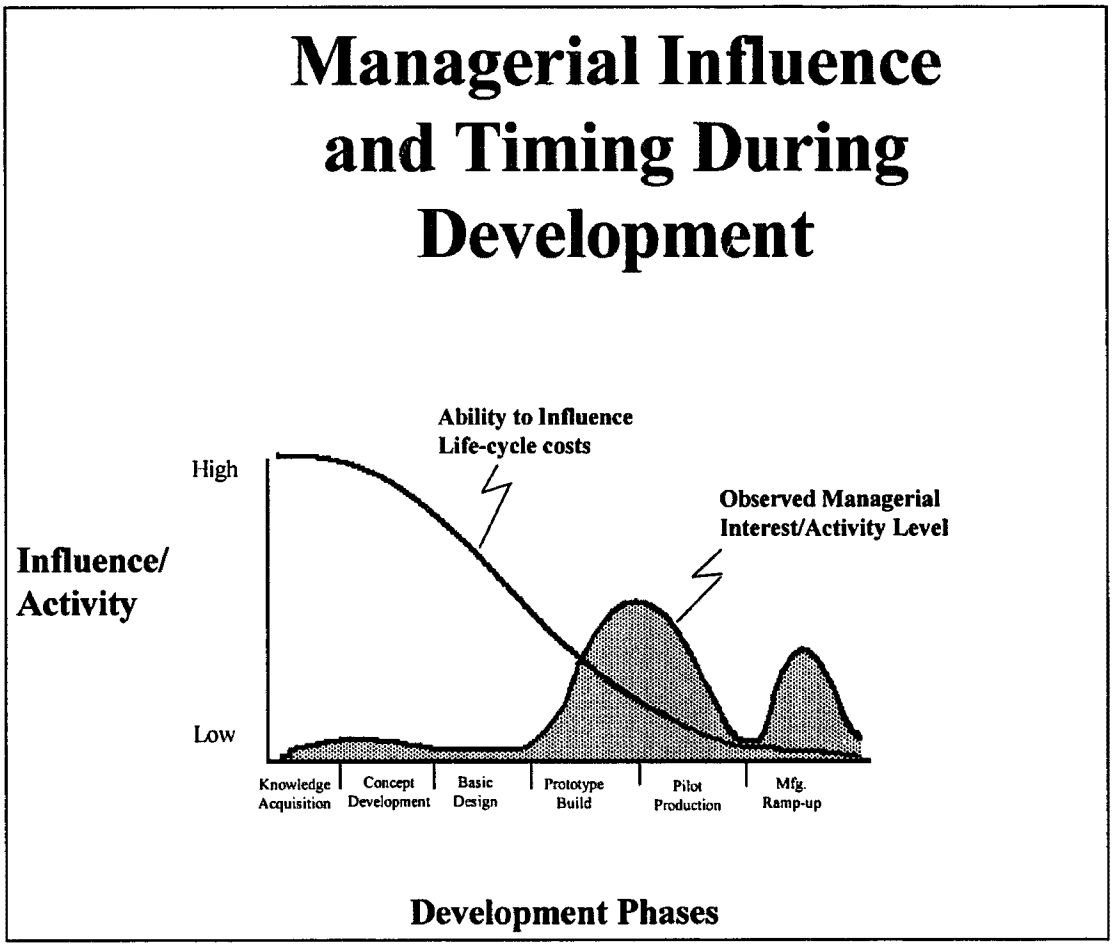
	Top-down (budget allocators)	**Bottom-up** (development personnel)
Resource estimates	deflationary	inflationary
Development philosophy	discretionary	radical
Viewpoint of development activities	reductionist	interdependent
View of processes	linear, complete	non-linear, fragmented
Experience/Background	abstract/illusory	practical/concrete
Perceived resource needs	static	dynamic

Budget Timing

Another bane of the development manager is that funds for development activities are notoriously not available ***when*** they are most needed. This points to the problem of incongruent assessments of timing between developers and budget allocation personnel. As mentioned earlier, there was a predominant sense among engineers that activities which appear "closer" to production get priority over "early" development activities. This is consistent with the findings of other studies (McAfee, 1991; Hays & Wheelwright, 1988; Gluck & Foster, 1975), which independently realized both the inverse relationship of cost impact to development time expenditure and the development interest by

management --too late in the development process late to have significant impact. Refer to Exhibit F.2.

EXHIBIT F.2.



A corollary to this oft observed fact is the observation that ***high-impact decision-making*** (and thus high-impact costs, when one considers preliminary prototype builds as part of decision-making) ***occurs very early in the development cycle.*** In a defense system development (McAfee, 1991), it was approximated that 85% of total product life cycle (PLC) decision-costs reside in the concept development and preliminary design stages of

the life cycle. Yet, our current study has revealed that, once again, minimal management emphasis is placed on such early stages of the life cycle, much less development. At times, managers indicated they would rather deal with the problems of a known, pseudo-stable entity (current product or soon to be released product) than provide direction to a new, unfamiliar, unproven, and dynamic concept. Given an association between managerial interest and budgetary allocation, it is no wonder that the concept of ***product championing*** has had such high exposure in the formal and informal training of product-development managers.

Budget Integration

The involvement profile of management points to another sub-issue of innovative development cost: ***integration***. Much has been declared in the research and professional literature about the need for activities within an organization to relate appropriately to one another. Terms such as ***enterprise integration, interfacing, activity networking, re-engineering, CIM*** (both Computer Integrated Manufacturing and Corporate Information Management), ***CASE*** (Computer-Aided Software/Simultaneous/Systems (pick one!) Engineering), and others have been developed to describe particular facets of this integration problem.

In manufacturing, the methodology of performing detailed activity analyses, followed by development and implementation of an integrative control architecture has enabled significant cost reductions, particularly in the areas of set-up, materials handling, inventory, and scrap reduction. Such a methodology, though appealing, has not been successfully applied to development activities. A major reason for this seems to be the lack of ***repeatability*** in development processes, a characteristic which is predominant in

manufacturing. Even for manufacturing lot sizes of one, a control architecture can work because the process is predictable. In innovative product development, predictability has not been a convenience generally available to the development manager or developer.

Accuracy of Assessing Costs

By understanding the activities which are performed during new product development, however, this study has found that significant cost reductions can be found. Here, we have learned some lessons from the manufacturing sector. As automation has entered the manufacturing environment in larger numbers than ever before, the cost justification processes have become significant--and questionable. As the direct labor component of production continually drops (some estimate that direct labor is less than 10% of cost for some plants) labor is increasingly becoming a less substantial cost. As a result, traditional labor-based cost accounting systems are becoming less and less relevant. (Recall that cost accounting was formally established by David Ricardo in England over 200 years ago to analyze the labor-intensive process of harvesting corn (Vangermeersch 1986)).Thus, the allocation of overhead to direct labor is becoming less and less appropriate. Such sentiments have actually been expressed by accountants at NAA conferences!

Through the use of Activity-Based Costing (ABC), manufacturers and their sales and marketing personnel have begun to get a better grip on their actual production costs. Of course, ABC is rooted to satisfactory activity-based cost accounting, which is itself dependent on accurate activity identification and assessment.

This is where integration and accuracy of costs come together as significant cost assessment issues. It was observed that many activity-based cost accounting projects for

product development processes failed to produce meaningful, representative assessments of actual development activity. This can be attributed to the reductionist viewpoint of rigid activity identification strategies. By attributing ***similar-looking*** activities as the ***same*** activities, rigid hierarchical activity structures make undue assumptions about savings from elimination of activity redundancy. Surely, there are many redundant activities which can and should be pared from development processes. There is little gain from "reinventing the wheel" over and over again, aside from perhaps the learning processes when young engineers are, de facto, in apprenticeship mode. Such redundant activities are often masked, however, through mislabeling and/or over-simplifying, and thus are misinterpreted by management (and many zealous management consultants!) "looking" for places to cut and chop the process. It is observed first-hand that this can have the negative consequences of throwing the proverbial "baby out with the bath water"; value-added activities are cut along with those of negative or zero value. If this continues to be conducted as blindly as has been observed during this study, then currently fashionable "process re-engineering" is primed to take a major hit, just as the infamous industrial engineering time-and-motion studies of decades past[61].

Recent efforts by a few sites appear to be partially alleviating this reductionist problem, by focusing on both activities *and* interfaces between activities. This provides process analysts an additional dimension upon which to judge excess redundancy of effort. As the sophistication of interface analysis improves, it may be expected that time-dependent

---

[61] Aside from such technical limitations, there are also psychological and cultural problems with scrutinizing activities in detail. I have run into situations where managers do not want to feel that their analysis has been responsible for the unemployment of others. In a few cases, it is highly suspected that managers are fearful of finding that ***their*** activities are not value-added, and that they could become victims of their own analysis!

interfaces will be recognizable. When automated sufficiently, this will provide managers with much better, dynamic cost and value-added activity analysis tools.

Development Cost Predictability

Yet, such analysis tools, when developed sufficiently, will still only be of an *ex post facto* nature. It may provide accurate assessment of what ***was*** spent, at what times, for what purposes. It will not provide ***predictability***, however, of what will happen during the next development process.

It is with this need in mind that the dynamic Complex-Process Path (CPP) was developed in this study. As discussed later in this work, there can, at times, be observable trends which permit better *a priori* management decisions. Failure to recognize such trends, however, can result in chaotic-looking development processes which are beyond the forecasting capability of even the most gifted strategists, managers, or developers.

Quality: How is development performance judged?

Defining Quality

An examination of the effects of quality must begin with an understanding of what is meant by "quality." According to Webster, quality is "any peculiar and essential character." As it applies to the innovation, development, and manufacturing environment this definition is excessively broad and non-specific. In the current environment, there seems to be a prevalent sense that one can successfully develop "scales" for quality--that

one can determine the "quality level" of particular products and/or the development or manufacturing processes behind such products.

Perhaps a better definition can be derived from concepts presented by Towey (1988). In his review of the work of Morse, Roth, and Poston (1987), he suggested that there are three primary determinants of quality in the manufacturing environment:

- The first determinant is ***customer expectation***; the degree of excellence or uniqueness that some downstream customer expects from the finished product. In development, "product" may refer to the final, manufactured item or any intermediate prototype assembly. Likewise, "customer" in this context may range from end-user to any participant within the development process.

- The second determinant of quality is ***product specification***; the extent to which the engineering/design of the product is actually of merit.

- The third determinant is ***actual results***; the degree to which the product, as manufactured, matches its product specifications.

Though all three of these "determinants" clearly exist, they cannot be presumed to be universally equal (indeed, they are rarely congruent). Some precedence needs to be established.

As Whiting and Walsh (1986) have indicated, quality is in the eyes of the beholder. Customer expectations and perceptions may be the main drivers for quality. Product

specifications and results should follow this lead, for their inputs should depend upon the wants and needs of the customer. The goal is for the end product to match customer expectations. "Quality," as an entity, is a gauge of how well results match expectations. Products and processes that are close to or even beyond expectations are perceived as "high-quality." Products and processes that fall short of expectations are deemed to have moderate or low quality.

A word to the wise is in order, however, particularly to those advocates of closely following the "voice of the customer." At a few notable development sites, developers were forced to think several years ahead of the currently available technology and start designing to an ***expected*** technological feasibility window. One of these sites, known for extremely high quality products and possessing a long-standing reputation for innovative consumer applications of new technology, had an additional problem: since their lead times are quite long, and product quite complex, their personnel (specifically strategists, managers, and developers) found they must deliberately ignore certain customer requirements, which they feel will evaporate by the time the product is released to the market. Rather, their need for requirements is more technical and inbred in nature. Often this technical requirement is never even understood, much less voiced by the customer or user. Said one strategist: "We cannot wait for the customer to tell us what he wants, because he doesn't always know what he wants! We show the customer what he wants, when he sees new concepts on our product."

Note how this relates to the concept of negative market response time, discussed earlier. Under several documented cases, organizations found that the developed products preceded market needs (actually, market ***wants***) for the product. In some of these cases,

the market grew to understand and accept the features of the product, resulting in revolutionary market creation and much sales success. In others, the market never accepted such internally generated product features, resulting in lackluster sales. Thus, certain "leap of faith" judgements about future market requirements has been a necessary gamble for successful innovative new product development.

Note that the perception of quality can also vary at different stages of the product life cycle. Finished products have certain perceived qualities to end users, while distributors/jobbers/dealers have their own perceptions about the quality of products and the processes surrounding those same products. For example, manufacturers are concerned about the qualities of both raw materials and the processes used in converting raw materials into finished goods. Suppliers may worry about many of the same quality issues and quality associated costs as the manufacturers they are supplying. Designers are observed to have a slew of detailed quality measures; often these are established through engineering standards societies and, more recently, governmental regulations.

Cost and Time Effects of Quality

Quality affects both the revenue and the cost sides of the company balance sheet. Though there is limited empirical data on quality cost effects, there has been considerable subjective discussion on the subject. Researchers have even developed their own nomenclature for the costs of quality. According to Towey (1988), "quality costs" are "those costs incurred because poor quality does or may exist."

Four major categories of quality costs have been identified.

1. Prevention costs

2. Appraised costs

3. Internal Failure costs

4. External Failure costs

Roth and Morse (1988) define each of these categories as follows:

"***Prevention costs*** are [those costs] incurred for establishing, implementing, and executing projects to prevent errors and defects from occurring. Included are the costs of quality planning, standards development, and training programs.

"***Appraised costs*** are those incurred for identifying defective materials and products. They include costs of sampling, inspecting, testing raw materials and finished goods to determine if they meet quality standards.

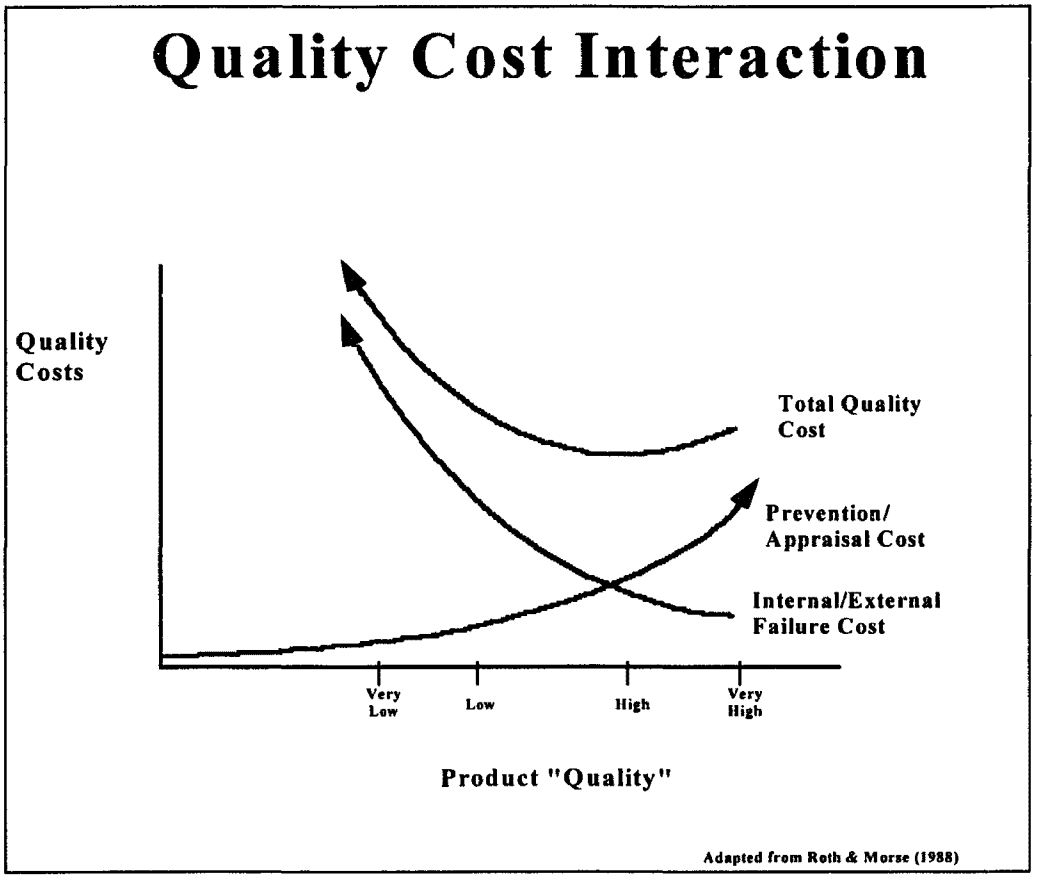
"***Internal failure costs*** are incurred because products identified as defective before shipment to customers are repaired, scrapped, or sold at reduced price....[They] include the cost of rework, repairs, scrap, and downtime occurring because of production failure.

"***External failure costs*** are incurred because defective products are sold in the marketplace as conforming goods. These costs include the cost of fulfilling warranties, repairs, and product liability cases. They also include the cost of lost customer goodwill."

Clearly, product development activities can have major impact on every one of these quality cost components. This is a point that seems to be ignored by some financial analysts, when they balk at the high costs of quality product development.

Cost accountants have begun to look at the relationship between rejection rates and quality costs. Exhibit F.3. demonstrates the qualitative nature of Prevention and Appraisal (PA) and Internal and External Failure (IEF) costs. Clearly, as PA costs rise, one expects the reject rate to decrease, as products are designed and manufactured better. Additionally, as PA costs rise, the need for IEF failure costs should fall. JIT systems require consistent materials inputs, and TQC/TQM philosophies are a step in this direction.

EXHIBIT F.3.



Godfrey & Paseank (1988) and Roth & Morse (1988) have indicated that there should be, for each product, a conformity level that optimizes total quality expenditures, where TQC is the sum of the four major cost categories (TQC = P + A + I + E). In the same breath, however, they acknowledge two problems (to date) in this analysis:

1. Quality-cost relationships ***change over time***,
2. Objective, quantitative measurement of many quality costs are ***not readily computable or assessable***.(e.g., How does one accurately assess the effect of design or production quality on goodwill?)

Further, as the cost of quality changes over time, it may also change across the product line. This can produce tough problems for managers of CIM/FMS environments. In the multi-product design environment, this dynamic effect can breed competition and frustration among design teams.

The natural question arises: how does one attribute quality costs to specific design activities? Invariably, a significant time lag exists between development, production, distribution, and consumer utilization of the product. Pegging faulty development activity as a culprit in external failure costs is easy. Correctly ***specifying*** what could have or should have been done differently, *a priori*, however, appears to be next to impossible. With cost analysts breathing down the necks of developers, it may be easy to see why some engineers feel they are in a "no-win" situation. As discussed in section A.5.2. of this chapter, the earliest stages of design have the highest impact on life-cycle product costs, including quality costs. Yet, as design processes are financially cut to the bone, some engineers feel they are given less opportunity to invest in quality design activity.

Quality assessment seems to be an integrative problem, intricately inseparable from cost and time considerations. Under some scenarios, increased quality is perceived as proportional to the cost and time expenditure of design. Yet, it is also judged by many studies and managers that cost and time expenditure can be ***reduced*** through

improvements in design quality. How can both of these seemingly conflicting statements be true?

We suggest the hypothesis that each are true under different conditions. Realizing present and upcoming conditions in an accurate and objective manner, without undue bias towards "the way things used to be" or "the way I think it is", is a very humbling and, at times, frustrating task. Yet, it is part of the task of environmental scanning, which is necessary for prudent management decision-making. As the ever-changing "system" of new product development increases its metamorphic rate, this condition assessment is proving to become more and more difficult.

Appendix G: Documentation Techniques for New Product Development

The Concept of "Process Analysis"

The diversity, magnitude, and complexity of the many observed new product development activities provided a wealth of potential information to examine. As with any data collection and analysis project, it was prudent to attempt to isolate significant findings (drivers) from incidental "noise" (or symptoms). In this effort, some data triage was inevitable. The basis for such action, however, had to be both ***agreeable*** and ***objective***.

- The first characteristic, ***agreeability***, was necessary to prevent the study from being accused of using selective, self-serving data which only demonstrated a stratified subset of field activities. This is an honorable use of agreeability. It has been the stance of this study that such agreeability need go no further than data acquisition and filtering. Results and interpretation of results were not dictated by what was paradigm-friendly, but rather by what "made sense" according to the data presented. This, we believe, helped preserve the need for objectivity.

- The second characteristic, ***objectivity***, is supposed to be the watchword of *all* analyses. Nevertheless, subjective judgements seem to enter many analyses, particularly when specific time and results "constraints" have been pre-ordained on the researchers. Traditional stereotypes suggest that academic research slants towards objective study, while managerial analysis is inclined to accommodate more subjective considerations. Such prevailing images, while clearly not

universally true (and, at times, even appear inverted to actual observation!), seem to affect the relationship between the worlds of academia and management. Since this study incorporates the experiences of many experienced managers, objectivity was a more focused concern than might be warranted in other studies.

Unfortunately, many existing field methods for assessing new product development were observed to be far from agreeable *or* objective. Quantitative as some observed methods were, many had roots in what may be called "organizational theology." Many methodologies were steeped in tradition--"handed-off" from preceding management with little consideration of ***why*** they may have (at some earlier time) been useful measures. Naturally, several different "pet" methodologies could be found from department to department. This resulted in methods and measures which satisfied local (departmental or personal) objectives, regardless of the organizational objectives. Efforts to reconcile conflicting methodologies satisfactorily would, in many cases, appear to be a monumental task, to be done with only a few secular (and bold) individuals.

Moreover, the ***process*** of new product development was not well understood, nor agreed upon. This could be partially attributed to the fact that many development activities are not physical in nature, but rather cognitive (i.e., more thinking, less action). Because each department manager had his own set of objectives and metrics to assess his department's performance, there was little incentive to look at the overall process. The effective general attitude was: "that's not my job--that's executive management's job." The resulting ignorance, coupled with a certain degree of managerial loyalty, appeared to further this isolationist (and sometimes confrontational) condition.

Contrast this with many manufacturing operations, in which production processes are fairly well documented[62] and quite tangible. One can usually walk into a manufacturing facility and directly *see* the process. Though it may be quite overwhelming (e.g., a steel or paper mill) or relatively modest (e.g., a transistor doping or rubber molding facility) in size, each facility tends to have a well-defined manufacturing process which one can follow, either visually and aurally, or with the aid of computer-based monitoring equipment. Obviously, knowing the existing process can provide a strong basis for conducting constructive analysis. For instance, locating the "bottleneck" in the process is relatively simple once one has an appropriate, visual representation of the process. Results and suggestions from such analysis can then be graded, based upon how well they serve to improve the process.

It was with this intent (i.e., developing ***process understanding***) in mind that documentation of new product development was to be undertaken. When this task is successfully completed, subsequent process analysis could provide managers with concrete solutions to the many problems facing development. Specifically, the problems of ***responsiveness***, ***cost***, and ***quality*** of product development were to be examined. Based upon a clear, objective "as-is" process, an analysis team could determine the appropriate (relevant) measures which mattered most. Then, given some overall objectives, process managers could "tune" the process to best meet such objectives.

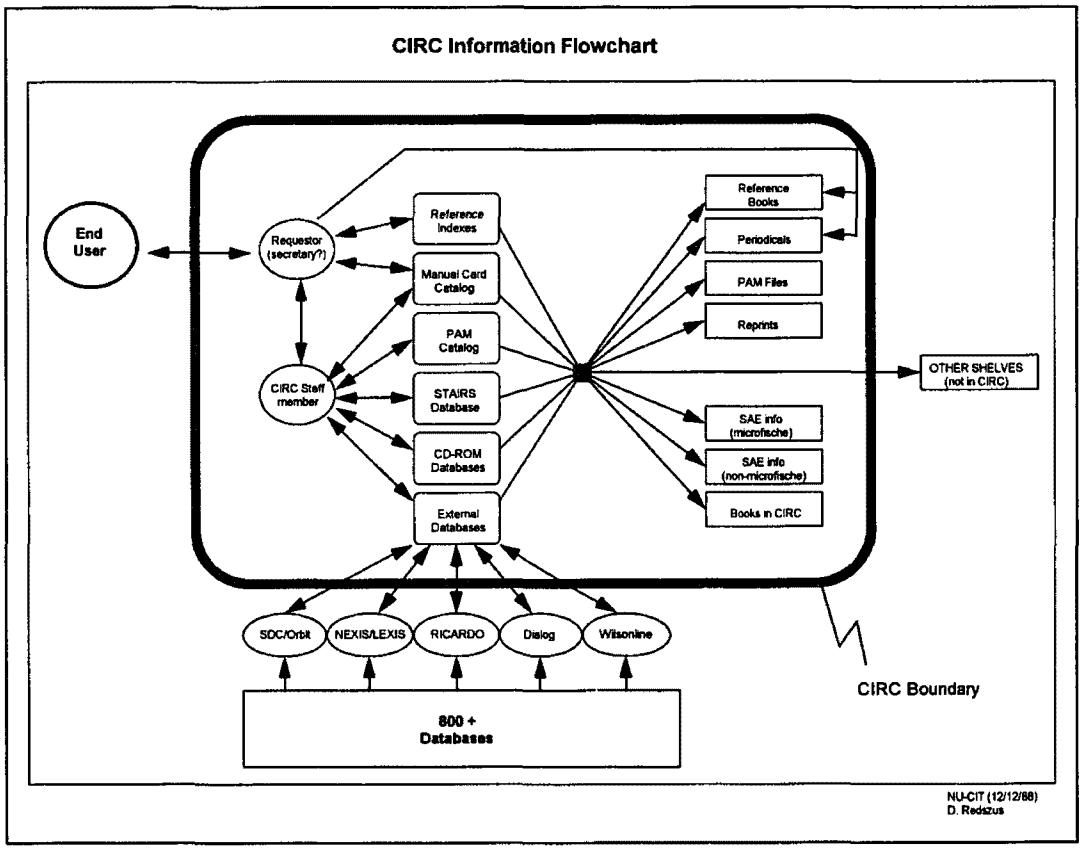
---

[62] Many manufacturing managers would likely disagree with an assertion that their processes are documented ***adequately***. In fact, many manufacturing facilities conduct on-going process re-engineering activities, to make sure that their understanding is as up-to-date as possible. While such activity is far from complete (and not as universal as many would hope), manufacturing processes tend to be much more ***documentable***, and subsequently more documented than new product development processes.

Basic Process Documentation Considerations

The concept of examining the process of new product development did not originate with a firm modeling framework. Rather, it began with the very simplistic concept that one should be able to draw "the process" on a single sheet of paper. During visits to Site #1 (an Automotive Engineering Library), it became apparent that this simple drawing was not readily available. Since Site #1 analysis was essentially a pilot study, to further refine the thesis topic, a rather simple modeling technique was developed. This methodology considered the various information processing locations as ***stations***, and any flow of materials as ***information transfer***. Reference Exhibit G.1., which is one of several simple information retrieval flow diagrams developed from visits to the library site.

Exhibit G.1.



As this research progressed, drawings and analysis of major development processes grew considerably more sophisticated. Remarkably, however, this simplistic visualization was not appreciably deviated from over the next four years. Thus, the concepts of ***stations*** (also referred to as ***functions*** throughout this report) and ***information transfer*** were robust descriptors of the actual elements in product development.

Upon the arrival at Site #2, a more structured modeling technique was utilized. For the remainder of this research, the IDEF methodology was utilized as a basic documentation tool, with specific data collection techniques supplemented on an as-needed basis. A specific subset of the IDEF methodology[63], IDEF0 is a functional modeling structure which, in concept, is similar to the simple methodology presented above. Its appeal for this analysis was multi-faceted. Favorable attributes included structure, simplicity, flexibility, popularity, and availability. We address these attributes below:

Structure: The IDEF0 modeling technique has a rigorous structure (e.g., symbol meanings, sizes, codes, etc.) which facilitates its use across many departments of an organization. With this structure, independently derived models of separate organizations can be "hooked up" to one another. When multiple interfaces between organizational activities are apparent, the IDEF0 model can accommodate this as well. The IDEF0 structure ***does not depend on the hierarchical structure of an organization***. Rather, it is structured according to the ***commonalty of functions*** in the organization.

Simplicity: Model(s) developed with this method are simple to read, given a small amount of training. This is a direct result of the ***decomposition*** structure of

[63] For information systems developers, IDEF1 (and IDEF1x), is a standardized entity-attribute(+ relation for IDEF1x) documentation methodology which can be used to trace the association between fields of databases of varying structures.

IDEF0. In the matter of a few minutes, a reader can readily derive an overview of how a process operates, and gradually investigate to any detail which he may wish. Creation of such models is naturally more complicated than reading them, but can be done with only moderate training. Only a few basic techniques can cover the vast majority of modeling needs.

Flexibility: IDEF0 models can be developed for ***any*** organization, large or small. In fact, the model developed for the Army Materiel Command covered the activities of seven organizations simultaneously. This is due to the functional/functional-interface orientation of the models, not an organizational structure.

Widespread Use: A public domain methodology, IDEF is a well-established technique throughout much of US industry. Originally developed by the US Air Force as a methodology for documenting aerospace manufacturing operations[64], it is now used in selected European organizations, as well. Recently, IDEF0 was declared the approved method by the DOD as the principal tool to be used in business process re-engineering (BPR).

Software/Analytical Availability: Perhaps the most appealing aspect of IDEF today is that it is available for use on several computer platforms from a variety of vendors. When used on a PC, for instance, the IDEF-ine Family^{TM} software worked very well as a field documentation tool. Because of their functional orientation, IDEF0 models currently provide natural foundations for cost accounting, PERT analysis, QFD analysis, and some selected (linear) simulations. Recently, large strides have been made in establishing an IDEF Repository, in which thousands of locally developed models can be downloaded for use by any member of the repository. As an educational tool for employees and managers, this has significant potential. When the CPP model presented later in this study is incorporated into a parallel processing computer architecture, the analysis capability of special (iterative) IDEF0 models will be taken to new heights.

---

[64] IDEF is an acronym for **I**CAM **DEF**inition, where ICAM was the **I**ntegrated **C**omputer **A**ided **M**anufacturing Project at Wright Patterson AFB.

In addition, the IDEF methodology provides a basic framework, or "skeleton", upon which specific findings and nuances can be hung. This is accomplished through the use of detailed project glossaries and textual descriptions which accompany every visual diagram (per the structured rules of the methodology). Thus, IDEF0 was a convenient structure to help document large-scale operations in an objective, consistent manner. This was found to be particularly helpful for documentation teams which were made up of several diverse process participants. In such cases, each team member was responsible for developing and verifying major sections of the integrative model of the overall organization under study. Thus, the structure, simplicity, flexibility, universality, and availability of this methodology provided a unifying environment for process documentation.

This thesis is not about IDEF, however. It is about some specific findings which arose through the use of IDEF in documenting the "process" of new product development. However, the characteristics inherent in the IDEF methodology facilitated an insight to the process which would have been extremely difficult to obtain otherwise.

Perhaps the most dramatic aspect of this insight was the acquired capability to ***simultaneously*** "see" development processes from both global (hi-level) and local (low-level) perspectives. Because of the functional decomposition inherent in the modeling methodology, it was (in principal) relatively simple for the trained eye to follow the transfer of ***elements***, information or prototype assemblies, from function to function (or, in some cases, from department to department). Thus, such models indicated movement or, more precisely, the ***channels*** over which movement of elements could occur.

The concept of movement, or "flow", throughout an organization enabled a unifying theme upon which this analysis was based: that ***product development could be characterized as an "accumulation" process***. In this self-described process, a variety of ***inputs*** (broadly categorized as raw materials) are gradually transformed into one or more organized ***outputs*** (which may be composed of the "physical prototype," supporting "prototype documentation," and any remaining materials (discarded or not) which are not encapsulated in the above two). Along the way, there are certain ***controls*** (marketing requirements, machine capabilities, government regulations, company policies, social customs, laws, budget allocations, etc.) which direct and limit the process. There are also certain ***mechanisms*** (property, human beings, machines, funds, etc.) which enable or propel the process. Thus, inputs gradually accumulate with one another, according to the directions outlined in the controls and with the facilitation of the various mechanisms. The end results, the output(s), are a reflection of such efforts. The precise way in which this all happens is the ***process***.

Documentation of the process consisted of three basic steps:

1. ***Identify and categorize the major functions*** which are performed in new product development;
2. ***Identify and classify the critical entities*** (inputs, controls, mechanisms, and outputs) which are utilized and created during new product development;
3. ***Build and verify a model*** which conveys the association of the functions from step (1) with the entities from step (2).

Thus, documentation efforts were not limited to ***identifying*** the entities and functions of new product development, but extended to finding out ***how*** the various entities connected the many functions. As a result of such effort, it could also be possible to see how functions ***affected each other*** during the course of new product development.

Early Process Documentation

Although many prior efforts had acknowledged the magnitude (complication) of the development process, there was an over-riding sense that the process of product development could be documented. Before any model(s) could be developed, however, objective data collection about the process had to be performed. This task predominantly involved a substantial collection of process participants.

At Site #2, for instance, the core process documentation team was composed of 10-12 design engineers and 2 managers, with an average (per person) site experience of 19.4 years. These core team members were not permitted, however, to rely solely on their own experience. Rather, each member of the team returned to their respective specialty "home" departments to collect information on the process from their colleagues. Naturally, this increased the effective team size several-fold. This also permitted more complete and more up-to-date information about how the organization actually operated.[65] This decentralized data collection was a feature of all team-based process documentation efforts in this study.

---

[65] Contrast this with such notions as "...how it ***used to*** operate" or "...how we ***think*** it operates."

As alluded to earlier, questionnaires, interviews, surveys, first-hand observations, and records analysis were conducted throughout this study. For each of the five development analysis projects for which teams existed, distinct formal data acquisition protocols were developed. For the remaining "solo" sites[66], at which particular process attributes were being investigated, information collection protocols were relaxed. Rather, pointed inquiries were made at "appropriate" times to process participants who seemed most able (and willing) to offer direct, "un-filtered" responses. ***In many regards, these unstructured, casual conversations revealed much more about actual process operation than the more formalized (often "politically correct") responses.***

The features of the models generated during early documentation efforts reflected the attitudes of core team members. Because there was a pre-disposed notion among many members that the process of development could outwardly resemble a manufacturing operation (one manager even created a schematic which characterized the engineering facilities as a giant factory!) some of the early models were simple, sequential representations of activities that could be easily translated to a PERT diagram.. Any documented indications of feedback, or "rework", were treated as very special cases which were not part of the linear "main-stream" process[67], and thus were largely ignored. In short, these models represented reductionist views of processes which actually contained significant underlying complexities.

---

66 "Solo" sites are those which I visited first-hand, without the structure of a multi-person team.

67 The analogy of the process resembling the flow of a river was not an extreme one. While any river has branches, streams, stagnation areas, undercurrents, or localized eddies, the "overall direction" of the river was the focal point of study among the core teams. As long as the peripheral effects were small, it was asserted that the overall course of the mainstream could be determined. As we discovered, such a premise was not necessarily valid for new product development.

When many functions operated simultaneously, and/or used many different elements (inputs, controls, or mechanisms) at different times, the simple models became more "complicated-looking." However, the fundamental, overriding belief was that development was merely a collection of simple processes which happened to be overlaid upon one another (we refer to this concept as a ***complicated*** system.). With this thought in mind, complicated-looking functional models could be "untied" to produce a collection of simple, linear processes. There was still little consideration, however, that the system under study had significant ***complex*** (non-linear) undertones.

Visual Process Discovery

Each team-based documentation effort provided very revealing aspects of the organizations under study. Four aspects which were obtained from creation of a visual model included ***multiple perceptions, process "looping" behavior, non-repeating process paths,*** and ***functional vs. organizational structures.*** Each of these are discussed briefly in this section. Subsequently, we discuss the creation of the CPP Structure.

Multiple Perceptions

Revelations about an organization's culture (and isolated counter-cultures) included mutual misconceptions about participant's respective activities, perpetual optimism of the activity capability of colleagues, and an almost mystical delusion that executive management had a well defined (but never available for perusal) agenda for the future direction and control of their organization. This thesis was not focused on the socio-psychological aspects of organizational behavior; it was readily realized, however, that many localized assertions about "the process" (no matter what process) did exist, which

were based upon very little or no concrete evidence. As a direct result, the many perceptions of development processes ***at a given organization*** did not necessarily "fit" with one another.

In the product development organizations visited in this study, participants proved to be adept at describing their own specialized activity areas. When the aggregation of each of these activity areas was composed into a more global model, however, the result was an incomplete picture of the true development process. There were a number of "non-significant" activities at some local levels that proved to be very significant at other local levels. Conversely, crisis issues to some were considered trivial aggravations to others. This appeared to be ignored by many manufacturing-oriented product development managers. This pointed to an alarming, but significant recognition: ***the priority structures among participants who conduct development activities were not necessarily compatible***. When such varying perception were not resolved, latent problems were automatically inserted into evolving designs.

Process "Looping" Behavior

The dormancy period of such problems could cover a broad time spectrum. Some problems never manifested themselves, or not seriously enough during the process to warrant redress. Some problems were recognized and resolved early enough to have little effect on the developmental budget or schedule. Other problems, however, had a knack for emerging at very "inconvenient" times--during late gate reviews, just after release to production, during production, after customer delivery, etc. The significance of such problems can not be overestimated. Ramifications of such process behavior can include increased development costs, delayed production, higher-cost production, increased

warrantee costs, internal strife, supplier disharmony, and perhaps most importantly--loss in customer confidence towards the organization.

When a serious problem arose (i.e., when such a problem was finally recognized), it was natural for members of the development team[68] to proceed "backwards" through the process to identify the source of the problem area. At times, this meant backing up only a few "steps" to rectify the problem. Delays in such instances would be related to the number of retraced steps (activities) and the time necessary for each activity. Likewise, the costs of such "looping" behavior to the development organizations were dependent upon the size (# of steps) and number of loops which were conducted. Naturally, it was in the best interests of the organization to minimize the occurrence of such loops.

Yet, the process documentation teams repeatedly discovered that such loops could be induced at just about any time in the process. A frustrating corollary to this discovery was that each organization had little capability to predict the timing or severity of such developmental loops. Further, team members began to realize that the propensity of their organization to engage in such looping behavior was much more significant than previously believed or hoped.

Non-Repeating Process Paths

Even as some documentation team members became accustomed to the concept of looping process paths in their development organization, another troublesome process

---

[68] Provided the original product development team was still in existence at the time of the problem discovery! As can be easily visualized, many product problems are realized after production has began, after the "original" team has dissolved.

characteristic began to emerge: non-repeatability. Two forms of this characteristic were observed, as it affected both mainstream and looping processes:

1. ***Non-Repeatable Mainstream Processes:*** As development participants documented their process, it became abundantly clear that the process being documented was in transition. Thus, a model of last month's process would look different than today's process or next month's process. Such change could be a reflection of continuous improvement efforts in place by corporate or local departments, technological process changes (new equipment, etc.), new or modified document management systems (MIS), or even changes to organizational structure. It could even be a reflection of the documentation effort itself: as managers saw their development process on paper (during validation sessions, for instance), they might have reacted accordingly and changed or improved their procedures. Further, development processes were seen to be different from ***product to product*** within an organization. This was frequently thought to be a direct result of changes to the formal organizational structure.

2. ***Non-Repeatable Looping Processes:*** Even if a mainstream process appeared stable, looping (or "rework") paths were not necessarily stable. Thus, when loops were incorporated into a process model, it was not certain what activity path would ensue after the basic rework function(s) had been completed. In some cases, reworked development projects were diverted from the original mainstream process and never followed the mainstream process again. Some reworked developments partially followed the mainstream process. In a few cases, the original mainstream process path resumed during and after the rework (as if the

whole process had just jumped back in time). ***Determination of such loop return paths appeared to be situationally specific, and thus was beyond the predictive capability of even the most experienced team members/participants.***

Functional vs. Organizational Structures

The functional modeling projects provided another major visual discovery which may not have been noticed otherwise: ***functional structures and organizational structures of development organizations bear little or no resemblance to one another.***

Since the documentation/modeling methodology was ***functionally*** based, the orientation of the visual model was on ***what*** activities were completed, not ***who*** completed them. (After the activities had been determined, it was a relatively simple exercise to designate which departments/individuals performed them.) This provided a convenient means to categorize the development process, independent of traditional organizational boundaries. For instance, in the AMC project, seven major development organizations at distinct geographic locations around the country could be modeled ***using the same activity model***. To observe the activities of a single organization, one only needed to filter the overall model by site (recall that "site" --in the form of property, workforce, and machinery-- was one of the *mechanisms* illustrated in the model).

This specific capability offered two more observations of organizational performance which likely would not have been possible otherwise: ***functional redundancy*** and ***materiel movement***.

1. ***Functional Redundancy:*** By generating a 2-dimensional matrix which pitted specific functions to organizations/departments that performed those functions, it was possible to see duplicate activities across the developmental "system" of study. Although redundancy is a "built-in" feature of many organizations, it was insightful to many process participants to see that multiple organizational bodies performed the same functions. Conversely, such a matrix revealed which functions were the sole domain of a single organization. Depending upon the criticality and frequency of such single-site functions in the process, their existence raised valid concerns over the viability of a development process should such a local organization become incapacitated[69].

2. ***Materiel Movement:*** When the IDEF0 models were created, the functional architecture provided an easy-to-follow order to the activities performed in development. Because of this order[70] (including the fact that no functional redundancies exist in a functional model), the functional development process was somewhat less dreadful to interpret than an organizational chart. Nonetheless, materiel and information movement within a functional model could be mapped to an organizational model. A sample mapping is demonstrated in Exhibits G.2. and G.3. While there can be value to either of these visual forms, neither one readily offers the insight of geographical movement around a ***facility***. Surprisingly, of all the major development sites visited, only one had incorporated

---

[69] Although the need for such insight is easy to see for the defense sector sites, it could also prove useful for many commercial organizations.

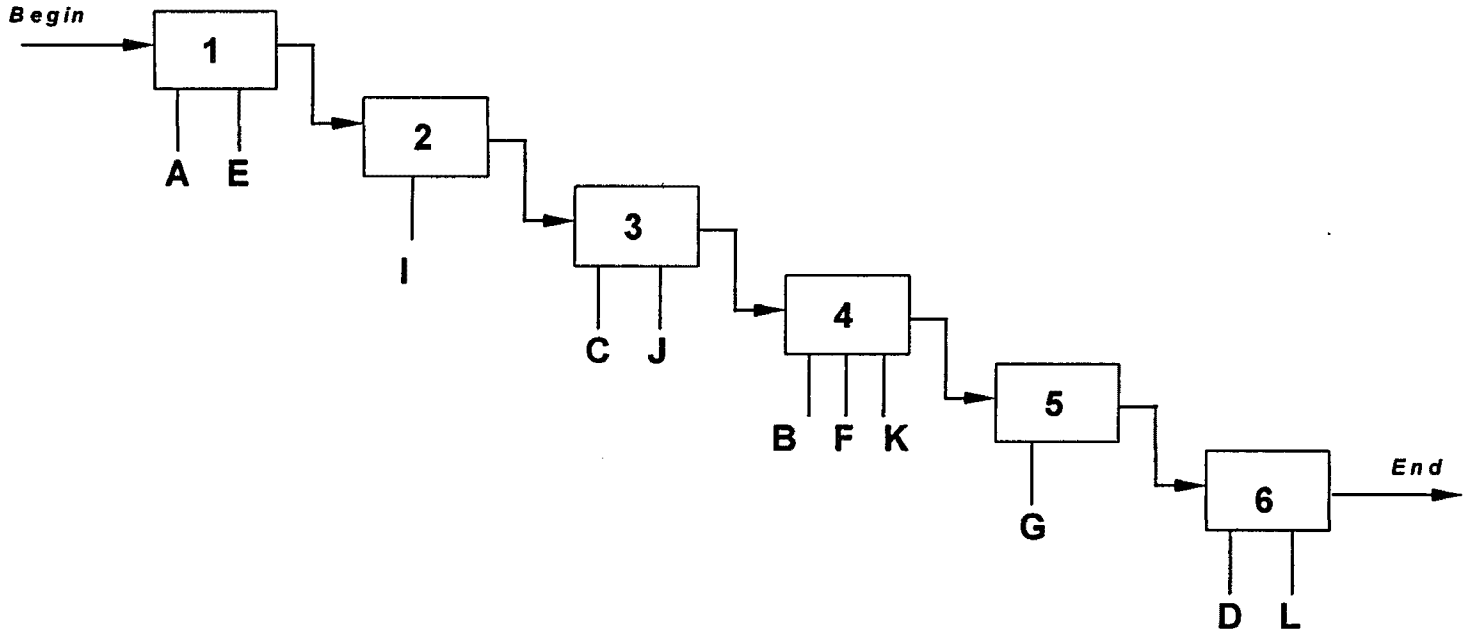
[70] An even easier-to-follow graphic, a ***functional-flow model***, is described in Appendix G.

the functional aspects of development into the physical construction and departmental layout of its engineering facility.

As the functional modeling efforts matured, there were still other areas of increasing concern which had not been previously addressed. The most striking realization, however, was that the process of development was not like a manufacturing process and could not be successfully modeled, or even visualized, like a manufacturing process. Thus, the plethora of manufacturing-oriented documentation tools which were available, though fundamentally well-intentioned, could not be directly applied in an effective manner. For instance, even the progression into the realm of product development was a fundamental shift in the use of IDEF as a non-linear documentation tool.

EXHIBIT G.2.

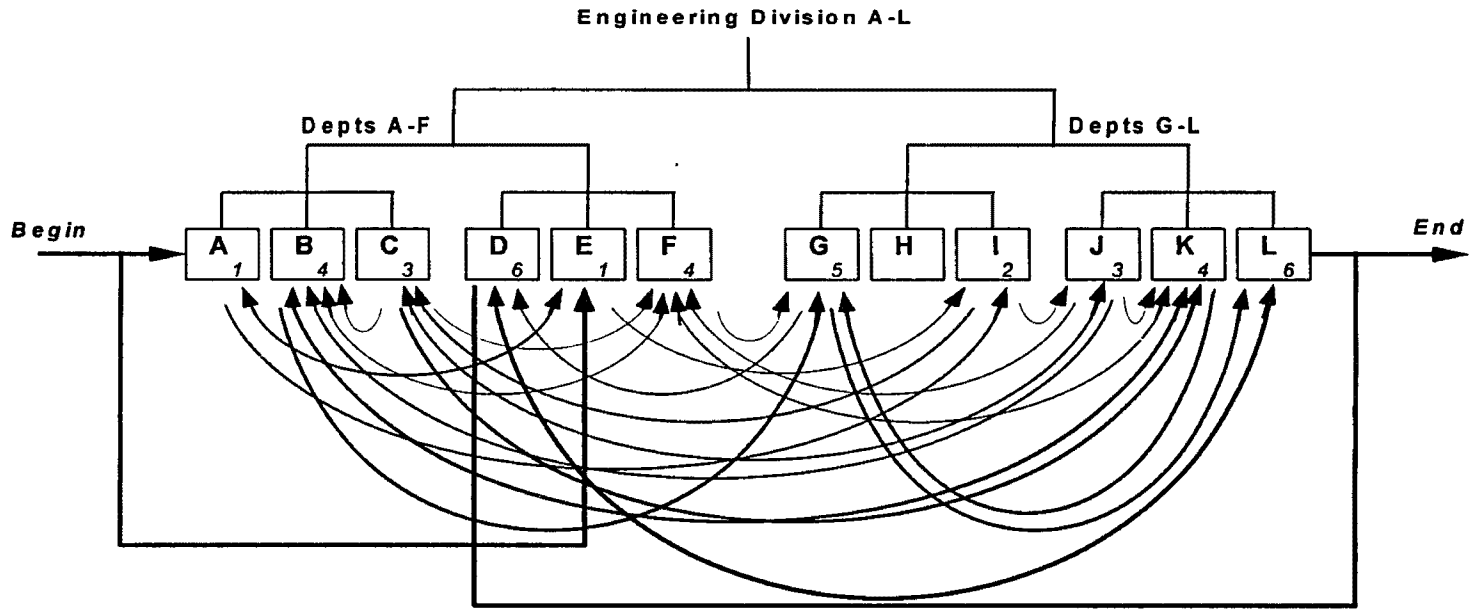
Linear Development Process (Functional Perspective)



This is a hypothetical, simple development process composed of six functions. Collectively, these six functions are conducted by 11 different departments. Each department's functional contribution is indicated by the letter code below each function box. For instance, departments A and E are involved in performing function 1, department I performs function 2 , etc. Note that, for simplicity, we have illustrated a ***linear*** process, with no feedback.

EXHIBIT G.3.

Linear Development Process (Organizational Perspective)



The ***same*** process illustrated in the previous diagram. This perspective shows how the process fits into the hierarchy of a development organization. Note how complicated the process appears in this perspective. When one considers this is the only viewpoint many managers often work with, it is easy to realize why development process documentation can be a non-trivial exercise. Further, recall this illustration is still of a linear process. The CPP dynamic analysis structure developed in this study permits us to examine non-linear variants of such a process.

Dynamic Model Development

As the concept of a dynamic, non-linear development process crystallized, another problem sprung up: no adequate tools exist to ***evaluate*** such a process. As this concept was verified at each new site, this non-linear analysis problem loomed larger and larger. In time, as an analyst, it became difficult to even consider looking at new sites because of the increasing frustration of knowing that adequate analysis tools did not exist and worse yet, were not even on the research horizon. It had almost become a question of ethics: Should one embark on an analysis for which one is confident that one's existing and foreseeable analysis tools are inadequate?

Subsequent to our field studies, our emphasis shifted towards developing a new methodology for analyzing such systems. The result of this effort was the ***CPP (Complex Process Path) Structure***. Before major strides could be made in developing a new analysis methodology, several backgrounding activities were necessary. These included tasks such as:

- Determining realistic system requirements for the methodology (so that it helped answer the right questions),
- Searching for useful elements of existing methodologies (to keep from "reinventing the wheel"),
- Determining acceptable upper and lower bounds for methodology complexity (so that it could be simultaneously insightful and not overbearing to understand),
- Selecting an appropriate platform for executing the methodology (specifically, to provide enough flexibility for more than one "custom" application).

Conveniently, many of these tasks were being conducted while documenting new product development processes at the field study sites. The process documentation teams, who were gradually realizing the differences between development and manufacturing processes, offered an excellent forum to refine requirements for the methodology. By observing, first-hand, the degree of use and familiarization of existing methods among managers, methodological overlap and interfacing characteristics of this new analysis technique could be considered. Such observations also offered a sense of how elaborate this technique could afford to be. In this regard, it was evident that the methodology must be simple to understand and quick to implement. ***Too little patience existed in the field to dwell on sophisticated analysis which could not be well communicated***[71].

Fundamentally, it was clear that existing managerial tools required a substantial (and unrealistic) level of confidence in predicting activity times and sequences. In fact, these are the very bases upon which various PERT and Gantt analyses depend. Even a stochastic model, in the form of a Markov process, had some unrealistic requirements, such as single (unique) state space--only one function could be conducted at a time. It was apparent that a new structure would, in practice, contain characteristics of both PERT networks and Markov processes. Essentially, a bi-directional PERT network was initially visualized. In short order, however, it was apparent that actual process paths were not so conveniently structured, but looked much more random or complex. Thus, the concept of the ***Complex Process Path (CPP)*** was born.

---

[71] It is largely because of this field impatience that this thesis was written in the manner that was. At this point in time, it is implausible to think that managers will engage in development analyses as wide-ranging as we conducted. Yet, it is hoped that this flavor of analysis will increase, however, and that many more interesting findings (even if more local in nature) can be derived from such form of analysis.

Because the analysis method to be developed would ultimately need to cover a wide range of models (of varying sizes) with various forms of non-linear process flows, it was immediately evident that the analysis tool would involve a high degree of computational ability. This naturally led to the need for computerization. (In retrospect, it is difficult to imagine that one could have ever contemplated otherwise.) Surprisingly, considering the novelty and importance of the direction we were cultivating, the selection of appropriate computational platform (i.e., software and hardware) was among the more difficult tasks. In many regards, it is still a valid issue which has yet to be fully resolved.

Some major platform issues which needed to be considered included computational capability/speed, visual representation, field availability, researcher availability, and interface capability.

- ***Computational capability/speed***: Because of the size of the functional models (at one organization, over 1200 distinct functions were identified) and their varied looping characteristics, it was recognized that a tremendous amount of computation would need to be accomplished. If continuous processing (mathematical integration) was to be done, at least this many variables would need to be considered. Integration of over a thousand partial differential equations smacked of the need for supercomputers, perhaps several supercomputers working simultaneously.

  If discrete (integer) analysis of the system was to be conducted, the computation requirement could be reduced somewhat. Since such analysis would involve the simulation of functional operations, with distinct time intervals, the number of

operations per unit time would be a function of the number of distinct time intervals (or resolution) which were deemed required. Since the majority of new product development projects had lifetimes between 6 months and 8 years, it was desired to have at least 10 "years" of available dynamic analysis range. Resolution was expected to be on the order of days or weeks (worst case). Thus, approximately 2500 discrete time intervals were expected to be covered in an analysis. To facilitate adequate what-if analysis in this study, a simulated run would be expected to be complete in less than a few hours. Better resolution could be obtained through the use of faster computational capability or lengthened run-time.

- ***Visual representation***: Although the analysis (computation) methodology had no requirement to be visual in nature, underlying elements (i.e., functions and entities) of the computed analysis were required to be representative of actual elements observed in nature, and comparable from model to model. In the majority of computer-based analysis methodologies today, some visual representation is provided, even if these are "re-created" images of previously computed results. Since many developers seemed to be at ease with the functionally based IDEF0 models being developed, it was prudent to provide a visual forum which was compatible with such models. Further, visual (object-oriented) model development could open the new methodology to "non-programming types" of managers. The underlying methodology software could translate such visual images to code. Thus, the tool would likely enjoy the most widespread use if it employed a high-level, easy-to-understand user interface.

- ***Field Availability***: It was observed that individuals within the majority of development organization had some degree of computer familiarization. A significant number of them had close access to a PC, even if not on their desks. Although most individuals (particularly managers) did not use sophisticated CAD/CAM equipment, personal computers were used to assist in various engineering functions--from engineering drawing development to mathematical engineering analysis, to costing to scheduling to memorandum correspondence to proposal preparations. Mainframe computer usage was rare, and usually limited to batch processing by computer professionals within the organization's MIS departments. Very little or no product decision-making was observed to be in response to mainframe computer use[72]. Even the recent increase in workstation availability has offered little more to the typical developer than a more efficient PC.

- ***Availability to me***: Due to the nature of the computing facilities at/through Northwestern University's Vogelback Computing Center, there was little or no practical limit to the computational power (hardware) available. With supplemental workstations available within the IE/MS department as well, previously untenable local processing was possible. Thus, aside from supercomputer availability, hardware did not present any foreseeable constraints. The more immediate concern was software availability.

  Although my programming background was not overly extensive (primarily limited to archaic BASIC, Pascal, FORTRAN-77, and COBOL languages), it

[72] Notwithstanding, a few engineering analysis applications, such as FEA or FMA, did use these higher capability computing devices.

would not have been too difficult to write a customized software for use on any available platform. The more relevant question was whether this was the prudent thing to do. Numerous simulation and mathematical analysis programs were readily available for use on a variety of platforms. In fact, many "candidate" programs had built-in capabilities beyond that immediately necessary for the CPP structural analysis, but which could prove useful for a variety of follow-on studies. In discussions with the software developers of most of these prospect programs, it was revealed that most contained enough flexibility to accommodate our non-linear processing needs.

Selection criteria was thus based upon adaptability of existing software, its cost, and available computer time. Considering that initial CPP model development was going to be a highly iterative task, the latter criteria was highly valued. A personal-computer based software could be portable (used anywhere with my laptop computer) and schedule convenient (could be used any time of day without conflicts of other computer users). Further, this permitted me the opportunity to focus on the overall characteristic dynamics of the new product development process in the field, rather than spend thousands of hours writing software in the laboratory.

- ***Interface w/other software***: Although, in retrospect, this issue was a little premature, there was an underlying requirement that the developed CPP Structure be compatible with existing documentation and analysis tools. As mentioned previously, the IDEF0 functional models were agreeable representations of actual operations, even if they were only static ("snapshot") models. Thus, an interface

"port" between the IDEF-based models and the CPP Structure could lend some visual credibility and acceptance among process participants who considered their system to be linear in nature. Further, it was envisioned that an output "port" of dynamic analysis results to other analysis tools would facilitate better understanding of results. As it has turned out, other analysis tools have been used, but often in ways never imagined prior to the development of the CPP Structure.

For purposes of this study, it was determined that a smaller, less complicated model would be a preferable means to communicate real-life findings. Such a small-scale model of, say, four departments could be readily understood (in principle) while still incorporating the types of complexities (non-linear behavior) of actual development organizations. Once such a pilot system was analyzed and understood, future work could concentrate on more complicated (and possibly more complex) real-world systems.

This approach permitted the use of a moderate PC platform (specifically, a 386SX computer was used) with windows-based software (CACI's SIMPROCESS™ software was utilized for this study).[73] Some SIMPROCESS programming[74] was required with this selection, though this constraint is becoming less prevalent as the CPP Structure becomes more refined. Specifically, much of the programming can now be accomplished through visual object selection and characteristic menus, rather than low-level code. Once this platform selection was made, formal scenario models were being tested within 3 weeks.

---

[73] Incidentally, IDEF0 models developed with and existing IDEF0 modeling software (Wizdom System's IDEFine-0™) already are exportable to SIMPROCESS.

[74] SIMPROCESS (and its near clone SIMFACTORY®) is written in the SIMSCRIPT® simulation programming language.

Analysis of Dynamic CPP Models

Once the basic structure of the CPP had been established, 52 parametric variations were employed. In the course of these runs, over a dozen alternate structures were also contemplated and tested (these alternate structures and results are not discussed in this report, however). Although there was no "formula" for how these variations were determined, a few guidelines were established and followed. These included selection of realistic parameters, holding consistent system structure, and using single variations of parameters:

- ***Realistic parameters:*** During construction of the CPP model used in this analysis, many variables were considered as potential parameters. We sought to examine the effect of changes in recognizable, realistic parameters. For instance, we considered changes in information processing rates (via INFO Efficiency), prototype processing rates (via MAIN Efficiency), information system bandwidth (INFO buffer sizes), and simultaneous prototype "bin" size (via Prototype buffer sizes). All of these had realistic counterparts (as well as many local advocates!) in actual development organizations. Since we were trying to gain insight into the overall systematic effects of these parameters, we looked at each of them in detail. In conjunction with such parameters, we also considered the variation of engineer allocations for each departmental function. This provided relevant and significant insight into the effects of various personnel allocation strategies. These insights can be used for development managers at a wide-range of development organizations.

- ***Structural consistency:*** Although several parameters were varied in significant ways (see the next guideline), the basic structure of the tested CPP model was held constant throughout the analysis. For instance, processing priorities, prototype paths (from dept. to dept.), and departmental structures were all held constant. This permitted us to focus on the parametric effects, rather than on other system structure differences which could change results in a significant manner. Thus, the ***specific*** analysis results obtained here are only true for the specific model tested. The ***nature*** of the results, however, can be expected to be equally insightful (though perhaps vastly different!) for any other developed model[75].

- ***Singular parametric variation:*** For each alternative test, only one parameter was changed at a time. This eliminated covariant effects, thus enabling direct, cause-effect, analysis of the parameter in question. For instance, when various buffer sizes were tested, all other parameters were held constant (e.g., INFO and MAIN Efficiency was held at 100%). Of course, multiple combinations of parametric changes were tested, as well. Development of such tests, however, were the result of sequential, single parameter changes. For instance, when the relationship between INFO processing and MAIN processing was being investigated, a "cross-product" of INFO Efficiency and MAIN Efficiency was developed, with all other parameters held constant. This resulted in 16 different tests, to accommodate the pair-wise combinations of the four efficiency levels (50%, 100%, 150%, and 200%) for each processing variable. Thus, one could assert that all dynamic

[75] It is well recognized that actual organizations engage in semi-continuous variations of even our "constants". Naturally, future analyses will be expected to consider such structural transformations, as well. We discuss this research requirement at more length in Chapter 8.

analysis runs were "controlled" experiments. Such capability was one of the major advantages of creating such a dynamic analysis model.

Over the course of twenty weeks, the desired parameter variations were developed and incorporated into the CPP Structure. Execution of the each of the various runs consisted of five basic steps:

1. Determine the parameter settings for a run;
2. Modify the CPP model to incorporate the new parameter settings;
3. Run the CPP model in SIMPROCESS;
4. Visually observe the CPP model in action;
5. Collect numeric data from the SIMPROCESS datalogs.

Once the CPP results data had been collected, the data was transferred to spreadsheet files for more convenient manipulation. Some of this data transfer was automated (using ASCII import protocols); some was manual (cell-by-cell keyboard data entry). Upon creation of these data tables (one or more spreadsheets were generated for each run variant), various statistical tests and graphical representations were generated. Specific software utilized for these analyses included the following:

- DATAGRAPH™ (statistical analysis) utility within the SIMPROCESS™/SIMFACTORY® software
- EXECUSTAT™ (statistical analysis)
- Fractint™ (fractal analysis/graphics)
- IDEFine Family™ (modeling documentation tools)
- Lotus 1-2-3® (spreadsheet/graphics)
- MathCAD™ (mathematical analysis/graphics)
- MATLAB™ (specifically, the SimuLink™ non-linear analysis tools)
- Quattro® Pro (spreadsheet/graphics)
- TimeLine® (managerial scheduling/analysis)

An important consideration throughout the analysis process was that non-linear process analysis of development had not been conducted before. Thus, though many traditional tools existed for conducting process analysis, new methods had to be created for examining these generated non-linear processes. In some regards, the analysis approach resembled that of an electrical engineering review of a complicated, simultaneously negative/positive-feedback system. One had to maintain, at once, a holistic and a focused account of how the system behaves in response to alternative parameter settings. Because of the seemingly schizophrenic behavior of the CPP structure under certain parameter regimes, it was exceedingly difficult to accept that simple, reductionist tools would suffice in understanding (and subsequently predicting) the system's performance.

By far the most insightful method for reviewing performance was physically observing the dynamic conduct of the model in real-time. This was accomplished by switching the simulation tool into visual mode (rather than the much faster hidden mode). By observing

when and where backlogs of prototype and information physically accumulated in the model, it was possible to see how and why each member (and each function) of each department was dependent on other members in other departments. No anticipated degree of formal mathematical analysis could convey this sort of insight, a priori.

Appendix H: Development of a New Product Development Framework

In this appendix, we shift our focus from observation-based descriptions and multi-sourced interpretations of the process and move towards a more complete explanation of the process of development. Thus, we are less concerned with the ***what's*** or ***how's*** of the process, and more concerned with the ***why's***. Admittedly, this thesis cannot consider every facet of a development manager's many concerns. Rather, the structure described in the following pages merely illustrates some systematic concepts which seem to be underlying ***drivers*** or, at minimum, catalysts for many of those concerns.

Our objective here is to introduce some analysis considerations with regard to the "process" of product development. First, a functional framework is developed and explained. Using simple Markov chains, some observed categorizations of structures which fit within this framework are discussed. Then, a new dynamic analysis structure is presented, which incorporates the concepts described throughout this work. We begin with a humorous, though integrative and revealing, analogy.

The Interactively Dependent process of innovation

In chapter IV, we illustrate some of the perceptions of product development from the participants. Clearly, the process of development is far less straightforward than any of us trying to analyze it would like to see. It has even proven to be difficult to get a dated "snapshot" of how the process of development looks, much less an up-to-date real-time analysis. For much of the previous discussion, our efforts have been focused on "routine" development of new products. For established organizations with established products,

designing the "next generation" of the product supposedly follows a basic routine. There exists some corporate memory of how things were done the previous ***n***-times: what mistakes were made; what techniques were successful; what impaired the process; what "came close" to (or succeeded in!) arresting the process; how teams should be structured; how they shouldn't be; who to work with; who to avoid; what to test; how to test, etc.,... the list is long. Using such "lessons learned," managers and developers are expected to avoid previous traps and look for methods and tools which will accelerate the process in time, product quality, and cost efficiency.

In many regards, asking an organization to perform routine product development should be much like asking an experienced over-the-road trucker to carry your critical merchandise from New York City to Los Angeles. With some experience with cross-country travel (perhaps even this trip!), an up-to-date atlas, a functioning CB radio, decent weather reports, and enough resources to fuel the truck (and maybe to eat), you can say "sayanara"... he'll be in LA in six days. With the right tools, a clear objective, and a little experience, the task is accomplished with little or no complication. Those complications which do occur might be the result of new construction, accidents, unforeseen weather, or mechanical difficulty, for example. For the most part, however, the driver is comfortably in control of his destiny (and your merchandise). By locating his origin and destination, he will select a decent, if not optimal, route which acceptably meets both his and your standards for timeliness, cost, and safety.

This is a best case scenario, however. Let us get closer to what a developer typically experiences during "routine" product development. Suppose, that Interstate 80 (the main East-West thoroughfare which our trucker knows and loves so well) was shut-down in a

variety of unspecified areas (which he doesn't know ahead of time) and that there existed only a few "questionable" maps of the metropolitan areas of New York, Chicago, St. Louis, and Los Angeles... and *no* rural maps. How complication free would his trip be now? Given that he was accustomed to driving the interstate for the past 20 years, and rarely wavered from the highway, how well could he predict his arrival time? What would his total cost of travel be now? How confident would you be about the condition and location of your merchandise, after the plethora of detours, stops, and starts to verify location? And yes, (did I fail to mention this?) after the first three days of traveling, you (the "customer") had asked the dispatcher to radio-in a destination change (to Seattle instead of LA), but are not sure if the message ever got through.

This is all in a days work for a "routine" developer, who takes on the trucker's role in this analogy. He merely follows the directives he is given, using the best judgment he can, given his situation. If you, the customer, are unhappy, the developer might not ever even know[76]. If the driver gets enough of these unfavorable, no-win assignments, however, then *he* starts to look bad, despite his best efforts.

This brings us to the assignment of conducting ***new*** product development. Using the same basic analogy, we have a situation more like the following: Our trucker is asked to take our pallet of merchandise and deliver it to "Fred or Harry on the west coast." Given his

---

[76] In fact, problems with inadequate customer feedback reaching the developer was widely cited as a peeve of many engineers. At one site, the marketing department had such a stronghold on customer interaction (the engineering department was, unbelievably, forbidden from talking to dealers or customers) that a PDT for one automotive product line tried a semi-bootleg approach: they engaged in track test focus groups with non-engineers (secretaries, accountants, purchasing agents, etc.) *from within the engineering department.* "Luckily," as one engineer described this "experiment", these focus group studies were successful enough that "management took notice and blessed" this activity for other vehicle platforms to emulate. Naturally, this was not documented as *verification* of marketing-defined requirements, but rather as a form of *prototype validation.*

preoccupation for turning nothing into something (sound like an engineer?), the driver takes on this assignment, despite its high level of ambiguity (no address or city). Once again, the navigational amenities for interstate travel are unavailable. He has no radio and has never driven west of the Mississippi River. In fact, he doesn't even know which coastline is being referred to, but feels confident that "west coast" means the same as "Pacific coast." Thus, starting from New York, he begins on a westward route, perhaps assuming that the sun still sets in the west and that his compass is correct, for these are among his few references. After a few days of travel, and fruitlessly trying to extract more detail from his dispatcher via on-the-road phone calls, he discovers that he will have to find out who and where "Fred and/or Harry" are by himself. Using a "development route" similar to the routine route described above, he realizes that much more investigative work is needed. After two more weeks of investigative discursions, foul weather, and more than 1000 miles of coastline to search, more investigation work is needed. With some more phone work and discussions with locals, he has determined that there are three Freds in San Diego and five Harrys in San Francisco. Upon arrival in LA, he is still contemplating whether to go North to San Francisco or South to San Diego, he stops at a service station to ask for directions to the nearest restaurant. To his utter surprise, two gentlemen, whose names are ***Ted*** and ***Larry***, and happen to be from Seattle, greet him at his truck and say, "We've been looking for this critical shipment from NY for almost three weeks--where have you been?"

Serendipity, misinformation, unclear objectives, changing objectives, misunderstood (and seemingly unhelpful) management: these characteristics complement the more formalized aspects of the innovative product development process which were described in the

previous chapter. Together, they illustrate the observation that there exists no comprehensive "road-map" for developing and introducing innovative products to the marketplace. Further, it should be realized that any ***maps which do exist can be suddenly deemed inappropriate once objectives or techniques[77] change***.

The process which has been studied in this research is continuously in flux; its changes come from both within and outside the jurisdiction of individual process participants. The process may turn back on/repeat itself or head into entirely new directions, in which past experience may offer little or negative assistance. Some degree of "routineness" may enable evolution of a familiar organizing structure, which seems to be necessary for large groups of individuals to work *with* each other. Yet, embryonic ideas seem to flourish best in ***unstructured*** environments, which may help explain why so many individual ideas get no further than the individual or small groups that advance them.

Both routine and new product development projects are currently managed as if they were well-established, repeatable, and predictable processes. With such assumptions, stoic performance measures become widely used. For progressive developers, this can be very frustrating. Turning back to our driver's situation, it is like having a dispatcher, who is unaware of the driver's confusing situation, evaluate the driver's performance. His own personal efficiency measure--how well drivers chain stoplights together and minimize idle time--may have no bearing on the effectiveness of the driver's timely, safe delivery.

---

[77] Suppose our driver was a pilot, instead--the road maps available would not help much.

The next two sections of this chapter help prepare us for a new era of generating and scrutinizing ***effectiveness*** in product development, not merely creating and diffusing collections of independent ***efficiencies***.

The functional structure of development

This study has reviewed the product development processes of a number of companies, using the IDEF0 functional modeling methodology. Illustrative as it is, functional modeling in itself is but one tool for analysis[78]. One type of analysis which can be performed, but which has not been fully exploited in research and industry, is dynamic analysis of specific functional flows from an existing functional architecture. By understanding how a functional model operates under the progression of time, we can better understand the dynamics of the engineering organization in its quest for reduced development times, increased quality, and reduced development costs. In this section, we distinguish the concepts of functional-flow from functional architecture and introduce a simple, high-level architecture to be used in future sections. The simple concept of sequencing is briefly introduced at the end of this section.

Functional Architecture vs. Functional Flow Models

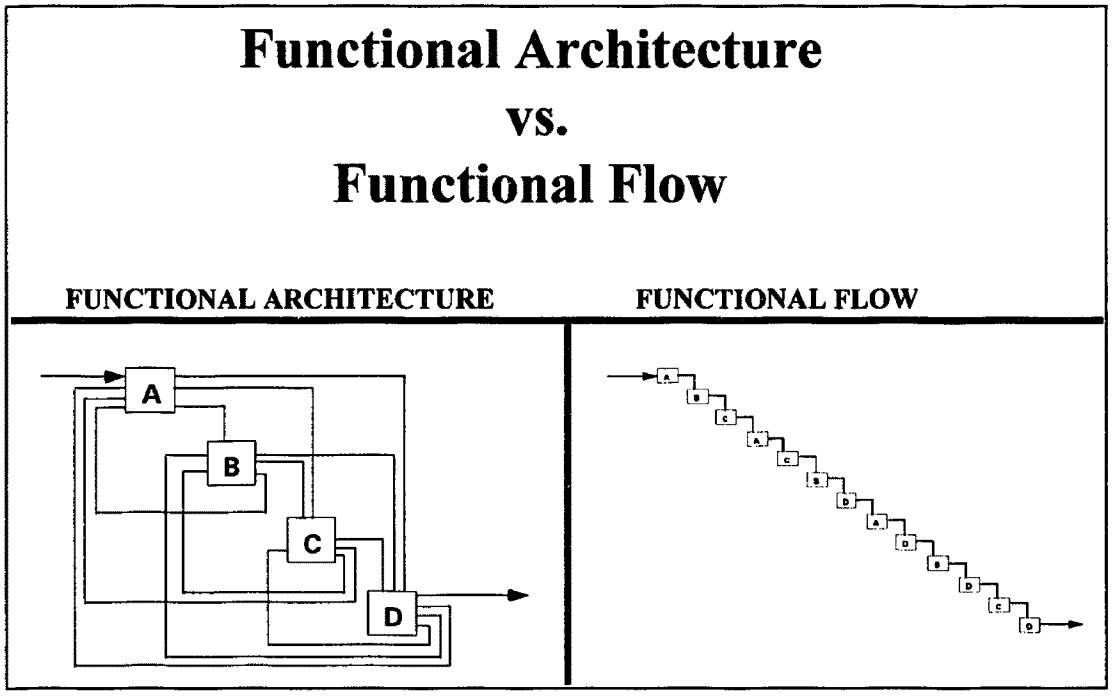
Activities, functions, nodes, cells, responsibilities, processes, operational flows,... many names have been given to symbols which describe specific operations of manufacturing, engineering, and other organizations. Though these terms do not appear to differ significantly in meaning, their varied use implies differences in the models in which they reside. For our discussion, IDEF0 functional architecture models are compared to functional-flow models, for reasons which will quickly become apparent. Before specific innovative product development scenarios are presented, it is useful to distinguish between ***functional architecture*** models (in particular, an IDEF0 model) and ***functional-flow*** models.

---

[78] Strictly speaking, we cannot consider functional modeling to be an analysis tool. Rather, established functional models offer a ***basis*** upon which analyses can be performed, using a variety of analytical tools.

For reference, a sample functional architecture model and a functional-flow model are demonstrated in Exhibit H.1.

EXHIBIT H.1.

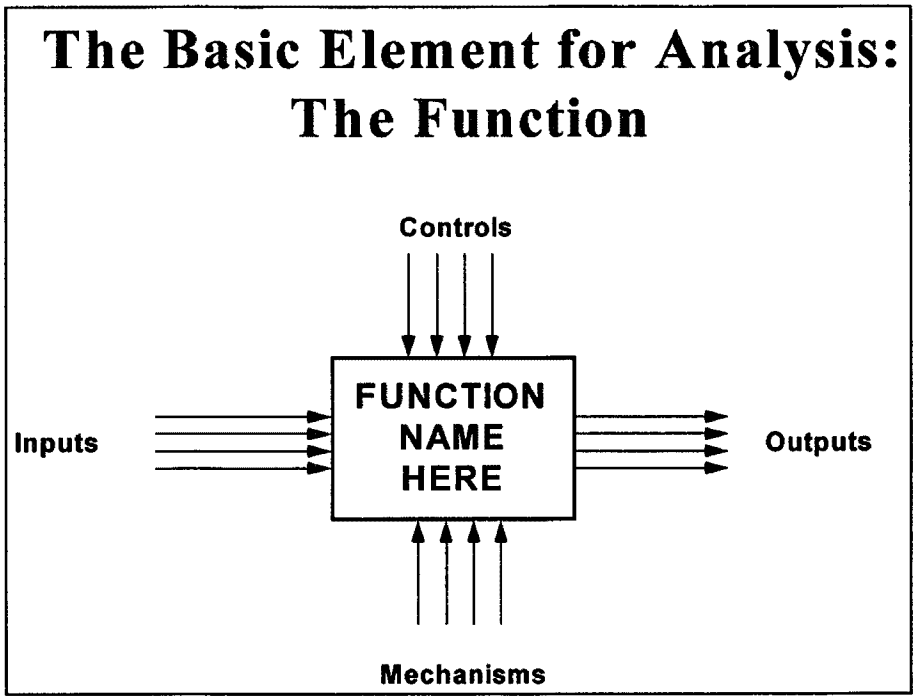


Functional Architecture

A functional architecture is an arrangement of unique operations, or functions, which are performed within the system under study. Such functions may be performed by many different people, using different tools, in different parts of an organization, at different times. Yet, in an IDEF0 model, a particular "function" only appears once. When functions are conducted concurrently or in different ways across the organization, their different modes are indicated in an accompanying textual description and demonstrated with graphical interfaces between that function and all relevant *upstream* and *downstream*

functions. Specific, unique functions are represented by boxes or rectangles, with their unique identifying name within the confines of their box. Refer to Exhibit H.2.

EXHIBIT H.2.



In IDEF0, a highly developed and structured form of functional architecture modeling, functional *interfaces* include inputs, outputs, controls, and mechanisms. Typically, a function utilizes multiple *inputs* to create one or more *outputs*. Inputs are always shown as arrows which enter the left face of a function box. Outputs are always shown as arrows leaving a function box from the right side. Outputs are the only arrows which are shown to leave a function box.

Facilitators for a function are called *mechanisms*. Typical mechanisms are physical tools, computers, people, or machines. These are shown as arrows which enter the bottom of the function box.

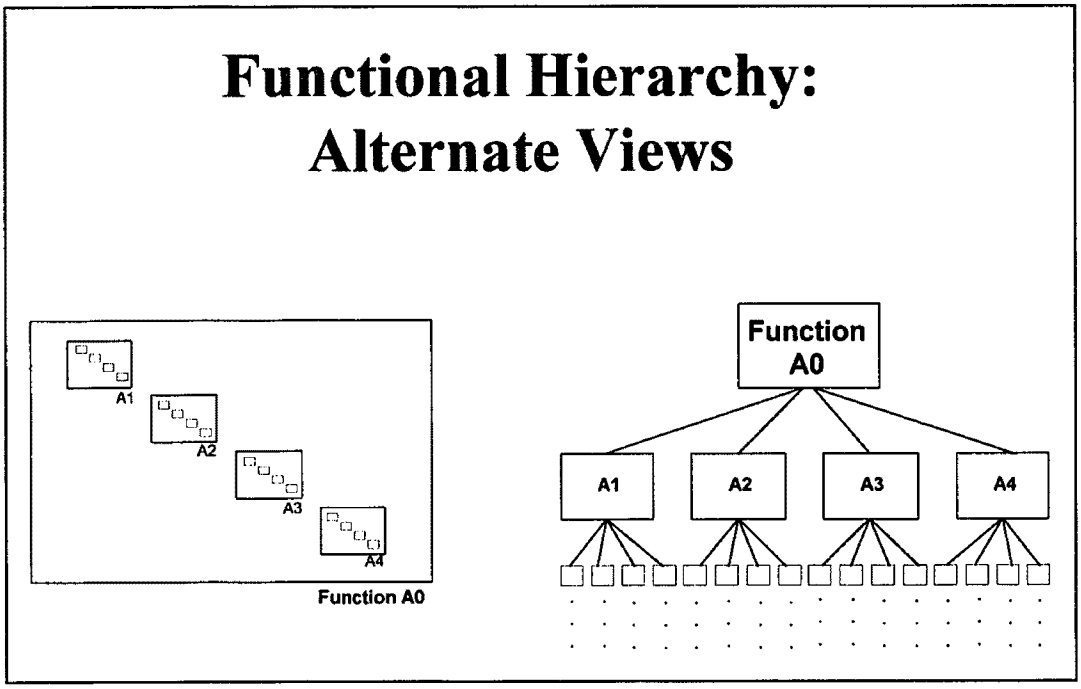
Additionally, an individual function may be governed by a *control*. Typical controls include policies, laws, cultural canons, instructions or requests, schedules, budgets, or information about the status of other functions. Controls are commonly thought of as constraints on functions, but this is not necessarily so. Controls are more accurately described as inputs which are not physically transformed during the operation of a particular function. They may indicate an upper or lower limit of a function's performance, but may also provide valuable information which is critical to performing a function; such information is generally not directly affected by the specific function under consideration.

One strict rule for constructing readable IDEF0 models is that no diagram may have less than three or more than six functions. If an organizational system under study has only 3-6 functions, this is not a problem. However, in every organization studied to date, including several family-run organizations, 6 functions have been found to be inadequate.

To resolve this problem, IDEF0 permits *decomposition* of functions. This entails developing 3-6 distinct "high-level" functions which collectively cover the scope of the system under study. For each of these 3-6 high-level functions, separate diagrams are developed. These "child" diagrams then follow the same rules that apply to their "parent". Thus, an IDEF0 functional architecture model is organized by a hierarchy of functions. In this hierarchy, every function is either a bottom-level function or is further described by

3-6 sub-functions. A functional architecture is usually demonstrated as a series of functional diagrams; each successive diagram gradually shows more detail than its parent diagram. Exhibit H.3. demonstrates two equivalent views of a hierarchy of functions, without the functional interfaces.

EXHIBIT H.3.



Functional architecture models are extremely useful because they provide an organizing framework for *all* functions of a system being previewed. Interfacing between functions is demonstrated by arrows from one function (the source) to another function (the destination). In an IDEF0 model, any function may be a source, destination, or both. Just as functions may be decomposed, so may arrows. Thus, the IDEF0 hierarchy can provide overall or specific views of both functions and functional interfaces, depending upon

one's analysis needs. At the same time, any given diagram has a reasonable number of elements, with no more or no less detail than one can cognitively comprehend.

The ordering of functions on a particular IDEF0 diagram is loosely based upon either chronological precedence or priority of function. However, it is widely recognized that, in practice, functions may both precede and follow one or more other functions, as part of a review process, for example. This iteration is provided for via feedback arrows. Such feedback arrows leave a function as an output and may enter any other function as an input, mechanism, or control. Through the use of such interfacing, we can preserve the singularity of each function within the functional model, regardless of its recurrence rate. Further, we can identify how any particular sub-function relates to *any* other sub-function in the architecture, regardless of either function's "location" in the hierarchy.

As mentioned earlier in this section, a functional architecture is an arrangement of operations, or functions, regardless of when, by whom, or how they get done. The mechanisms demonstrated on a functional architecture model delineate such diversity of operation. Thus, a particular function is only included once in the architecture, regardless of who performs it, when it gets performed, where it gets performed, or how it gets performed[79] .

Functional-Flow

---

[79] This characteristic of the IDEF0 functional modeling methodology is one reason that it can be so revealing in documenting redundancies in organizations under study. It is also a characteristic which is highly confusing to innumerable novice IDEF0 modelers. It is extremely common for such modelers to quickly assume that an organization with functional divisions will be straightforward to model. In fact, such apparent convenience is often a source of confusion and resultant delay when modeling a system for study.

Though a functional architecture model provides a relatively complete assessment of the functions which are performed, it does not explicitly define the order of operation or simultaneity of functions. Such capability is reserved for a functional-flow model.

Functional-flow models can be considered sequential orderings of functions, or activities. Due to their chronological ordering, such models reveal the sequence of operation from beginning to end. Thus, rather than using feedback loops, as seen in a functional architecture model, functional-flow models usually repeat the function or series of functions, as they occur in time[80] .

This directly implies that a standard functional hierarchy is almost always unattainable for functional-flow models. When hierarchical functional-flow models are attempted, they are often grouped by chronology or organizational entity (for those cases when major departments are responsible for major chronological stages, for example). Many unique variations of functional-flow models exist, each offering as many twists as the developer wishes to dream up.

It should be clearly apparent that functional-flow models exhibit linear, sequential orderings of activities[81]. In this situation, iterative feedback is treated as an exceptional

---

[80] For simplicity sake, however, some representation of functional-flow models *do* show feedback loops. This is done when a series of functions recur an unspecified number of times, as in a QC cycle. This is also often a result of insufficient knowledge about the iterative characteristic of functions, such as the conditions which induce feedback.

[81] Naturally, such a "linear" description does allow for simultaneous (parallel) functional-flows. This is, after all, an elemental basis for many PERT charts, which are used by managers and developers throughout the development process. The important consideration here, however, is not just such concurrency, but rather the fact that function-flows represent unidirectional, step-by-step procession, as the process moves through time.

case; in many functional-flow models, such feedback is illustrated for only the most frequent interfaces. In contrast, the functional-architecture model demonstrates as many relevant feedback loops as reality poses. This is one of the characteristics which make IDEF0 models more difficult to read, but more completely representative of the system under analysis. On the other hand, functional-flow models are more intuitive, prompting such analysis techniques as timeline analysis and unidirectional network analysis, such as PERT/CPM.

In contrast to functional-architecture models, functional-flow models do not generally possess a high degree of ***functional closure***. This means that functional-flow models are more difficult to bound, for use in simulation, for example. As we shall see in the upcoming pages, functional-flow models may, in many real cases, string out over an undetermined, conceivably infinite number of steps. This unwieldy sequence can be harnessed in a functional architecture model in as few as three functions. In first-hand experience, functional architecture models have ranged from 20 to 1200 distinct bottom-level functions, depending on the scope of system under study. Nevertheless, we can effectively bound functional architecture models for a manageable analysis.

A High-level Functional Architecture

For the following discussion, let us consider a condensed case of product development. We shall review the highest level functions of a typical functional-architecture model of product development. Thus, we begin with an intentionally simplified model. One could consider this is the "highest-level" diagram of a more comprehensive model. We shall return to the concept of comprehensiveness later in this discussion. In the meantime, let

us consider a case in which new product development is composed of just four basic functions:

(1) Define & Review Requirements
(2) Design Parts
(3) Build Prototype Parts
(4) Review Designs/Prototypes

Initially, it is assumed that each function has its own internal review mechanisms, which ensure that the output of each function being performed matches the specific requirements delineated for *that* function. This has the effect of eliminating the likelihood that a function will need to be done again, immediately after it has been completed. Thus, cycles will be demonstrated from function to function, not recursive on a given function.

The Concept of Sequencing

In how many ways can we perform these four functions? Let us begin with a few simple cases, establish a baseline for analysis, and then introduce some major variations which better reflect observed operating conditions. We shall see that the analysis of sequencing of functions is not as simple as one might hope, and that different sequences can induce significant differences in how well product development proceeds. Moreover, we are developing a unified framework which exhibits how such sequences change over time.

Classification of the Process

Here, we introduce two categories of process structures: ***Pre-defined*** structures and ***Experiential*** structures. Upon developing three simple, pre-defined structures, we shall introduce Markov descriptions of these processes, a stochastic ordering methodology, and comment on the long-run behavior of such simple linear systems.

Subsequently, time-based and frequency-based variations to these simple models are discussed, as they impact the development of an analysis model. Recognizing certain limitations to existing methods, a new analysis structure is introduced, which incorporates the non-linearity of the observed process into a previously linear network analysis structure. Finally, transient conditions are discussed as they relate to the development process.

Pre-defined Structures

In this section, we present three basic development structures, as well as an integrative compilation of these structures, using the functional-architecture methodology just described. These are classified as follows:

- Single-Pass Development
- Multi-Pass Development
- Internal-Iteration Development
- General Case Development

We have dubbed these structures Pre-Defined structures, due to their inherent structural stability: they do not change structure over time. Thus, for our modeling-visualization purposes, they behave according to a simple, non-changing set of production functions.

This proves to be very convenient from a mathematical modeling point-of-view. It is expected that losses in specific relevance may be gained in better understanding of general, system-wide tendencies. In section C.2., we introduce certain observed deviations from such stoic structures.

Single-Pass Development

We could be very ambitious and propose that the four functions in our high-level architecture are performed correctly in one sequential pass:

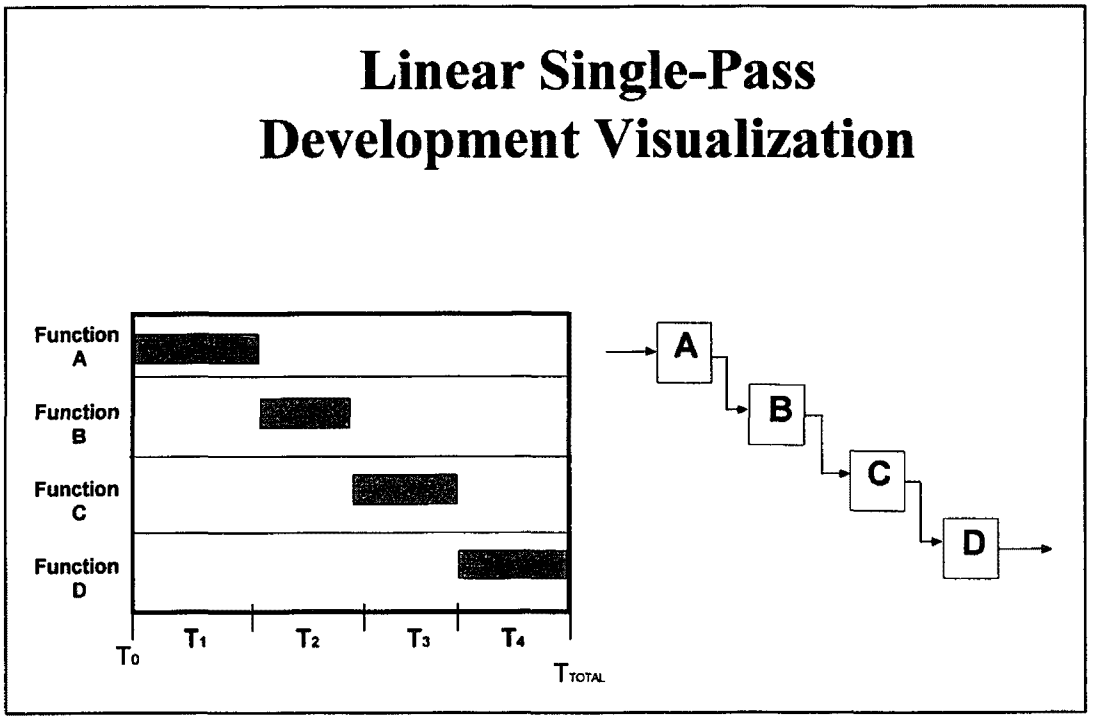
(1) --> (2) --> (3) --> (4).

Such a linear, single-pass scenario is the simplest process to conceptually understand. It assumes simple dependencies with no overlap, or concurrency[82]. Likewise, no delays between functions are apparent; when function (1) is complete, function (2) begins. Thus, it is clear that there exists no backlog or inventory of jobs ahead of each function. Implicit in such a case, as with any interesting queuing system, is that service times of each function are faster (on average) than the arrival rate. Such a model, and an accompanying traditional Gantt chart, is demonstrated in Exhibit H.4.

---

[82] All of these structures, in fact, ignore true concurrency, for the Markov process assumes that only one state (activity) may be in service at a given time. We shall relax this constraint in the next section, *Experiential Structures.*

EXHIBIT H.4.



If we know the process times for each function, say T_1, T_2, T_3, and T_4, then the total time for engineering the product is, straightforwardly enough, $T_1+T_2+T_3+T_4$. It is significant to note that many company executives are of the belief that their developments operate in this fashion, at least at the highest functional levels.

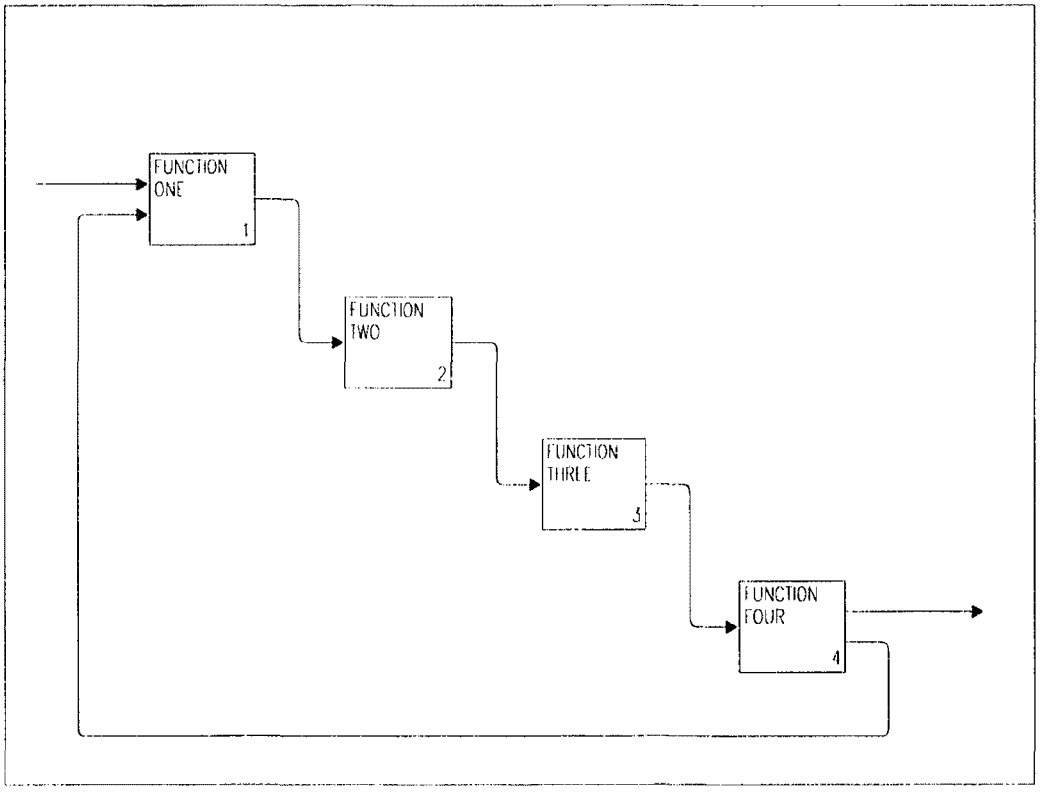
Multi-Pass Development

Now suppose that it is impossible to correctly conduct these four functions in such an orderly, sequential process, on a single pass. Let us suppose, for example, that the Define & Review Requirements function (1) is conducted after each particular prototype part has been reviewed, so as to redefine the requirements for other prototypes, interfaced parts, or necessary tools. This scenario results in a process that looks like this:

(1) --> (2) --> (3) --> (4) --> (1) --> (2) --> (3) --> (4) --> (1) --> (2) --> (3) --> (4)...until the operators of function (4) are satisfied with the total result.

For simplicity, regard this as a 123412341234... or ***1-2-3-4*** process. In IDEF0 modeling format, it looks like Exhibit H.5.

EXHIBIT H.5.



This is also recognized as the ***design-build-test*** cycle. In many organizations, it is expected that this cycle will iterate several times during a single development. Each iteration, however, is expected (hoped) to be completed faster than the previous one, as familiarity with the objective(s) and interfacing capability with other functions--integration-- improves.

Assuming that (1), (2), (3) and (4) were the only functions necessary for development and that few iterations of such a sequence was necessary, this multi-pass scenario would be considered a "good" development algorithm. "Small", independent inventors and innovators repeatedly use such an algorithm. It works well for them.

For development of an interdisciplinary, complex product (with numerous interconnected sub-systems), however, this algorithm does not work well. A suggested reason for this is the large number of functions which need to be performed by large numbers of people. The basic four functions used here are not performed by one or even a handful of individuals. It is common to see several hundred people involved in the development of a complex product. Thus, these four functions are decomposed into sub-functions, and sub-sub-functions, and deeper $(sub)^{X}$-functions: the basis upon which some departments are formed within companies. Of course, interdisciplinary product developments may extend well beyond the bounds of a single company, as specialist outsourcing is regularly conducted.

When a multitude of sub-functions are considered, we see that the ***1-2-3-4*** process development is fatally flawed. Why? The key reason for this is the vast time lag for feedback between functions. If a problem is identified in a sub-function of (4), for instance, and the source of the problem was a disconnect of requirements (1), then many of the sub-functions within (2), (3) & (4) were performed needlessly. This is analogous to manufacturing lines creating work-in-process (WIP) inventory of unusable parts: no matter how *efficient* the build process is, any resources expended (time, labor, machine, raw materials, overhead, etc...) are unrecoverable.

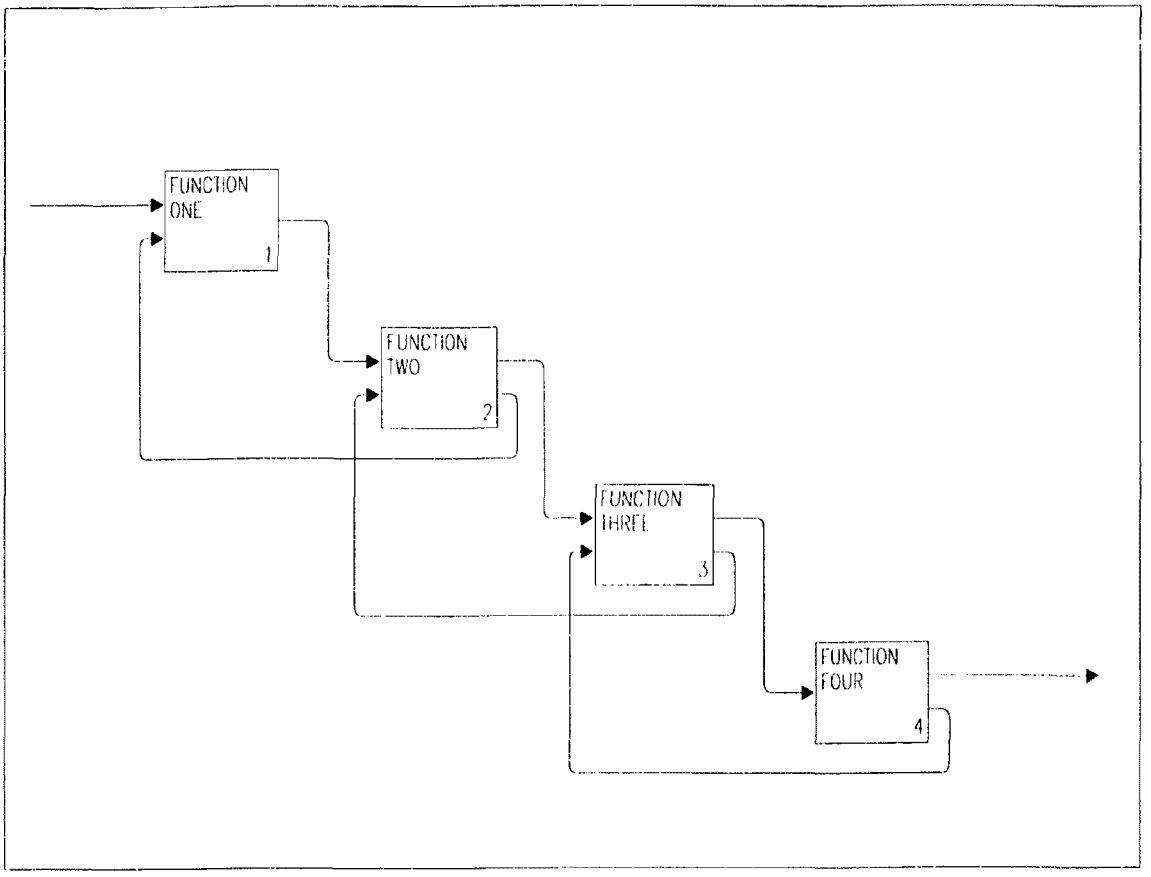
In many sites visited, this algorithm is still the status quo. This is particularly the case in smaller companies, which tend not have computerized CAD/CAE/CASE design facilities. Many large companies did have such facilities, but did not use them to assist their communications, however: they were just used as a replacement for the drafting table. There is a concerted effort within major companies in several industries reviewed in this study to get away from this development algorithm, though it is still widely apparent.

Internal Iteration Development

Let us take another step forward in the development process structure and introduce a third development scenario: Internal-Iteration Development. In this scenario, it is possible to introduce interfaces from any function to any other function. Thus, when a problem is encountered with the developing design, "previous" functions can be notified, to prevent further flawed development to take place.

It is important to realize that there are two frequent variants of this latter development scenario. The first is the most prevailing variant at most organizations visited. For lack of a better term, I have dubbed it the *one-step restriction*. This means that Internal-Iteration is permitted, but only one step backwards in the development process. Thus, problems discovered in function (3), for example, are only relayed to individuals who perform function (2). Only if engineers performing function (2) encounter further difficulty in resolving the problem does the process return to function (1). Such a scenario is illustrated in the IDEF0 diagram in Exhibit H.6.

EXHIBIT H.6.



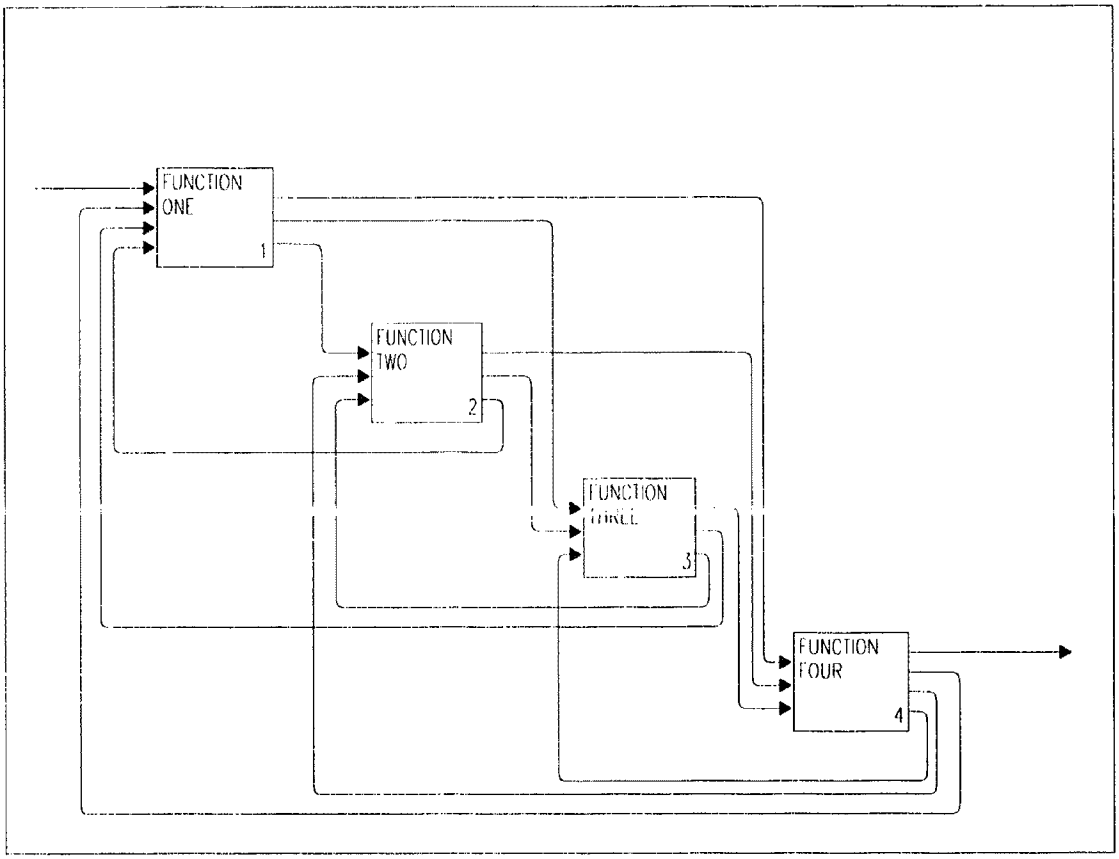
Another variant of the Internal-Iteration Development algorithm is what may be dubbed the *forward-feed option*. This means that information about encountered problems are sent ahead of the current function, so as to provide better timing information for downstream activities. Such send-ahead may include instructions to proceed with development, while providing for potential changes which will result from fixing problems.

It should be clear, however, that any of the Internal Iteration development processes are dependent on the ability for participants to locate and communicate with appropriate other participants, in a suitable time frame and with sufficient clarity. Naturally, inability of participants to do this effectively directly hampers the development process.

The General Case

When enterprise integration efforts are attempted, the intention is to "link" all development functions with all other functions. With channels available between any two functions in the organization, regardless of physical location or department hierarchy, lags in iteration (wasted efforts) are expected to be diminished. Exhibit H.7. illustrates the concept of such ***n-to-n*** communication among functions.

EXHIBIT H.7.



We shall consider this to be the most general, most comprehensive description of a functional model with four functions. Every function has an available communication channel to every other function. What's more, these channels are bilateral and universal in scope of potential carrying capacity (recall that a single arrow may represent several detailed categories of interfaces). Of course, no observed development organizations with more than a few employees operates in this "complete" mode[83]. Yet, we have potential to consider an organization which does. This permits us flexibility in modeling any development organization which has fewer communication characteristics than this model.

Markov Descriptions of Pre-defined Structures

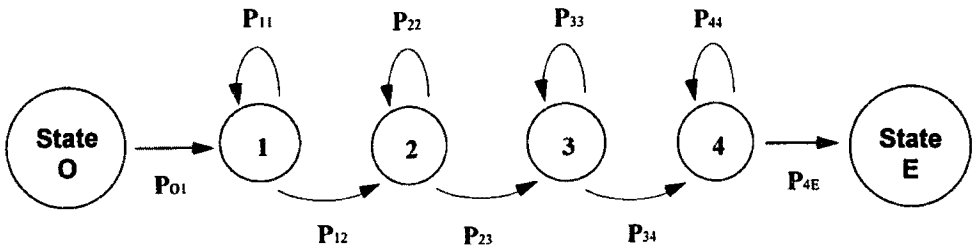
Each of the above developmental structures are either deterministic or deterministically stochastic, depending on the nature of parameters which describe their interactions. In this vein, it is conceivable that the process of product development could be characterized by relatively simple Markov chains. In such a description, each function may be considered a "state" in Markov parlance. Relations between such states may be characterized by the weighting schemes (probabilities) in the Markov description. Thus, states which are expected to take longer to process can be weighted with higher

[83]Very small firms, composed of the proprieter and perhaps one assistant, may be considered to behave very close to this complete integration mode. To some extent, integrative channels are conceived as the neurons within the person's brain. Even in this case, simultaneous integration among functions may be limited to the cognitive capacity (including long-term and short-term memory, neural transmission rates, and a host of other ill-understood cognitive processing dynamics) of the individual.

recurrence probabilities. Movement from state to state is accomplished by assigning a non-zero probability to the appropriate transition cells in the Markov matrix.

For the three basic structures described above, the Markov processes have the following forms:

For the S*ingle-pass development structure:*



where:

P_{01} is the probability that the process is initiated from the outside environment, state 0. As is apparent from the remaining variable definitions, this represents the expected time until development of the product begins.

P_{11} is the probability that the process remains in state 1 from time *t* to time *t+1*. Similar definitions apply for P_{22}, P_{33} and P_{44},

P_{12} is the probability that the process proceeds from state 1 to state 2. Similar definitions apply for P_{23} and P_{34}.

P_{4E} is the probability that the development process is complete and is ready for release to the outside environment, (E). In this case, the environment may be the production facility for the item being developed.

In matrix form, these are translated as follows:

State t+1

			0	1	2	3	4	E	
		/							\
S	0	|	P_{00}	P_{01}	0	0	0	0	|
t	1	|	0	P_{11}	P_{12}	0	0	0	|
a	2	|	0	0	P_{22}	P_{23}	0	0	|
t	3	|	0	0	0	P_{33}	P_{34}	0	|
e	4	|	0	0	0	0	P_{44}	P_{4E}	|
	E	|	P_{E0}	0	0	0	0	P_{EE}	|
t		\							/

Assuming that state 0 and State E are one and the same (E), we may simplify this to a 5x5 matrix:

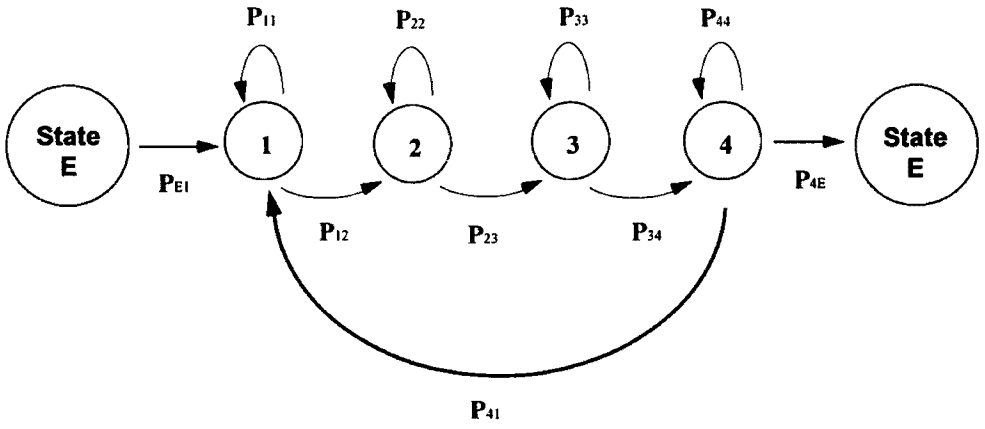
State t+1

			E	1	2	3	4	
		/						\
S	E	|	P_{EE}	P_{E1}	0	0	0	|
t	1	|	0	P_{11}	P_{12}	0	0	|
a	2	|	0	0	P_{22}	P_{23}	0	|
t	3	|	0	0	0	P_{33}	P_{34}	|
e	4	|	P_{4E}	0	0	0	P_{44}	|
		\						/
t								

where:

P_{E1} is the probability that the process will start at the next time increment. Currently, P_{EE} (= 1-P_{E1}) is the probability that it will *not* begin in the next time period.

For the M*ulti-pass development* structure, we have:



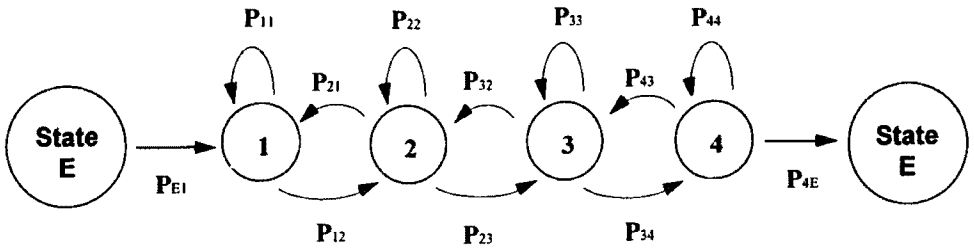
in matrix form, this reduces to:

State t+1

			E	1	2	3	4	
		/						\
S	E	|	P_{EE}	P_{E1}	0	0	0	|
t	1	|	0	P_{11}	P_{12}	0	0	|
a	2	|	0	0	P_{22}	P_{23}	0	|
t	3	|	0	0	0	P_{33}	P_{34}	|
e	4	|	P_{4E}	P_{41}	0	0	P_{44}	|
		\						/

t

For the *Internal-Iteration Development* structure, with the one-step restriction, the Markov process has more communication channels:



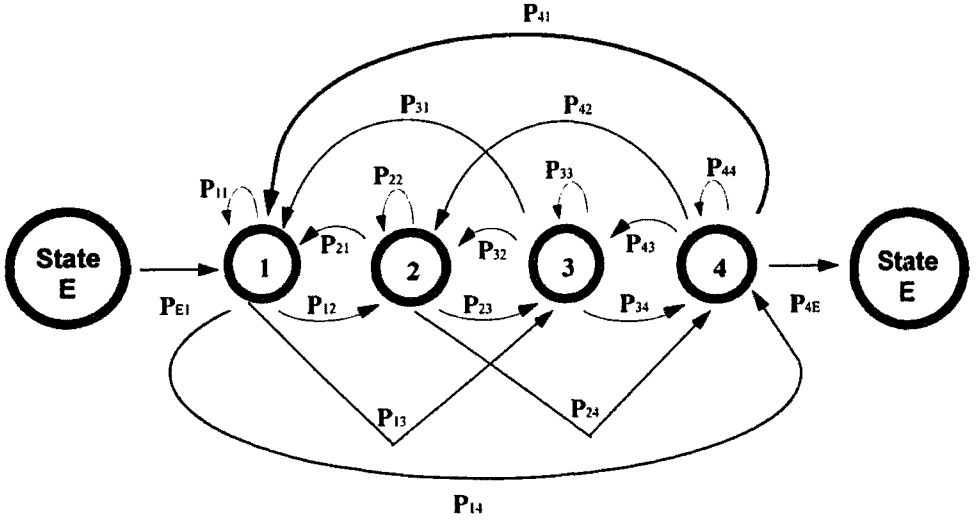
and the following matrix:

State t+1

			E	1	2	3	4	
		/						\
S	E	|	P_{EE}	P_{E1}	0	0	0	|
t	1	|	0	P_{11}	P_{12}	0	0	|
a	2	|	0	P_{21}	P_{22}	P_{23}	0	|
t	3	|	0	0	P_{32}	P_{33}	P_{34}	|
e	4	|	P_{4E}	0	0	P_{43}	P_{44}	|
		\						/

t

In the general case for *Internal-Iteration*, all states are but one time unit away from any other state:



Resulting in a more complete matrix:

			State t+1					
			E	1	2	3	4	
		/						\
S	E	|	P_{EE}	P_{E1}	0	0	0	|
t	1	|	0	P_{11}	P_{12}	P_{13}	P_{14}	|
a	2	|	0	P_{21}	P_{22}	P_{23}	P_{24}	|
t	3	|	0	P_{31}	P_{32}	P_{33}	P_{34}	|
e	4	|	P_{4E}	P_{41}	P_{42}	P_{43}	P_{44}	|
		\						/

t

Stochastic Orderings

For each of the above Markov models, two presumptions have been made. First, we have assumed that the expected time to completion, $E[T_i]$, for each function is known. This provides a basis upon which to assign a probability for each P_{xx}, all of which lay on the diagonal of the Markov matrix. Specifically, this probability is defined as,

$$1 - \frac{Incr}{E[T_x]}$$

where ***Incr*** is the time increment between each discrete Markov time unit. Thus, if the expected time for state 2 is 120 days ($E[T_2]$=120) and the time increment is five days (Incr=5), the assigned probability for recurrence of state 2, P22, would be 1-(5/120) = 0.9583.

Under such a time increment, P_{xx} = 0.99 would correspond to an expected time of 500 days, while P_{xx} = 0.80 would correspond to an expected time equal to 25 days. If P_{xx} = 0, the process is required to leave the state and proceed to some other state at the next time interval. Of course, it should be understood that, under this time defining methodology, no function is completed faster than the smallest time interval. Since the time intervals are arbitrary, this does not pose a problem: we can just set the time interval to a number less than the smallest non-zero service time, min{E[Tx]|E[Tx]>0}.

The second assumption concerns our knowledge of the transitional probabilities from one state to another. Using the requirement that each row of the Markov matrix must sum to unity, we can determine the sum of all remaining probabilities on a given row. Specifically, we start with this unity requirement:

$$\sum_{j=1}^{n} P_{ij} = 1 \quad \text{for all } i$$

Separating the diagonal values, P_{ii}, from the summation, we have:

$$\Longrightarrow \quad \left(\sum_{i \neq j} P_{ij} \right) + P_{ii} = 1 \quad \text{for all } i,j$$

Rearranging, to solve for the non-diagonal values,

$$\Longrightarrow \quad \sum_{i \neq j}^{n} P_{ij} = 1 - P_{ii} \quad \text{for all } i,j$$

Thus, simply enough, the sum of the non-diagonal probabilities on a row is merely 1 less the diagonal (self-recurrent) probability for that row. Coupled with this fact, we have assumed that the relative likelihood of proceeding from one state to another is known. (e.g., "From state 2, the process is twice as likely to proceed to state 3 as it is to proceed to state 1 or 4") This is accounted for by arithmetically computing weights as follows:

1. For a given i, determine the relative likelihood of non-recurring transitory probabilities (i.e., P_{ij} where i<>j for all j).

2. Multiply the each of the relative likelihoods by $(1-P_{ii})$

   Example: For relative likelihoods of (1/3, 1/2, and 1/6) for (P21, P23, and P24) and P22 = 0.95, we have

   P21 = (1/3)(1-.95) = 0.0167

   P23 = (1/2)(1-.95) = 0.025

   P24 = (1/6)(1-.95) = 0.0083

A natural result of defining and computing the elemental probabilities of each state of the Markov matrix is an automatic "ordering" of functions. This means that an "averaged" development process, when characterized by Markov-type probabilities, can be automatically, objectively determined, regardless of the graphical depiction by IDEF0 or process-flow model developers. What is needed is an accurate assessment of the degree of interfacing between functions, and the resulting transitory probabilities. A more sophisticated version of this automatic ordering methodology is suggested as a potential management tool. Work has begun on just such a tool, though it is not discussed here.

Determining Long-Run Behavior

Markov chains are perhaps most remarkable in their ability to determine long-run behavior in relatively complicated, but well-defined processes. Thus, the ability for us to

determine the long-run behavior of any of the model types presented here is a trivial task, given that we possess accurate probability estimates. Using accurate estimates, we can calculate expected completion time for the entire process, various conditional and interim completion times (e.g., "Given that the process is currently in state (2), how long can I expect to wait for it to arrive in state (4)?"), and percent of time that the system is in any particular state.

However, there is a more significant result which arises out of this simple type of structuring. This comes about from examining how such models operate in both the short-run and the long-run.

The great strength of Markov chains is also a weakness in this type of analysis: the assumption that transition probabilities are well known. Even if we suppose that the transitional probabilities are reasonably correct, say ±2%, we can fool ourselves into believing that the long-run behavior is understood. Why is this the case? The answer lies in the fact (assumption) that the Markov states are considered to be memoryless states: they do not base their next action upon previous actions or paths through the state space. While ±2% may appear to be very accurate, the repeating nature of this process can blow this seemingly small variation way out of proportion, resulting in observed differences of up to 30% in residence time for a particular state. Of course, this exploding variation depends upon the specific interrelations among states. In experiments with Markov models, it is apparent that this sensitivity is even more significant when there are more states and close-to "balanced" transition probabilities (i.e., each probability is close to zero).

Expanding Resolution

What does this entail for product development managers? Essentially, it means that a small change, or perturbation, in communications may result in large changes in system-wide behavior. Further consider that the systems presented here are only simple, conceptual models of the new product development process. As indicated earlier, our experience is that system models typically need significantly more than four functions to be representative of actual systems. Thus, in practice, more detailed models are inevitable.

This begs the question, "What does the behavior look like for more complicated systems?" Does the behavior become robust or more degenerate? Does it depend more and more upon a few states, or less and less?

Let us conceptualize a framework which permits reasonable, though complete understanding of a more complicated system --a system characterized by more numerous, more detailed functions. Consider that the four high-level functions introduced earlier can each be decomposed into three sub-functions, for a total of twelve functions:

(1) Define & Review Requirements
  (1.1) Review Customer Needs
  (1.2) Develop/Refine Conceptual Requirements
  (1.3) Develop/Review Detailed Requirements

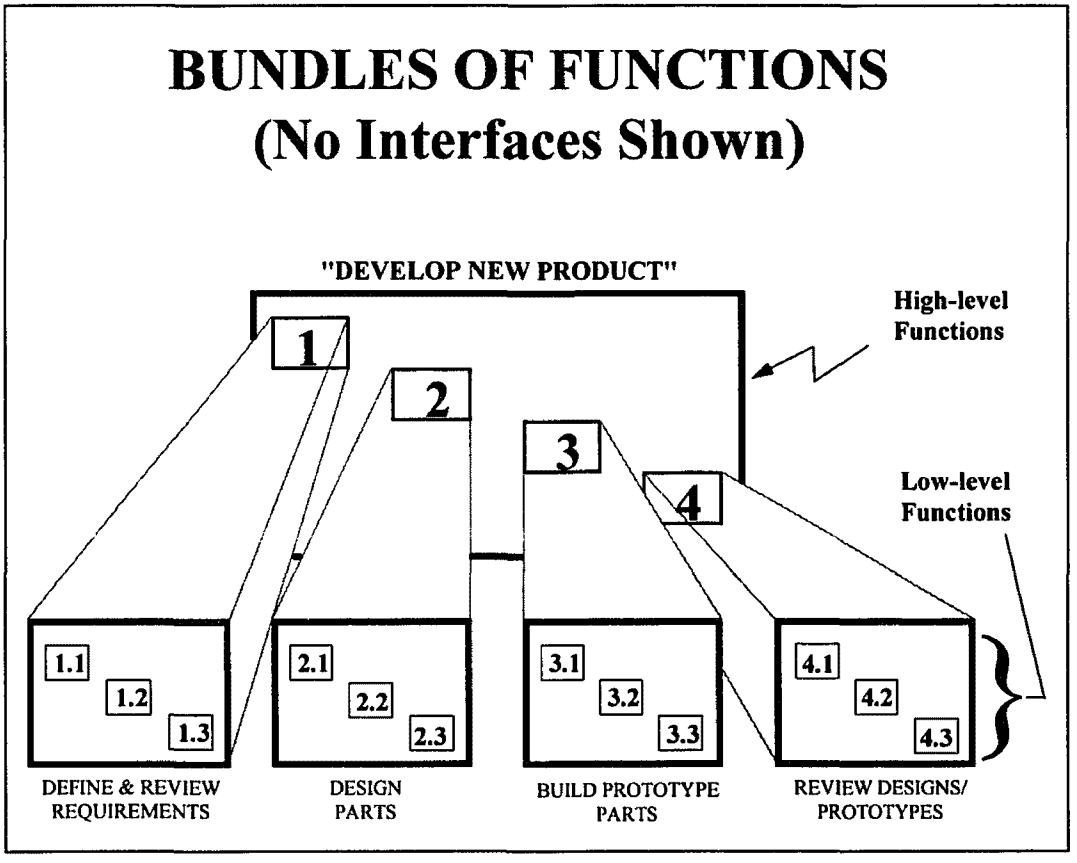
(2) Design Parts
  (2.1) Develop Conceptual Designs
  (2.2) Perform Research
  (2.3) Develop Detailed Design

(3) Build Prototype Parts
(3.1) Plan Build Process
(3.2) Develop Tools, Dies & Fixtures
(3.3) Construct Prototypes

(4) Review Designs/Prototypes
(4.1) Conduct Physical Tests
(4.2) Compare Parts to Designs
(4.3) Compare Parts & Designs with Requirements

These twelve functions, in total, compose the same operation of the basic four, albeit with more detail. Exhibit H.8. demonstrates the bundling of these child functions into the parent diagram, using IDEF0 modeling structure.

EXHIBIT H.8.



Without attempting to graphically depict all 144 potential interfaces between these lower-level functions, let us directly develop the Markov matrix of this higher resolution system:

$$
\begin{pmatrix}
P_{EE} & P_{E,1.1} & P_{E,1.2} & P_{E,1.3} & P_{E,2.1} & P_{E,2.2} & P_{E,2.3} & P_{E,3.1} & P_{E,3.2} & P_{E,3.3} & P_{E,4.1} & P_{E,4.2} & P_{E,4.3} \\
0 & P_{1.1,1.1} & P_{1.1,1.2} & P_{1.1,1.3} & P_{1.1,2.1} & P_{1.1,2.2} & P_{1.1,2.3} & P_{1.1,3.1} & P_{1.1,3.2} & P_{1.1,3.3} & P_{1.1,4.1} & P_{1.1,4.2} & P_{1.1,4.3} \\
0 & P_{1.2,1.1} & P_{1.2,1.2} & P_{1.2,1.3} & P_{1.2,2.1} & P_{1.2,2.2} & P_{1.2,2.3} & P_{1.2,3.1} & P_{1.2,3.2} & P_{1.2,3.3} & P_{1.2,4.1} & P_{1.2,4.2} & P_{1.2,4.3} \\
0 & P_{1.3,1.1} & P_{1.3,1.2} & P_{1.3,1.3} & P_{1.3,2.1} & P_{1.3,2.2} & P_{1.3,2.3} & P_{1.3,3.1} & P_{1.3,3.2} & P_{1.3,3.3} & P_{1.3,4.1} & P_{1.3,4.2} & P_{1.3,4.3} \\
0 & P_{2.1,1.1} & P_{2.1,1.2} & P_{2.1,1.3} & P_{2.1,2.1} & P_{2.1,2.2} & P_{2.1,2.3} & P_{2.1,3.1} & P_{2.1,3.2} & P_{2.1,3.3} & P_{2.1,4.1} & P_{2.1,4.2} & P_{2.1,4.3} \\
0 & P_{2.2,1.1} & P_{2.2,1.2} & P_{2.2,1.3} & P_{2.2,2.1} & P_{2.2,2.2} & P_{2.2,2.3} & P_{2.2,3.1} & P_{2.2,3.2} & P_{2.2,3.3} & P_{2.2,4.1} & P_{2.2,4.2} & P_{2.2,4.3} \\
0 & P_{2.3,1.1} & P_{2.3,1.2} & P_{2.3,1.3} & P_{2.3,2.1} & P_{2.3,2.2} & P_{2.3,2.3} & P_{2.3,3.1} & P_{2.3,3.2} & P_{2.3,3.3} & P_{2.3,4.1} & P_{2.3,4.2} & P_{2.3,4.3} \\
0 & P_{3.1,1.1} & P_{3.1,1.2} & P_{3.1,1.3} & P_{3.1,2.1} & P_{3.1,2.2} & P_{3.1,2.3} & P_{3.1,3.1} & P_{3.1,3.2} & P_{3.1,3.3} & P_{3.1,4.1} & P_{3.1,4.2} & P_{3.1,4.3} \\
0 & P_{3.2,1.1} & P_{3.2,1.2} & P_{3.2,1.3} & P_{3.2,2.1} & P_{3.2,2.2} & P_{3.2,2.3} & P_{3.2,3.1} & P_{3.2,3.2} & P_{3.2,3.3} & P_{3.2,4.1} & P_{3.2,4.2} & P_{3.2,4.3} \\
0 & P_{3.3,1.1} & P_{3.3,1.2} & P_{3.3,1.3} & P_{3.3,2.1} & P_{3.3,2.2} & P_{3.3,2.3} & P_{3.3,3.1} & P_{3.3,3.2} & P_{3.3,3.3} & P_{3.3,4.1} & P_{3.3,4.2} & P_{3.3,4.3} \\
0 & P_{4.1,1.1} & P_{4.1,1.2} & P_{4.1,1.3} & P_{4.1,2.1} & P_{4.1,2.2} & P_{4.1,2.3} & P_{4.1,3.1} & P_{4.1,3.2} & P_{4.1,3.3} & P_{4.1,4.1} & P_{4.1,4.2} & P_{4.1,4.3} \\
0 & P_{4.2,1.1} & P_{4.2,1.2} & P_{4.2,1.3} & P_{4.2,2.1} & P_{4.2,2.2} & P_{4.2,2.3} & P_{4.2,3.1} & P_{4.2,3.2} & P_{4.2,3.3} & P_{4.2,4.1} & P_{4.2,4.2} & P_{4.2,4.3} \\
P_{4.3,E} & P_{4.3,1.1} & P_{4.3,1.2} & P_{4.3,1.3} & P_{4.3,2.1} & P_{4.3,2.2} & P_{4.3,2.3} & P_{4.3,3.1} & P_{4.3,3.2} & P_{4.3,3.3} & P_{4.3,4.1} & P_{4.3,4.2} & P_{4.3,4.3}
\end{pmatrix}
$$

In general, this matrix expansion takes the following form for any desired individual transition probability, P_{ij}:

$$P_{ij} = \begin{pmatrix} P_{i.1,j.1} & P_{i.1,j.2} & P_{i.1,j.3} & \cdots & & & P_{i.1,j.x} \\ P_{i.2,j.1} & P_{i.2,j.2} & & & & & \\ P_{i.3,j.1} & & P_{i.3,j.3} & & & & \\ \vdots & & & \ddots & & & \\ & & & & \ddots & & \\ & & & & & \ddots & \\ P_{i.x,j.1} & & & & & & P_{i.x,j.x} \end{pmatrix}$$

where x is the maximum number of sub-functions for any given (hi-level) state.

To permit the matrices to bundle properly, one should use x sub-functions in each matrix, even if that particular state is composed of less than x sub-functions. (For example, if a state has 3 sub-functions, and x=5, then set $P_{i.4,j.4} = P_{i.4,j.5} = P_{i.5,i.4} = P_{i.5,j.5} = 0$.)

In essence we have created a more detailed Markov model, using the decomposition structure seen in the IDEF0 model. Note that this stochastic matrix still has the requirement that each row must sum to unity. To ensure appropriate behavior, we further require that the sum of each cell of the sub-matrices equals the overall transition probability which its own sub-matrix represents. Thus, if there is a probability of

remaining in a state, this is adequately accounted for in that state's sub-matrix. Transferring among sub-states at each time interval is considered one method of remaining in a high-level state[84].

Thus, for the $P_{1,1}$ sub-matrix, we have:

$$P_{1,1} = \begin{pmatrix} P_{1.1,1.1} & P_{1.1,1.2} & P_{1.1,1.3} \\ P_{1.2,1.1} & P_{1.2,1.2} & P_{1.2,1.3} \\ P_{1.3,1.1} & P_{1.3,1.2} & P_{1.3,1.3} \end{pmatrix}$$

The value of $P_{1,1}$ is the sum (not the determinant) of these sub-matrix values:

$$P_{1,1} = P_{1.1,1.1} + P_{1.1,1.2} + P_{1.1,1.3} + P_{1.2,1.1} + P_{1.2,1.2} + P_{1.2,1.3} + P_{1.3,1.1} + P_{1.3,1.2} + P_{1.3,1.3}$$

It should be clear that the 13x13 matrix (recall that the first column and first row are dedicated to the "outside environment", E) is much more difficult to comprehend than the simple 5x5 general case presented earlier. Yet, we can develop a straightforward

---

[84] It should be realized that this procedure is only valid for conditions where there is a finite probability of leaving the high-level state from any "sub-state." Validity is seen to break down when one considers the a condition in which a sequence of sub-states is required (with probability = 1) before possible "escape". This would have net effect of making the retention probability of the high-level state equal to one, for one or more time intervals. When this happens, the Markov process is no longer stationary.

mechanism for reducing this complicated system to a more simple 5x5. Simplification of the 13x13 matrix is possible by bundling and summing states in each row of the matrix. A procedure for doing this has been developed, but is not presented here.

Based upon our observations and functional modeling of organizations in the field, a 13x13 matrix, corresponding to 12 discrete functions, does not nearly approach the highly complicated nature of true-to-life innovative product development.

Thus, if we are to continue using the Markov representation for conceptualizing the innovative development process, we need a method for incorporating even *larger* systems. This issue has "automation" written all over it and is addressed at the end of this chapter.

Thus far, we have conceptualized process and functional models, offered a few examples of how the product development process can be structured using such conceptualizations, and introduced discrete stationary Markov models as a preliminary framework for analysis. Next we shall discuss some facets of the development process which give us reason to relax the stationary, discrete, and linear restrictions on the Markov models. Though we shall leave the Markov models developed here behind, we continue to incorporate the principles developed here into the remaining efforts to model product development.

Experiential Structures

Throughout the field studies, it became widely apparent that schedule and budget constraints were predominant controls on development functions. Though we will not examine budget controls directly in this analysis, it is evident that they do have effect on total process time[85].

Explicitly, scheduling controls affected the completion of local development tasks. In numerous cases, early activities were compressed, to meet schedule constraints, only to show up again as rework at a later time in the process. It was readily observed that project participants experienced a pattern of initial enthusiasm, followed by general decline of interest. As the expected completion date approached, however, intensity of project personnel increased. This often comprised large degrees of overtime and, at times, allocation of additional personnel[86].

Despite often heroic efforts on the part of design personnel near scheduled due dates, completion times in excess of pre-defined schedules were prevalent. Under this condition, several different effects were observed. In a few cases, the intensity level of participants continued beyond the schedule date until the task was complete. More often, the

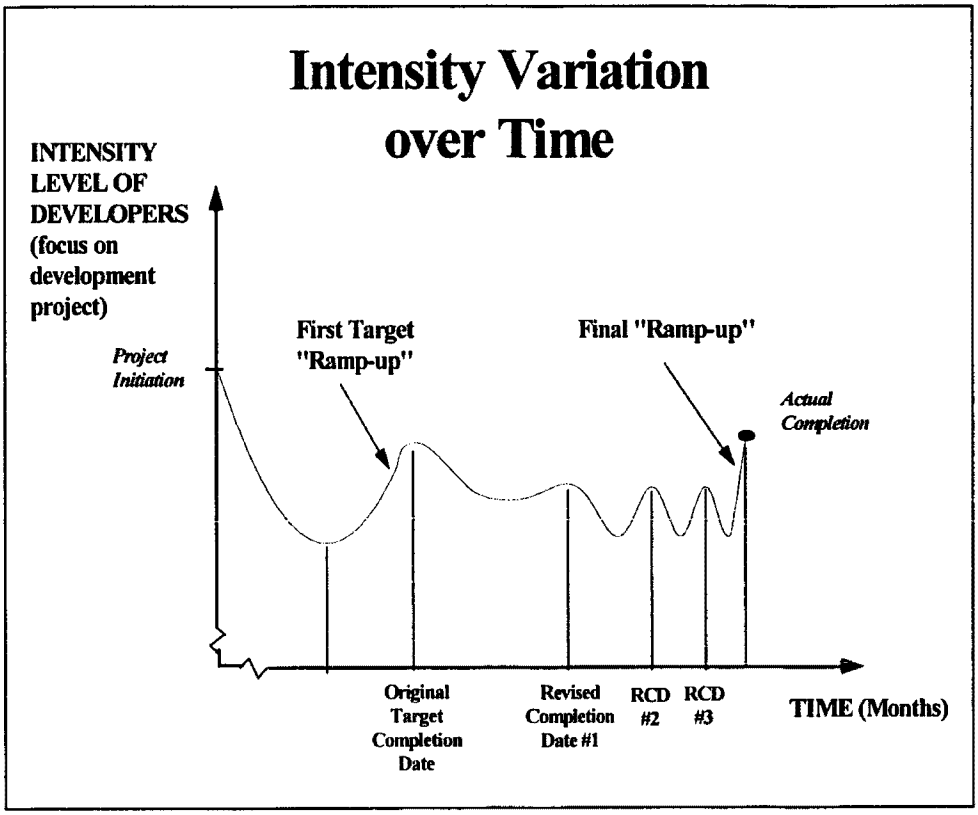
---

[85] From a simplistic standpoint, budget must have influence; for if there is no money to pay for equipment, materials, and personnel, work stops. More subtly, however, the manner in which budgets are allocated by management seems to influence the attitude of development personnel. Without confidence of their own management's commitment, developers indicated less desire to expand the "development window", and merely complete their tasks with minimal personal hassle or conflict. It is suggested this change in attitude affects personal performance, which may adversely affect the number of functional iterations alluded to in the previous section.

[86] At a particular site, this characteristic of the process was even given a name - "Rambo Engineering." When engaged in this mode, developers gave the development their undivided attention, forsake unnecessary or distractive tasks, and banned middle management limitations by involving senior management. In a sense, caution was thrown to the wind when in this mode. Various localized "policies" and existing "ways of doing business" were questioned and disregarded if they did not help satisfy the current "crisis". When modeling certain military activities, a similar distinction was made: should we look at "war-time" mode or "peacetime" mode?

scheduled due date was postponed well prior to the original due date. When this occurred, the intensity level only moderately (and temporarily) increased. Further, when multiple due date extensions occurred for a single task, the intensity level generally decreased, though not to the same low levels of the first intensity cycle. As due dates arrived, however, an increase in local intensity level appeared to be the norm. Exhibit H.9. shows this intensity variation as a function of time during development.

EXHIBIT H.9.



Experience level of engineers was directly attributed to development efficiencies by many participants, especially those in Europe. The measure and importance of experience

composes many cultural considerations which are outside the domain of this study. However, experience level does appear important enough to deserve attention in model formulation.

Through interviews and direct observations, it was evident that experience is a mixed bag, offering both advantages and disadvantages to the innovative product development process. For conceptual development, experience seemed to have little or no effect on performance. Early in the design process, however, when interfacing considerations were needed, experience was repeatedly shown to be invaluable. In the application and incorporation of state of the art practices, however, experience appeared to be a detriment, as existing paradigms influenced attitude and behavior. In almost every interview in which the issue was raised, respondents indicated that an experienced team leader was important.

In many functions involved with innovative product development, a learning phenomena was apparent, regardless of experience level. This may be attributed to the extent of familiarity with the specific item under development. In the earliest stages of development, when requirements and physical characteristics are barely understood, the performance of an activity seems to be sacrificed, to enable learning more about the intended product characteristics. As development participants were "re-exposed" or acclimated to the product, their functional performance (in time, expense, and quality of work) appeared to increase.

Given these observations, it is reasonable to postulate that the completion time (and thus the instantaneous completion probability) for any given task is a complex function of the

time spent on the task, the time remaining on a the completion schedule, the number of iterates of the particular function in the process, and the relevant experience level of personnel performing the task. Thus, the non-stationary Markov process is likely an impractical over-simplification of true system behavior. Further, because of the observed variances of functional performance with time, and the inherent round-off errors which may result from conducting discrete time analyses, it may also be unreasonable to use discrete-time Markov decision processes.

The continuous time, non-stationary Markov decision model offers a better conceptual fit for the conditions of ***decision drift***, or changes in the operating rules or system structure. Specifically, it offers us the opportunity to examine the effects of two types of controls on the system which are highly relevant to this research: ***infinitesimal generators*** and ***direct impulsive control***. Descriptions of these phenomena, as used in analysis of non-stationary Markov decision-drift processes, are described in (Van der Duyn Schouten (1986), Van der Duyn Schouten (1983)).

Infinitesimal generators can be thought of as those "internal" aspects of a function which affect its service rate or other measure of performance. In the Markov models mentioned already, these are represented by internally generated changes to the transition probabilities. It is proposed here that factors such as operator experience, functional learning curve, and process/function dwell time are representative infinitesimal generators. Such generators have an indirect impact on the evolution of the process, as they generally only have affect on a specific function. Naturally, each function has its own set of continuously changing infinitesimal generators.

Direct impulsive controls are considered instantaneous changes or shocks to the state space of the system. For our purposes, these may be considered "external" influences on the behavior of a function and its larger "system." It is suggested here that they are representations of managerial influence on the development process (e.g., changes in communications policy or capability, divisional/departmental restructuring, promotions, firings, new schedules, new development requirements, etc.). Direct impulsive controls may range from a change in state-to-state (function-to-function) transitory probabilities to the addition/removal of states.

These two categories of controls to the system should be incorporated in any realistic model of the development process. It should be clear by now, however, that infinitesimal generators may naturally be defined by the system after direct impulsive controls have been established. Thus, such controls are convenient to consider as management directives or policies to which infinitesimal generators "should" respond[87]. In economics, infinitesimal generators may be considered the actions of the microeconomic players (e.g., individuals and businesses); direct impulsive controls are analogous to the action of macroeconomic players (e.g., the Federal Reserve Bank, U.S. Treasury, etc.). As in economics, the cause-effect relationship between macro changes and micro responses have been repeatedly shown to be ill-understood. This research does not attempt to explain every facet of the relations between direct impulsive controls and infinitesimal generators, but rather open the possibilities to the research and industry environments.

---

[87] The issue of the response rate of individuals to a management directive or policy is directly relevant to the concept of *lag*. Later in this chapter, we will briefly address the limits of the communication process, and how impatience and miscommunication can foster great instability in the tasks at hand.

It is proposed here that direct impulsive controls and infinitesimal generators can be modeled as generic entities, perhaps eventually through highly sophisticated mathematical means. The success of such modeling is expected to directly depend on further development in the cognitive sciences, especially communication theory, "artificial intelligence", and psychology, as well as continued increases in computational methods. Lacking sufficient development in these areas, we relegate ourselves to simplified generators and controls. Even with such simplification, we shall see how complex this continuously changing system can be.

Let us define two classifications of variation which can help clarify our analysis: ***time-based*** and ***frequency-based*** variation. These have been developed as infinitesimal generator variations, but future research will surely apply them to direct impulsive controls variations as well.

Time-based variations are those changes in functional performance (and thus system performance) which are dependent upon the amount of time which has been spent conducting the function. Thus, an individual who has been working on an activity in a particular manner at time period t, and performs differently at time $t+1$ would be considered to be behaving with time-based variation.

Frequency-based variation ignores the time spent performing functions, and only relates to the number of times a function has been performed. Thus, a function which is being performed for the nth time in the process may produce different results than when it performed the previous $n-1$ times.

Without knowing the source of variations, it is often difficult to objectively pinpoint actual variation as time-based or frequency-based. To further complicate understanding, such variations may be overlayed on each other, in that they may be occurring simultaneously on a given function. Short of conducting Fourier transformations (which are limited to numeric data intensive measures), we have little choice but to make educated guesses about which variation or set of variations is occurring at a given time.

However, if we are to establish realistic structures of the process of development, and perform subsequent simulation or algebraic analysis of it, we do need to contemplate the differences between these classes of variation.

A New Analysis Structure: The CPP

Up until this point, we have asserted that Markov modeling is a reasonable paradigm for describing and analyzing the development process. The Markov processes described or mentioned thus far do go a long way for conceptualizing the observed fact that functions are performed with continually changing levels of performance, do not always follow pre-described paths, and are recursive in nature.

The Need for an New Structure

Any Markov description, however, has a severely limiting characteristic which limits our particular analysis: singular state space. This means that each function in the Markov model can only be performed mutually exclusive of other functions; two or more functions cannot be performed simultaneously. This is a built-in restriction of Markov

processes, so that the concept of transition probabilities can hold. Unfortunately, the observed characteristic of simultaneous/parallel processing in development (well known in industry as *Concurrent Engineering*) makes this singular state characteristic too limiting for useful, credible analysis.

Even conversion from the discrete Markov models presented in the previous section to non-stationary, semi-Markov, or continuous-time Markov models would still be lacking. No matter the time quantization or state space growth characteristics, the Markov process still requires mutually exclusive state servicing.

What other methodologies capture the spirit of the structures developed here, without this singular state restriction? Several approaches were examined, including state-augmented decision trees and PERT networks. A methodology whereby simultaneous Markov processes operate in parallel was contemplated, but dropped from further consideration, as mathematical (and visual!) complication grew beyond reason, for even small (2 state) processes[88].

After several false starts, it was determined that PERT networks and Markov processes still came closest to satisfying observed conditions, though neither could be manipulated sufficiently on its own. As alluded to earlier, Markov processes have no provision for concurrency, but do permit looping behavior. PERT networks permit simultaneous

---

[88] Though this is difficult to mathematize a priori, there is an encouraging technique which may help us to understand *changing* definitions of states: cellular automota. If and when automota analysis, or some variant of it can help pre-define (and predict) states, then the dynamic interface analysis presented in this work can be coupled with it. The result would be a holistic dynamic analysis method which incorporates changing interfaces and changing underlying functional structures. An unheard of capability to date, these analysis methods could (will?) change project management philosophies about innovative (and routine) product development.

operations and dependencies (with ***and/or*** gates), but only in a pre-defined, unidirectional manner. Even stochastic PERT networks were shown to only be useful for visualizing variation in service times; the concept of continually changing dependencies appears to be outside of the research domain to date.

Thus, a new analysis process was conceived. It is called the ***Complex Process Path*** methodology. It provides for concurrency, feedback, varying dependencies, and changing service times. It accommodates both time-based and frequency based internal generators, and thus is self-induced, as real human systems seem to be. It is also possible to permit direct impulsive control. Unlike PERT, it does not require a particular beginning or end to each function within the process, except for a more global indicator of when all functions have been performed "adequately." Thus, the process may start at any node or set of nodes. Likewise, it may finish at any node or set of nodes, depending on the self-induced behavior of the system, although it is most likely that the system will always be waiting for some particular node to "finish" the process.

The Complex Process Path (CPP, for short) is composed of several components, which will be immediately familiar to those familiar with both Markov and PERT networks. Yet, CPP is not an outgrowth of a Markov process, nor an evolutionary PERT network. It is a different conceptualization technique, which requires different analytical methods. There is still some question of how even simple CPP's could be analyzed algebraically. Thus far, it appears that computational analysis, simulation, may be the only realistic method for analyzing CPP's.

Appendix I: Engineering Resource Allocation

For a department with 4 functions, composed of 1 MAIN function and 3 INFO functions. This table provides for the conditions of up to 3 engineers per department. The first page of this table considers allocations of one and two engineers. The following pages consider the nine possible allocations of three engineers.

Allocation	Concurrency Among Functions	Resource Utilization	Station Utilization (for each Dept.)	Notes
1/1/1	NONE	≤100% (if work exists, then not idle)	3+ stations always idle	One person can only perform one task at a time.
2/1/1	Any two functions	Up to 100% utilization if there exists work to be processed at 2 or more stations.	2+ stations always idle	If there is work at two or more stations, then any two stations can be worked on at any time.
2/1/2	NONE	When MAIN being performed, then 50% idle. When INFO being performed, then 100% utilization possible.	3+ stations always idle	Maximization of resource utilization does not provide maximum MAIN processing. Same system performance as 1/1/1 allocation.
2/2/1	NO concurrency permitted when MAIN function being performed. When MAIN not being performed, concurrency possible among any two INFO functions.	100% utilization when MAIN being performed. ≤100% when INFO being performed.	2+ stations always idle	MAIN and INFO may not be conducted at the same time.
2/2/2	NONE	100% utilization when MAIN being performed.	3+ stations always idle	Two engineers only work together (siameised). They work as if they are one person--no simultaneous operations. Same system performance as 1/1/1 & 2/1/2.

3/1/1	Any three functions	Up to 100% utilization when there is work at three or more stations.	1+ stations always idle	If work exists at three different stations, then engineers will work simultaneously.
3/1/2	Only between MAIN and one INFO function. NONE under all other conditions.	100% when MAIN and one INFO function being performed. 33% when only MAIN being conducted (2 odd-men out) 66% when only one INFO being conducted (1 odd-man-out)	2+ stations always idle.	No concurrency among INFO stations. This is the same as having only 1/3 the information processing capability when there is all three kinds of info in INFO buffer.
3/1/3	NONE	33% when MAIN being performed. 100% when one INFO function being performed	3+ stations always idle	Only one function can be done at a time. Note, again, that maximization of resource utilization does not provide for MAIN function. Same system performance as 1/1/1, 2/1/2, and 2/2/2.
3/2/1	Between MAIN and one INFO function or Among three INFO functions	Up to 100%	1+ station always idle	Resources never idle because of allocation interference. Permits a single INFO function to process while MAIN is being performed.
3/2/2	NONE	Up to 66%	3+ stations always idle	Same system performance as 2/1/2, 2/2/2, or 1/1/1, but costs more to have one or two more engineers.

3/2/3	NONE	100% when INFO function being performed. 66% when MAIN function being performed.	3+ stations always idle	Same system performance as 1/1/1, 2/1/2, 2/2/2, 3/1/3 or 3/2/2--just more $$.
3/3/1	None when MAIN being performed. Three INFO functions can be performed concurrently when MAIN not being performed.	100% when MAIN being performed. ≤100% when INFO functions being performed.	3 stations idle when MAIN being performed. From 1 to 3 stations idle when INFO being performed. No more than 3 stations operate at once.	Any and all idle resources can be attributed to lack of available input, not allocation interference.
3/3/2	NONE	100% when performing MAIN function. 66% when performing INFO function.	3+ stations always idle	Same system performance as 1/1/1, 2/1/2, 2/2/2, 3/1/3, and 3/2/3.
3/3/3	NONE	100% when performing any function.	3+ stations always idle.	All idle time comes from lack of inputs. This is just like 1/1/1 or 2/2/2 (siamesed resources)

Appendix J: CPP Model Structures

The following formal structures were tested during model construction. In addition to these models, several less-formalized experimental models were developed, which either "crashed", did not work as asked (via self-induced programming glitches!), or were not interesting enough for further review. This explains the odd numbering sequences for the MODELFORM names.

COMPLEX1

- ***Objective***: To see how variation of across-the-board station efficiency effects output of the system. (i.e., does doubling the speed of the system components necessarily double its output?)
- ***Layout***: Four Departments. Four stations per dept. (3 INFOx stations and 1 MAIN station) All INFOx stations connected to all other depts. Prototype flow is sequential.
- ***Resources***: Unlimited (any station can operate at any time, regardless of status of other stations).
- ***Transition Probs***: 33% chance of sending one unit of info to another department. NO DISSIPATION OF INFO.
- ***Information Generation***: Only from MAIN functions and only 75% of the time. Each other dept. has equal chance of getting this one "piece" of info.
- ***MAIN output***: Always send out one prototype, with 100% probability. With this prototype, one "new" piece of info is generated 75% of the time. Remaining 25% of the time, no info generated.

COMPLEX3

- ***Objective***: To understand the basic effects of resource limitations on the system.
- ***Layout***: same as COMPLEX1; Visually improved to see each station within a department operate.
- ***Resources***: Resources are first introduced in this MODELFORM. These were varied over several test runs, until "switching" effect was apparent. 3/3/1 resource level seemed to provide interesting basis for further model runs.
- ***Transition Probs***: same as COMPLEX1 (still no dissipation)
- ***Information Generation***: same as COMPLEX1
- ***MAIN output***: same as COMPLEX1
- ***Other***: Run time held to 500 days for this MODELFORM.

COMPLEX5

- ***Objective***: To observe the warm-up dynamics of the system, by varying run-time.
- ***Layout***: same as COMPLEX3
- ***Resources***: held at 3/3/1
- ***Transition Probs***: same as COMPLEX3
- ***Information Generation***: same as COMPLEX3
- ***MAIN output***: same as COMPLEX 1
- ***Other***: Run time was an additional variable in this model. (As low as 50 days and as high as 2500 days)

COMPLEX6

- ***Objective***: To examine the specific "switching" effects which occur with different resource allocations, over the duration of a full-scale run (2500 days).
- ***Layout***: same as COMPLEX3
- ***Resources***: Allocations variable
- ***Transition Probs***: same as COMPLEX3
- ***Information Generation***: same as COMPLEX3
- ***MAIN output***: same as COMPLEX1

COMPLEX7

- ***Objective***: To examine the effects of information and prototype interface profile changes. Also, to better understand the effects of prototype buffers on this system.
- ***Layout***: Prototype buffer size = 5
- ***Resources***: 3/3/1
- ***Transition Probs***: "Friendly neighbor" paradigm used. (Inter-departmental communication drops as depts. get "further away" from each other.) DISSIPATION incorporated, depending upon specific department's location in the system.
- ***Information Generation***: MAIN functions generate bundles of information with every prototype release. Bias towards sending information downstream , but do send rework information back upstream at times. Allocations of this is variable.
- ***MAIN output***: Prototypes released only 50% of the time that MAIN is performed. Bigger bias towards rework information transmittal when no prototype sent.

COMPLEX8

- ***Objective***: With "gelled" system structure, modify each variable in isolation to better understand its impact. Specifically, change processing rates to visualize their impact on overall system performance. Given

previous system understanding, incorporate communication counts as impetus for quality measures.

- ***Layout***: same as COMPLEX3
- ***Resources***: 3/3/1
- ***Transition Probs***: Fixed. (Same as COMPLEX7)
- ***Information Generation***: Fixed after run L.
- ***MAIN output***: Fixed after run L.

COMPLEX9

- ***Objective***: To better understand the effects of various resource allocation strategies. Beyond the resource mix, which is technically a structural modification to the system, this model structure is the same as COMPLEX8.
- ***Layout***: same as COMPLEX3-COMPLEX8
- ***Resources***: Variable. 14 allocations, ranging from 1/1/1 to 3/3/3.
- ***Transition Probs***: Fixed.
- ***Information Generation***: Fixed.
- ***MAIN output***: Fixed.

Appendix K: Quality Assessment Strategies

The assessment of quality in design or manufacturing is observed to be a highly variable and sometimes dubious exercise. A design which is valued as the epitome to one quality judge may be considered wasteful excess (inefficient) to another. It is probably accurate to declare that measurement of quality in a design is as dependent on the objectives against which the design is being judged as it is on the diligence and fairness upon which "quality data" is collected. As a result, design quality assessment has become a myriad of objective, subjective, qualitative, and quantitative measures, with specific purposes known only to the judges at work.

In this appendix, the intent is to develop alternative arithmetic views of quality. By developing such views, it may be possible to develop a framework for design quality assessment, under which the above myriad may be better understood.

In this analysis, five different quality computations have been developed. Each computation technique uses, at its core, a fundamental indicator for communication. Stated more simply, these quality computations are dependent upon the degree of communication among developers. The first part of this appendix is dedicated to establishing a standard upon which communication is measured.

The Fundamental Communication Indicator

It was established very early in this study that communication level by itself is not the end-all to development quality. In the field studies, balancing and timing of communication were observed to be important issues, as well as the communication entropy (Shannon and Weaver, 1963) which arises from basic linguistic, technological and cognitive limitations. By establishing a robust fundamental communication indicator (FCI), such issues could be incorporated, albeit in a rather simple manner.

After many manipulations and simulated experiments with a variety of indicators, the following FCI structure was developed. This particular indicator was chosen for its simplicity, ability to favor balanced communication, and its potential for cognitive tuning, as will be addressed later in this appendix. For a four-station CPP structure, the developed FCI framework takes the following form:

$$FCI_A = \sqrt{\alpha_A BA \cdot CA + \beta_A BA \cdot DA + \lambda_A CA \cdot DA}$$

$$FCI_B = \sqrt{\alpha_B AB \cdot CB + \beta_B AB \cdot DB + \lambda_B CB \cdot DB}$$

$$FCI_C = \sqrt{\alpha_C AC \cdot BC + \beta_C AC \cdot DC + \lambda_C BC \cdot DC}$$

$$FCI_D = \sqrt{\alpha_D AD \cdot BD + \beta_D AD \cdot CD + \lambda_D BD \cdot CD}$$

where

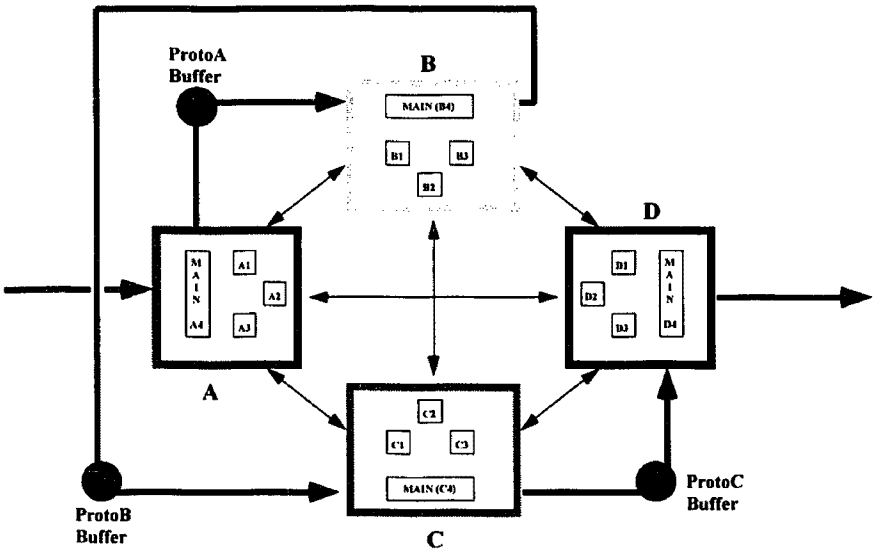
FCI_Y is the fundamental communication indicator for station Y,

XY is the number of communications processed from station X by station Y (this is also known as the XY channel),

and the α's, β's, and λ's are weighting coefficients which temper the effectiveness of individual communication channels.

The structure of this indicator is best understood by reviewing the structure of the CPP model:

Four Department CPP



In this simple, 4-station CPP structure, every station, represented with a box, has the capability of communicating with each of the other three stations. This results in n(n-1)=4(3)=12 possible communication channels. (Recall that there exist two distinct communications channels between stations X and Y: one *from* X *to* Y and one *from* Y *to* X.) Each of these channels, *XY*, is represented by an arrow between stations.

The basic FCI

One fundamental premise of FCI is that the greatest gains in communication can be made by increasing the activity on the channel with lowest existing activity. Hence, the FCI is based upon a *max-min* strategy for communication impact.

Further, FCI is based upon the principle of *triangulation*. This means that there is little or nothing to be gained from receiving more communication signals from just one source. By receiving information from multiple sources (source being synonymous with station in this closed CPP structure), the FCI of a particular station can be increased. In fact, the FCI structure developed offers zero communication indication until a signal has been received from at least two sources. Additional, though less dramatic gains are employed by the addition of a third source.

These two principles, max-min and triangulation, are conveniently illustrated with a sum of multiples structure (SMS). For station A, the basic structure is as follows:

$$SMS_A = BA \cdot CA + BA \cdot DA + CA \cdot DA$$

This structure demonstrates pairwise signals, in which the accumulated signal count of one channel synergistically affects the impact of an initial signal on a different channel. For instance, if the accumulated BA count is 3, CA=1, and DA=0, then the introduction of a single DA signal permits the second term of SMS_A to increase from 0 to 3. Additionally, the third term increases from 0 to 1. Thus, SMS_A rises from a level of 3 to a level of 7, just from the introduction of a single signal.

Unfortunately, SMS values are not comparable with the units of communication that are used for the individual channels. SMS has units of bits squared, whereas channel counts are measured in bits. To alleviate the problem of incompatible units, we linearize SMS, by taking its square root:

$$\sqrt{SMS_A} = \sqrt{BA \cdot CA + BA \cdot DA + CA \cdot DA}$$

This is known as a Basic FCI, for station A. Similar structures are just as easily developed for the other stations in the CPP model.

To see how this basic structure facilitates the principles of max-min allocation and triangulation, consider the following cases:

Case 1: Station A has received no communication signals from any stations. FCI should be zero. In fact, BA=CA=DA=0 and thus, FCI=0.

Case 2: Station A has received 10 communication signals from station B and none from stations C or D. In this scenario, via the triangulation rule established above, station A should have FCI equal to zero. In fact, with BA=10 and CA=DA=0, this is true.

Case 3: Station A has received 20 communication signals from station B, but still nothing from stations C or D. As with Case 2, the FCI for station A is still equal to zero.

Case 4: Station A has received 10 communication signals from station B (BA=10). Next, station A receives a sole signal from station C, incrementing CA from zero to one (CA=1). Still no signals have been received from station D (DA=0). Under these conditions, the FCI for station A should rise. In fact, $FCIA=((10)(1)+(10)(0)+(1)(0))^{1/2}= (10)^{1/2}$.

Case 5: Given an initial state of case 4 (BA=10, CA=1, and DA=0), station A has the opportunity to receive a single signal from either station B, station C, or station D. Which choice offers the biggest gain in information indication? If the station B signal is chosen, then BA rises to 11; FCIA rises from sqrt(10) to sqrt(11). If the station C signal is chosen, then CA rises from 1 to 2; FCIA rises from sqrt(10) to sqrt(20). Clearly, this is higher than choosing the "B" signal, demonstrating the balance (max-min) principle. If, however, the signal from station D is chosen, DA rises from 0 to 1; this raises FCIA from sqrt(10) to sqrt(21). This reveals the added benefit of both balancing and triangulation in this basic FCI computation.

Weighting scheme for FCI elements

In conjunction with this basic root-of-sum-of-multiples structure, we have incorporated a scheme to account for effectiveness of communication between specific stations in specific channel directions. This is the purpose of the alphas, betas, and gammas in the FCI.

Specifically, these are weighting coefficients which reflect the impact of deficiencies or efficiencies in communication between stations. As structured, these coefficients have the following interpretation:

α_Y is the weighting assigned to the communications channel between function Y and the two most upstream functions in the model, exclusive of function Y.

β_Y is the weighting assigned to the communications channel between function Y and the most outlying (the most upstream and the most downstream) functions in the model, exclusive of function Y.

λ_Y is the weighting assigned to the two most downstream functions in the model, exclusive of function Y.

Because these are pairwise coefficients, their value has impact on the effectiveness of more than one communications channel. Conversely, there is overlap between coefficients, in the manner that each channel's impact on FCI is affected by two coefficients. For instance, channel BA is tempered by both α_A and β_A. Yet, α_A tempers the effects of both BA and CA on the CFI_A. Meanwhile, λ_A simultaneously affects the impact of CA and DA on FCI_A. This can make assignment of coefficients less than straightforward.

Fortunately, this can be resolved through a simple algebraic manipulation of simultaneous equations. Specifically, if one knows the effectiveness of each communication channel, then the coefficients α_A, β_A, and λ_A are easily computed as follows:

$$\alpha_A = \frac{BAeff + CAeff - DAeff}{2}$$

$$\beta_A = \frac{BAeff - CAeff + DAeff}{2}$$

$$\lambda_A = \frac{-BAeff + CAeff + DAeff}{2}$$

where

XY_{eff} is the effectiveness of the communication from station X to station Y, as described below.

Any combination of channel effectiveness can be accommodated with these simple α, β, and λ coefficients. In fact, as long as the effectiveness specified for each channel is non-negative, the resulting FCI coefficients will produce a non-negative value within the square-root of the FCI. This is true, even if an individual coefficient has a negative value.

The methodology for assigning channel effectiveness scales are arbitrary. We have chosen to use a continuous 0 to 5 scale, with 5 corresponding to comprehensively efficient and effective communication between the source and the destination. According to mathematical communication theory (Shannon and Weaver, 1963), the English language contains approximately 50% redundancy, which cuts information transmission efficiency in half. By generously attributing 80% of these signals as useful, we have a channel *effectiveness* value equal to (5)x(.5)x(.8) = 2.0. If one assumes that all channels have the same capability for effectiveness (=2), then each of the FCI coefficients have value equal to one. These are the values which were used for the quality assessment methodology, which is explained in the rest of this appendix.

It should be readily apparent that the fundamental communication indicator described here is quite flexible in it ability to accommodate different levels of communication effectiveness. We have experimented with higher order CFI's, as well, in which CPP's with more than four functions are utilized. Though these are in need of more development, their characteristics are very similar to the CFI described here.

Development of Quality Assessment Strategies using the CFI

The CFI described earlier in this appendix is a linear measure of the amount of communication which takes place between a function and its "sister" functions in the CPP structure. It also takes into account the concepts of triangulation and max-min balancing. Regardless of the communication indicator used, however, there are various methods by which such communication can be interpreted as a factor in quality of design. The strategies outlined here are but some of the quality assessment indicators which could be used.

These were developed after several years of direct field observations of quality perception by development personnel in the private and defense sectors. In fact, these observations revealed that a wide variety of quality perceptions are apparent within and outside of a development organization. Quality of requirements, quality of design, quality of final product, quality of service, even quality of process descriptions of these types of activities and entities--all have been observed and judged with vastly different interpretations by designers, planners, executive managers, assembly-line operators, service technicians,

and, especially, customers and users. How then can one attribute this plethora of interpretations to a few simple equations?

The short answer to this question is that one probably cannot cover all bases with respect to quality interpretations. Rather, one can only attempt to develop understandable theories which demonstrate a substantial portion of this observed quality interpretation myriad. Then, one may test such theories against objectively collected data on quality development and human perception of such development.

The longer answer to this question is in itself a question: "How does one know that we cannot develop such theories?" Given the evolving progress of neurological network thoery and, in general, the acceleration of our understanding of human cognitive processing, it may one day be possible to attribute human interpretation/perception of quality to some elemental pattern recognition abilities. When and if such occurs, then developed indicators such as these may be proven completely wrong or conditionally correct. Until such day, however, we must do the best we can to reconcile our observations to date.

For a specific organization and application, specific assessment strategies are expected to be developed.

Nomenclature--Which FCI?

The nomenclature used here is "Qualcomp(n)", where n is a serial number, starting with 1. For the communication indicator, "FCI" is used. Because the strategies outlined here are generic and developed to demonstrate various design quality principles, it does not matter which FCI is used: FCI_A, FCI_B, FCI_C, FCI_D, or some formulation of these FCI's. For full disclosure purposes (if one must know!), the simulation data demonstrated in Chapter 6 was obtained with FCI_D as the communicative indicator of choice[89].

Time dependency of FCI

Recall from the operational structure of the CPP model that FCI is itself a function of time: the more time has expired, the more likely it is that some communication among two stations has occurred. Thus, FCI could be labeled as FCI(t). To help avoid confusion from excess terminology[90], however, we omit the (t) suffix from this variable. It is

---

[89] Station 4 was used for a number of other reasons as well. Primary among these other reasons was the lag effects demonstrated at this station. Thus, discontinuities between time expenditure and the linear FCI were more pronounced. Of secondary importance, but interesting consequence, was the observation that percieved quality levels of final, overall designs are often heavily weighted around activities in the latter stages of design.

90 In addition to simplification reasons, the (t) in FCI(t) is ignored because communication is actually an extremely complex function of time, as illustrated by the priority and rule criterion of the CPP. In fact, the

important to keep in mind that FCI does vary with time, making some of the Qualcomp(n) functions much more difficult to predict than their simple structure might imply.

The Qualcomp1 quality indicator

From the start, we were determined to keep this quality assessment strategy as simple as possible. What could be more simple that assigning a quality strategy that is identical to the FCI? In fact, the Qualcomp1 indicator is merely the FCI value, multiplied by a normalizing constant:

$$Q_1 = k_1 \cdot FCI$$

The normalizing factor, k_1, is used to limit the maximum quality value to unity. Thus, k_1 is defined as follows:

$$k_1 = \frac{1}{\max\{FCI\}}$$

The Qualcomp2 quality indicator

At times, quality is perceived as a time dependent function (specifically, one in which quality deteriorates with time). The Qualcomp2 strategy considers the effects of time expiration. In this assignment, time has an inverse effect on quality level:

$$Q_2 = k^2 \frac{FCI}{s+t}$$

where

t is the expired time from the start of the development process,
s is a constant, known as the "steepness" factor,
k_2 is a normalizing factor, which limits $\max\{Q_2\}$ to unity.

For the runs in this analysis, t is measured in days. The steepness factor, also with units of days, modulates the impact of time, particularly from the start of development until t is significantly larger than s. Specifically, larger values of s reduce the impact of early time

---

communication entropy charts described in Chapter 6 illustrate the highly variable. perhaps chaotic nature of comminication in even the simple orgaization modeled by the CPP.

expiration, resulting in flatter quality curves. For the analysis results described in Chapter 6, s=600. By definition, k_2 is as follows:

$$k_2 = \frac{1}{\max\{\frac{FCI}{s+t}\}} = \min\{\frac{s+t}{FCI}\}$$

The Qualcomp3 quality indicator

As was established very early in mathematical communication theory by Shannon, "information" can be conveniently defined as a logarithmic function of the number of choices in which one could have developed and/or interpreted a message. For instance, three on/off signals (each denoted with a "1" or "0") actually can produce eight message choices: 000, 001, 010, 011, 100, 101, 110, and 111. Although all eight choices are possible, only the three signals need be received for someone to accurately decipher the intended message. By taking the base-2 logarithm of the choices (because of the on/off nature of this simple system), $\log_2 8$, we have a measure of the minimum information necessary to accurately identify the intended message, 3 bits, in this case.

Using the same spirit of this information definition, we have developed a quality assessment measure. It is similar to Qualcomp2, however the FCI component is tempered by a logarithmic transformation:

$$Q_3 = k^3 \frac{\log(1+FCI)}{t} \qquad t > 0$$

where

k_3, the normalizing factor, is defined as:

$$k_3 = \frac{1}{\max\{\frac{\log(1+FCI)}{t}\}} = \min\{\frac{t}{\log(1+FCI)}\}$$

Note that this quality computation is undefined for t=0. Since the very structure of the CPP does not allow a design function to be complete at t=0, this time constraint is not of practical concern.

For any non-zero value of t, a design produced with no communication is assigned a Q_3 value equal to zero. This zero communication-zero quality indexing is accomplished through the addition of the integer within the logarithm.

In this quality computation, it does not matter which base is used to compute the logarithm. Since the k_3 constant is assigned a value which maximizes the value of Q_3 to unity, all computations of Q_3 will be equivalent, regardless of the base of the logarithm. One might say that k_3 "includes" the logarithmic equalizer.

This logarithmically derived quality computation is a substantial departure from the Qualcomp1 and Qualcomp2 assessments. By using logarithms, it is possible to discount additional information transfer in a unique and substantial way. In fact, it takes order-of-magnitude increases in information transfer to produce but linear increases in quality. This, of course, is tempered even further by the linear expiration of time.

Another way of interpreting this is as follows: to keep a constant or increasing quality level, communication increases need to rise *by an order of magnitude* for every linear increase of time. For example, a doubling of time requires a squaring of the FCI for the quality level to stay constant. Whether this occurs or not in our "information explosion" today would be an interesting exploration. Results of some studies have indicated that, despite dramatic increases in information processing capability, the effectiveness of many organizations which use these processing mechanisms has actually fallen (ref Brookings study).

The Qualcomp4 quality indicator

An alternative approach to limiting the communicative impact on quality is to accelerate the negative effects of time expenditure. In this Qualcomp4 scenario, quality is inversely proportional to the square of time expenditure:

$$Q_4 = \frac{k^4 \cdot FCI}{(a \cdot t + f)^2}$$

where
k_4 is the normalizing factor to bring $\max\{Q_4\}$ to unity,
a is a constant, known as the *ascension* factor,
f is a constant, known as the *falloff* factor.

As in the other Qualcomp equations, constant k_4 is calculated as the inverse of the maximum, non-tempered equation (computed as if k_4 was equal to one):

$$k_4 = \frac{1}{\max\{\frac{FCI}{(at+f)^2}\}} = \min\{\frac{(at+f)^2}{FCI}\}$$

The ascension factor, a, is introduced here as a constant which affects the rate of rise of Q_4, relative to time expenditure. It may be noticed that such a factor was missing from the Qualcomp2 and Qualcomp3 equations, which also had time expenditure as a factor in their denominators. In fact, ascension factors could have been incorporated in these equations. However, their impact would be nullified by the existence of k_2 and k_3, respectively. In the interests of simplicity and clarity, they have been left out.

The falloff factor, f, was shown to affect the rate at which the Q_4 equation falls, after Q_4 peaks in value. Hence its clever name. Higher values of f resulted in shallower falloff after the quality peak.

The impact of both a and f are illustrated by expanding the denominator of the Q_4 equation:

$$(at+f)^2 \Rightarrow a^2t^2+2atf+f^2$$

For low values of t, it is clear that the denominator is dominated by the value of f. Thus, f is responsible for discounting the FCI when little time has accumulated. This is seen by taking the limit of the denominator as t approaches zero from the positive side:

$$\lim_{t\to 0^+} a^2t^2+2atf+f^2$$
$$= f^2$$

As the time expires, however, its impact is continuously more apparent. Specifically, the a^2t^2 term dominates, leaving f as a mere constant and as a modulator for the growth of the $2atf$ term. As t grows even larger, even the $2atf$ term is small as a proportion of a^2t^2. This is seen as one explores the denominator value as time rises unbounded:

$$\lim_{t\to\infty} a^2t^2+2atf+f^2$$
$$= \lim_{t\to\infty} t^2(a^2+2\frac{af}{t}+\frac{f^2}{t^2})$$
$$= \lim_{t\to\infty} a^2t^2$$

The implication here is clear. Increases in FCI need to be on the order of t^2 to maintain existing, developed quality levels. As pointed out in the description of Qualcomp3,

whether such dramatic communication acceleration occurs or not is very much open to question.

For the dynamic quality analyses of the CPP described in Chapter 6, a was assigned a value of 3 and f was assigned a value of 2000. Other values were experimented with, to understand their impact, but not documented for presentation in this appendix.

The Qualcomp5 quality indicator

To avoid the overly tempting assumption that information transfer is a positive attribute to quality, we consider here an alternative case. We contemplate the thought that productive completion of designs is a positive contributor towards design quality and that information transfer, as measured by the FCI, is detrimental to quality level. This offers some direct impact of design experience on quality. The Qualcomp5 indicator is developed as follows:

$$Q_5 = k_5 \frac{N}{FCI}$$

where

N is the number of designs which have been completed to date,
k_5 is the normalizing factor which limits $\max\{Q_5\}$ to unity. Specifically, k_5 is defined as follows:

$$k_5 = \frac{1}{\max\{\frac{N}{FCI}\}} = \min\{\frac{FCI}{N}\}$$

Notice that time is not directly considered in this quality assessment indicator. Indirectly, expired time is considered, as it is naturally imbedded in the process which enables FCI to increase. Under this scenario, delays between design releases may have no effect on the assessed quality level. If communications occupy design time, however, quality is adversely affected. It is worth reiterating that, with Qualcomp5, quality decreases are attributable to communications, not time. This runs counter to the plethora of popular opinions regarding development time and product life cycles.

Obviously, the most productive quality gains result from maximum design throughput and minimal communications. However, it is still likely that some lower level of communication is necessary for every new design release. Thus, there may be a lower

limit of FCI for every increment of *N*. If so, then there is a maximum rate at which quality may increase, governed by the ratio *N/FCI*.

This measure for quality radiates a "quality with quantity" philosophy for design. Conditions under which this approach is valid need to be examined. However, select design organizations have encountered success according to this philosophy.

Use of Qualcomp Strategies

With the above quality computation strategies established, how could they be used? This, after all, is perhaps the best test of any theoretical device: demonstration of actual applicability.

The following five step strategy is suggested as a basis for using quality measures, such as developed in this study. The basic methodology includes the following iterative steps:

1. Accurately identify and model development activities
2. Assess communication channel capabilities
3. Select, tailor quality assessment strategy
4. Collect Data
5. Monitor/Assess development

1. Modeling: It is both amazing and depressing to observe how little managers know about the operations of their organizations. Just the act of understanding what occurs is a large step for many, many managers. The act of doing this has provided many analysts and progressive managers a basis for simplification and employee education strategies[91]. Such a model can be developed for any organization, regardless of its official hierarchical structure and lines of communication. In fact, functional hierarchies are often misleading indicators of how an appropriate functional model should be structured. Further, it is suggested here that one goes several steps beyond briefly listing what occurs, however, and include analyzing when, how, and why various activities occur, as well as who performs them.

It is important to include the interfacing of information and material (such as drawings or prototype sub-assemblies) between functions in a developed model. Which modeling techniques should be utilized is clearly open for argument, for there is a variety of noteworthy and viable modeling methodologies to choose from. Because of its simplicity of demonstration and its ability to incorporate

[91] Even simple functional modeling could be considered the necessary ancestor to the currently fashionable *activity-based costing* (ABC) and *activity-based cost accounting* methodologies.

information feedback in even very complex systems, the IDEF0 modeling technique is one natural choice. In addition, IDEF0 is currently used throughout industry today, as well as having distinction as the authorized methodology of business process re-engineering (BPR) in the Department of Defense.

Whichever modeling methodology is used, it should be visually simple to interpret, able to accommodate complex and complicated systems, and be able to demonstrate detailed and "high-level" activities and their interfaces. These are important considerations from a methodological standpoint (operations personnel should be able to understand what they are developing and verifying!) as well as an implementation standpoint (management should be able to understand précis versions of the model, so that its structure and the benefits arising out of analysis of its structure are readily apparent).

2. Assess Communication Capabilities: Knowing which functions occur and what they interface is interesting, but it is not enough. An understanding of *how well* various functional interfaces operate is also important. By realizing the limitations and capabilities of communication channels, one gets a better grasp of their potential for impacting the overall enterprise. Capability in this sense includes much more than mere band-widths of information systems. It includes the cognitive, linguistic, and technological ability of functions to communicate with each other. Thus, such factors as personality, political infighting, geography, technological compatibility (employee education?), personnel knowledge/experience, and corporate policies/protocols may be factors in the assessment of channel capability, or effectiveness, *XYeff*. Assuming that every function communicates with every other function, there are n(n-1) channels for which capability assessment need to be made. In those cases where the same individual performs multiple functions, the capability of channels between those functions is expected to be appreciably higher (but not necessarily!). Relative channel capabilities are used to help determine appropriate coefficients within the FCI generator.

3. Select, Tailor Quality Assessment Strategy: Based upon the nature of the product or subassembly being developed, the technological dynamics of the industry, market characteristics/values, and the observed communication characteristics within the organization, an assessment strategy needs to be selected and/or developed. Naturally, this is a judgmental activity which should be contemplated with more than a single individual's experience (recall that the overall objective is to obtain an unbiased measure of design performance).

Based upon the recent history of the organization, most likely at the departmental level, select which of the five strategies best suits the situation. At minimum, appropriate normalizing coefficients for the equations need to be developed.

Specific modifiers, such as *s*, *a*, and *f*, should be assigned based upon recently observed results. If another "new" strategy is found to be more coherent with reality, then that should be used[92].

4. Collect Data: This innocuous title is a gross simplification of a very complicated activity. Specifically, collecting data requires an appropriate data collection mechanism which can accurately reflect the nature and frequency of interfacing between functions. In days past, this would be exceptionally difficult and taxing on the personnel involved in development. With the increase of computer-based communication in and across design organizations, however, automated or semi-automated mechanisms can be developed with little technical difficulty.

In large measure, such a mechanism could be incorporated as a network monitoring device, completely transparent to users. In this mode, the mechanism also serves as a troubleshooting device, where networking problems can be quickly identified and diagnosed. As far as confidentiality is considered, the only information the "system" would need is count and classification of interface type. Specific contents of inter-personal communication need not be monitored.

An important characteristic of such a data collection device is that it operate in real-time, or close to real-time. If delayed by traditional MIS department backlogs, the data will not be recent enough to be useful for managerial decision-making.

5. Monitor/Assess Development: Using the automated communication monitoring mechanism described above, the next step is to assess development progress. The continuously provided data from step 4 offers input into the FCI and subsequently into the specific quality assessment strategies. By reviewing the characteristics of the FCI and resulting quality assessments, a manager can observe, in real-time, the progress of development. If problems are encountered, lags in communication are significantly reduced. Thus, decisions are based on a closer, more up-to-date "picture" of reality, rather than dated, "watered-down" middle management interpretations. The expectation is that the plethora of "emergencies" which occur during development can be reduced through better knowledge among managers of "what to expect."

---

92 As a reality reference point, I would be interested in knowing what alternative strategies are developed--the more minds the better! If you come up with one, or several, please contact me!]]

A warning is in order, however. Communications between any two functions are observed to be highly variable in frequency, content, and noise. Any manager that attempts to use instantaneous communication signals as a measure, without understanding the natural ebbs and flows of communication in development is likely to over-react to short-term changes. The very point of the monitoring mechanisms is to be able to reveal and identify what such natural variation looks like in the organization, and respond accordingly. This methodology is designed to accurately reflect system performance, not create data to be misinterpreted!

Appendix L: Expected Effects of Information Efficiency on DT and TTFR

Improvements in INFO processing efficiency result in diminishing returns. Consider the general case of a system with ***m***-INFO functions and ***n***-MAIN functions to be performed. The time spent processing information, T_{INFO} is as follows:

$$T_{INFO} = \sum_{i=1}^{m}\left(\frac{IR_i}{\varepsilon_i}\right)$$

where

IR_i is the baseline INFORATE for INFO function *i*,

ε_i is the INFO Efficiency for INFO function *i*.

Similarly, the time spent performing MAIN processing, T_{MAIN}, is as follows:

$$T_{MAIN} = \sum_{j=1}^{n}\left(\frac{MR_j}{\eta_j}\right)$$

where

MR_j is the baseline MAINRATE for MAIN function *j*,

η_j is the MAIN Efficiency for MAIN function *j*.

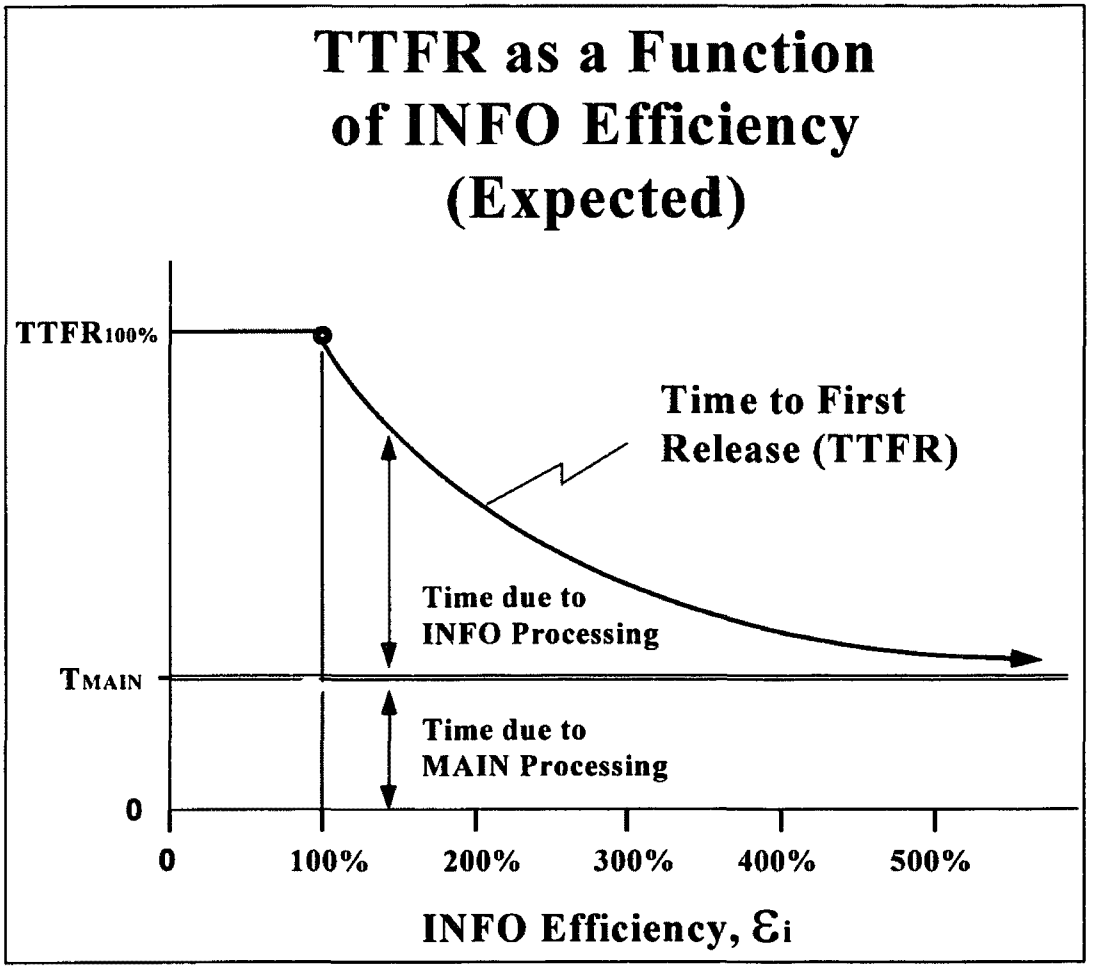
It follows that the total processing time (MAIN and INFO[93]) is the sum of these two:

$$T_{TOTAL} = T_{INFO} + T_{MAIN} = \sum_{i=1}^{m}\left(\frac{IR_i}{\varepsilon_i}\right) + \sum_{j=1}^{n}\left(\frac{MR_j}{\eta_j}\right)$$

Examination of this relationship reveals that improvements in INFO efficiency, ε_i, will continually decrease T_{INFO}, and thus T_{TOTAL}. However, as ε_i increases, the T_{INFO} portion of T_{TOTAL} becomes less and less significant. Eventually, as INFO efficiency approaches extremely high levels, T_{INFO} could approach zero. This would leave T_{MAIN} as the only value of significance. In the limit (as INFO processing efficiency becomes infinite), T_{TOTAL} approaches T_{MAIN}. This is illustrated in Exhibit L.1.

[93] We have not included idle time in this discussion, although such a factor does occur in practice. For completeness, the collection of idle times could be added to the INFO and MAIN times. However, these can severely complicate our simple system, as certain idle times are critical and others are non-critical. The dynamic CPP analysis automatically incorporates these complicated idle time possibilities.

EXHIBIT L.1.



Naturally, information processing efficiency does not rise to infinite levels in practice. Even the remarkable gains in technology since the introduction of the transistor over 30 years ago have not approached such information processing elimination. (It is important to remember that many processing activities have both technological and cognitive components, which can have very different improvement limitations.) Nonetheless, we do expect information processing improvements to affect system performance in the same general manner as just described. Any information processing improvement "should"

improve system performance. Each unit improvement in such efficiency should provide less and less improvement in TTFR, as MAIN processing time occupies a larger share of development time.

Such improvements in TTFR are also expected to be demonstrated in each subsequent design release (recall that this analysis considers multiple design releases, not just the first release). Thus, improvements in INFO efficiency are expected to also shorten the time to the second design release, third release, fourth release, etc. In effect, it is expected that the collection of multiple design cycles should "compress" in time as INFO efficiency is improved. Each release date is expected to follow the same type of performance improvement as illustrated in the TTFR curve (Exhibit L.1.).

Given this expectation about first and subsequent design release times, we can directly derive the expected design throughput of the system.

First, recall that elapsed development time $T_{TOTAL(k)}$ for each design, k, is merely the sum of the MAIN processing $T_{MAIN(k)}$ and the INFO processing time $T_{INFO(k)}$:

$$T_{TOTAL(k)} = T_{MAIN(k)} + T_{INFO(k)}$$

Next, consider that the time necessary to complete **N** designs can be represented[94] by the sum of each of these **N** $T_{TOTAL(k)}$ durations.

$$T_{(1...N)} = \sum_{k=1}^{N} T_{TOTAL\,(k)}$$

Now, given that we are examining design throughput for a given time period, ***D*** (***D***=2500 days in our CPP analysis), we may establish a relation between the release times, duration, and throughput. Simply, we can say that duration is proportional to the product of the number of designs and the average time needed per design. Considering the above definition for duration (***D***= $T_{(1...N)}$), this is just the expansion of $T_{(1...N)}$ using average $T_{TOTAL(k)}$ values:

$$D \propto N \cdot \left[\text{mean}\left\{T_{TOTAL\,(k)}\right\}\right]$$

Rearranging, we see that throughput is related to the ratio of the standard run duration to the average design time:

$$N \propto \frac{D}{\text{mean}\left\{T_{TOTAL\,(k)}\right\}}$$

---

[94] Strictly speaking, the total elapsed time is usually less than this simple summation. This is because of nesting behavior among functions of subsequent designs--the CPP organization often works on more than one design at a time. Thus, a proportionality constant can be placed in front of this summation to represent the ***degree*** of such nesting behavior. A value less than unity would indicate more nesting; a value equal to unity would indicate zero nesting; a proportionality constant greater than unity would indicate negative nesting--a condition where additional idle time exists between sequential development. In a real system, zero, positive, and negative nesting occurs, depending upon the specific time-frame observed.

Obviously, N must be a positive integer, for it does not make sense to have a fraction of a design as throughput. Thus, we perform a greatest integer transformation of this relation:

$$N \propto \text{int}\left[\frac{D}{\text{mean}\left\{T_{TOTAL\,(k)}\right\}}\right]$$

To examine how INFO efficiency affects this relationship, let us expand the denominator of this equation into its components, T_{INFO} and T_{MAIN}:

$$N \propto \text{int}\left[\frac{D}{\text{mean}\left\{T_{MAIN\,(k)} + T_{INFO(k)}\right\}}\right]$$

Holding MAIN processing rates constant, we expand the $T_{INFO(k)}$ term further:

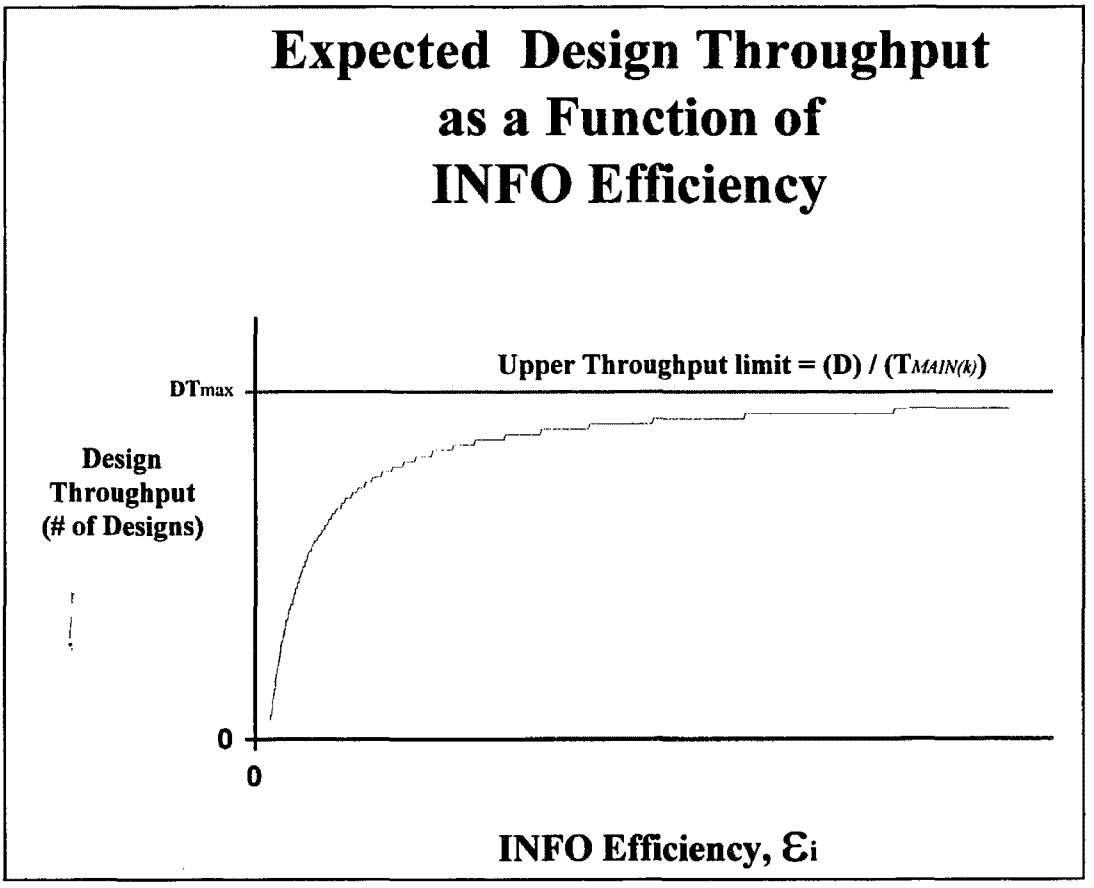
$$N \propto \text{int}\left[\frac{D}{\text{mean}\left\{T_{MAIN\,(k)}\right\} + \text{mean}\left\{T_{INFO(k)}\right\}}\right]$$

$$\Rightarrow N \propto \text{int}\left[\frac{D}{\text{mean}\left\{T_{MAIN\,(k)}\right\} + \text{mean}\left\{\sum_{i=1}^{m} \frac{IR_i}{\varepsilon_i}\right\}}\right]$$

Thus, design throughput can be considered a simple step-function which is anchored on the y-axis at N=0 and which asymptotically approaches the value $\frac{D}{\text{mean}\left\{T_{MAIN\,(k)}\right\}}$.

Refer to Exhibit L.2. below.

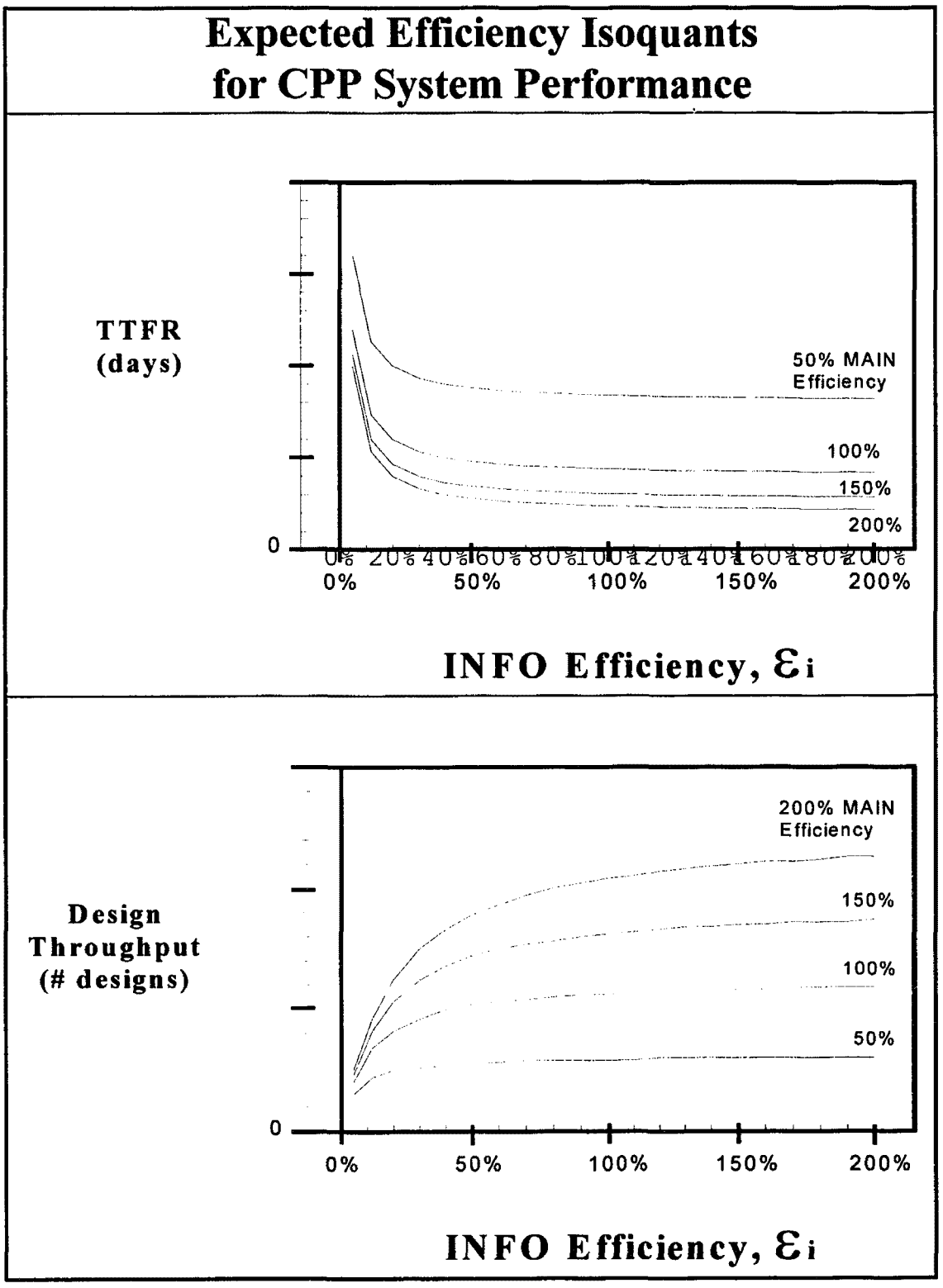
Exhibit L.2.



Naturally, one would expect to see a different step function for each different value of MAIN efficiency. Representative of the mathematical framework just presented, we expect both the TTFR and design throughput values to follow parallel curves as demonstrated by Exhibit K.3. The exact predictions of TTFR and DT are dependent upon the number of INFO activities which are expected to precede each MAIN function. Thus, we have left the y-axis scales of Exhibit L.3. dimensionless.

Exhibit L.3.

Expected Efficiency Isoquants for CPP System Performance



In our simple sequential model, increases in INFO efficiency are expected to improve system performance, although at diminishing rates of improvement. Likewise, MAIN efficiency improvements are expected to be "stackable" with such INFO efficiency improvements. From a mathematical perspective, we expect that INFO and MAIN efficiency effects on system performance are *separable*.

Appendix M: Reconciliation of CPP Structure with Real Development Systems

Model Validation

In this section, we discuss the relevance of the CPP Structure and its behavior to the structure and behavior of actual development systems. In large measure, this is a recap of the processes we undertook in creating the CPP Structure--in reverse. Recall that the CPP Structure was derived from a multitude of field observations, coupled with futile attempts at modeling using established techniques. Notwithstanding the fact that any model must be, by its very definition, a simplification of the real world, there are two important considerations to behold before moving further forward and attempting to *apply* this model to real organizations. First, we consider how well the modeling ***structure*** resembles real development organizations. Second, we consider is whether the ***behavior*** of the CPP Structure is a facsimile of real development organization behavior.

Validation of Model Structure

The basic elements of the CPP Structure are simple: ***stations, information, prototypes, buffers,*** and ***human resources***. To support such fundamental elements, we also define ***channels/communication structures, departments, functions,*** and ***processes***. In addition to these elements, we incorporate specific attributes, which include ***station processing rates, information transfer probabilities/policies, resource allocation, priorities,*** and ***buffer size limits***. How well do these elements and attributes, as we characterize them in the model, correspond to those of real development organizations? Refer to Table M.1.

TABLE M.1.

MODEL MAP TO REALITY
(Comparison of Elements)

Model Element	Definition	Organizational Counterpart
Buffers/ Buffer Sizes	Storage devices for ***information*** or ***prototype*** *waiting* to be processed. May have upper and/or lower limits. We distinguish between ***Information buffers*** and ***Prototype buffers***.	Physical means for storing prototype assemblies and non-engineering documentation.
Channel/ Communication Structure	The path and direction upon which units of ***information*** or ***prototype*** may move within the system.	Any existing communication mechanisms, formal or informal. Examples: Telephone lines, Voice-mail systems, Ethernet, Internet, meetings/conferences, face-to-face, memoranda channels, hallways, etc.
Department	A spatial location for ***stations***. ***Departments*** have a specified number of ***human resources***, some of whom are capable of working at more than one type of station *within* that department. Each person may only work at one station *at a time*, however.	Aggregate place for work activity. Originally organized according to previous assessment of disciplinary commonalty. Examples: Department, Division, Directorate, Branch, etc.
Function	A fundamental activity, conducted in response to the existence and character of inputs from other ***functions***. It performs within technical and logical constraints, as determined by the ***station processing rate***. and ***information transfer policies***. In the CPP Structure, we have two types of functions: ***MAIN functions*** and ***INFO functions***.	A fundamental individual activity, automated or manual. Organizations may have many ***functions***, distributed all over the organizational structure. Examples: Actions, Job functions, Responsibilities
Human Resources	Representations of people who perform ***functions*** at ***stations*** in ***departments***, according to ***resource allocation*** policy.	People who perform activities, usually within some departmental confines. Examples: Engineers Engineering Staff, Administrative Assistants
INFO Functions	Activities which are considered to be "off-line" from classical engineering activities. These functions are dedicated to processing ***information***, not ***prototypes***. To be contrasted from ***MAIN functions***.	Activities which have little or no relation to engineering tasks, but consume the time of engineering-trained individuals. Examples: Administrative activities, Unofficial, non-engineering activities, "Chasing parts"

Information	Any piece of work which is *waiting* for further processing by one of the ***INFO functions***. May be created by either ***MAIN functions*** or INFO functions. To be contrasted with ***prototype***.	Any part, specification, drawing, communiqué, etc., which is transferred from person to person (via any media) for processing, but *not* for engineering-related processing. Examples: Memoranda, Drawings, Documentation, Financial/Accounting Reports, Budget Requests
Information Buffer	A storage facility to hold incoming, ***information***. This may be considered the "in-box" of a department, from which ***human resources*** draw their work.	Depending upon the structure of a given organization, information buffers can carry a number of forms. They may be centralized, departmental buffers, and/or localized buffers at the desk of each individual worker. Examples: Mailboxes, "In-box", "To-do" lists, Desk tops, E-Mail, Voice-Mail, file storage
Information Transfer Probabilities/Policies	The structure of information transmission, in terms of both bundle size (how many information units at a time) *and* expected direction. Since both INFO and MAIN functions can produce information, they are subject to these "policies."	Sometime technically driven, sometimes culturally driven, the tendencies of information transmission, which dictate how much information is created and who gets it.
MAIN Functions	***Functions***, performed at ***stations***, which are specifically dedicated to developing ***prototype***. To be contrasted from ***INFO functions***. MAIN functions may create prototype and information.	Activities which directly relate to new product development. Often called "engineering activities." Examples: Drafting, Testing, Analyses/ Evaluation, Conceptualization/ Specification generation
Priorities	Relative ranking structures for ***information*** and ***prototype*** processing. Dictates what information or prototype should be processed first, based upon the nature of existing work in queue.	Policy, culture, or self-selection mechanisms for deciding "what to do next." Examples: Work schedules, crisis judgements
Process/Processes	A ***process*** is any ordered path of ***functions***. Though it may be pre-designed by managers during model development, the ex post facto process may be the result of the system's "natural" behavior. Such processes may or may not repeat themselves.	A real organization may have many intertwined processes. Rarely is the true structure of such processes explicitly available. Managers develop process representations, which are usually simple, conceptual models of the process. Organizations actually have continuously changing processes.
Prototype Buffer	A holding facility for ***Prototype*** WIP.	A holding facility for prototype assemblies. Examples: Prototype bins ("Banana Boxes," storage shelves, rooms, etc.). Working storage facilities, Design computer files

Prototype/ Prototype WIP	Any piece of work which is waiting for further processing by a ***MAIN function***,	Any part, specification, drawing, file, etc., which is an integral part of the development, from an engineering standpoint. Examples: Physical Prototypes, engineering drawings, customer requirements, interface specifications
Resource Allocation	Logic rules which describe the number of ***human resources*** per department, the number needed for ***MAIN functions***, and the number needed for ***INFO functions***.	Official or unofficial structures by which managers allocate engineers. Examples: Departmental Headcount, Work breakdown structures.
Station	A specific *location*, usually within a ***department***, where a ***function*** takes place. Physical hardware (i.e., tools) necessary for performing the specific function(s) are assumed to be a part of the ***station***.	Locations where engineers actually work. Examples: Engineering Workstations. Drafting Rooms, Offices, Prototype Build Facilities, Pre-Production Build Facilities, Test Facilities, Meeting Rooms
Station Processing Rate	The capability of a ***station*** to perform its ***function***(s), relative to a baseline rate. Thus, 200% means twice as fast as the "baseline."	The duration for any particular activity, usually expressed in units of time per completed activity. Examples: units per week,

In this table, we compare the above structural characteristics to corresponding structures in the real world. It should be apparent that all elements in the model have some real-world counterpart. There are, however, various real-world elements which are *not* included in the CPP Structure. Some major examples include strategists, managers, sellers, customers, end-users, suppliers, regulations, changing requirements, and closed-end schedules. This looks to be an important list. Let us review these and see how they impact the status of our model.

- ***Strategists:*** As we discussed in our field study findings, the perspective of an individual player in new product development may affect their visualization of "the development process", as well as their visualization of how the process "should be working" and how prototypes "should" look. In our field studies, no single individual had greater influence than the strategist. Yet, this role is not explicitly defined in the CPP Structure. The reason for this is that the strategist may be considered the ***creator*** of the system, via his ability to wield budgets and initial development objectives. He may also set the overall managerial tone, by which

management organizes and controls the system. Thus, the non-singular role of the strategist induced us to include his power implicitly, via changes in parameter settings. In this regard, the strategist has both internal and external influence on the CPP structure and its behavior, similar to his influence in real organizations.

- ***Managers***: Because the CPP Structure is established as a deterministic system, there is (in a technical sense) no need for management. The system runs itself, for better or worse. Real systems, of course, do not do this. This apparent gap between CPP and reality is, however, mostly semantic. The manager actually does exist in several forms within the CPP Structure. He may be camouflaged as an engineer (maybe as an information generator!), he may elicit internal influence (via enabling or restricting station capabilities), or he may exert external control (via real-time structural changes to the system, such as removing or adding an information channel). Most often, the manager is not an individual in the system, but rather is the person ***looking at*** the system. In other words, if you are using a CPP model and thinking about how to improve the system, YOU are likely a manager.

- ***External Players***: There can be little question that customers, end-users, sellers, and suppliers can influence the direction and performance of development organizations. For purposes of our analysis, we assumed that such external influences were beyond the control of managers. Thus, they were deemed to be outside the domain of this study. We feel that future development of CPP can and should incorporate such possibilities. Such influences may be considered part of externally-induced requirements changes.

- ***Changing Requirements***: As we have conveyed throughout this report, changes in requirements during the development process can create havoc. There are two important types of requirements changes that need to be considered: internally-induced requirements changes and externally-induced requirements changes.

  Internally-induced requirements changes may be the result of factors such as prototype testing, insufficient internal requirements understanding, stringent ("we have no more time") local schedules, and budget constraints. For the CPP model developed in this study, internally-induced changes are implicit to

the information-transfer probabilities. Since we were interested in demonstration of the flavor of the dynamics which could occur, such probabilities could suffice. In real-world application of the CPP method, such information transfer mechanisms can be modified, according to observations of the organization under study. It is largely because of the uncertainty of such information transfer needs, a priori, that development processes can possess highly unpredictable behavior, as witnessed in the model.

Externally-induced requirements changes are much more difficult to predict, in both the real-world and the model. Example sources of such "external" requirements changes include new government regulations, corporate-based policies, executive orders or preferences, market conditions, and supplier relations. In the CPP model demonstrated in this analysis, we do not address these directly. In a realistic, real-time model, however, we foresee the capability to perform "what-if" scenarios for these types of changes.

- ***Closed-end Schedules***: The CPP model presented in this study demonstrates product development as an open-ended process. In this visualization, development continues for as long as necessary to finish the prototype. In some industries, however, development is considered a closed-end process. This means that there is a fixed time window (often derived from a target production start date) for development to proceed. Provided that this window is smaller than the "natural", open-ended process time, at least one of the following two premises must be satisfied:

  1) conduct activities faster;
  2) conduct fewer activities[95].

  This presumes that the structure of the closed-end processes are similar to open-ended processes (e.g., the same levels of process concurrencies, technologies, product goals, etc.). Given a limited time constraint, an existing open-ended CPP

[95] A reduction in the number of activities may be translated as a reduction in the number of features or level of refinement of the final product. Of course, it is our intent to help reduce the number of non-value-added ("INFO") activities, while keeping the product feature/refinement base as high as possible.

model may be utilized by managers to predict when and where resources can be allocated to speed-up the process, and when such expenses are prodigal, until the closed-end target is reached. Even in the model presented here, the efficiency of each function (station) may be modified in real-time, to reflect such resource allocations. In future evolutions of the CPP methodology, we expect to incorporate real-time virtual activity reduction, so that predictions can be made about the *impact of reducing activities*. Integrated with a CAD/CAE process such as automated geometric modeling, such CPP modeling techniques would provide much better product feature/performance predictions.

Thus, we consider the CPP structure to be an adequate tool for establishing models basic non-linear organizational systems. With more development, it will provide even more detail/resolution. For gaining fundamental insight, however, it can prove useful in its present form. Let us turn to the behavioral characteristics of the model and compare it to the behavior of real development systems.

Validation of Model Performance

It is apparent that the simple CPP model presented here exhibits many performance characteristics which are found in the field. Consider just the following examples:

- Circularity of information transfer;
- Localized efficiency problems;
- Low engineering activity time among engineers;
- High levels of administrative processing;
- Limited human resources in critical development areas;
- Excess human resources in non-critical development areas;
- Irregular work-loads among engineers
- Moving development process bottlenecks;
- Irregular product release times;
- Longer than anticipated development time;
- Higher than expected development costs;
- Pressure to expedite among upstream participants;
- High degree of "waiting-time" for downstream participants;
- Backlogs of administrative work for engineers to perform;
- High priorities of "downstream" activities;
- Development "crystallization".

Such attributes were prevalent among engineers and managers we visited during this study. The CPP model was developed to help explain ***why*** such characteristics are a natural part of the process. In this vein, we are satisfied that the model has accomplished this primary purpose. A CPP model developed for a specific organization is expected to exhibit particular characteristics, many of which are not demonstrated in *this* model. On the other hand, there may be some organizations which have successfully avoided some of the problems established in our simple model. For such companies, more specific CPP models can be expected to show this.

There are two further categories of performance which should be addressed, to help put the capabilities and limits of the CPP model in perspective. First, there are ***performance characteristics of the model which are difficult to accurately document in the field.*** Second, there exist field behaviors which we have not incorporated into the CPP model.

Representing the first category, for instance, the CPP models exhibit some strange parametric response, communication synchronization effects, and time-information based quality predictors. These are extremely difficult to document in the field, but are supported in principle by many participants. We highly advocate further attention to these areas, for they seem to hold some keys to unlocking future improvements in development time, cost, and quality. Let us briefly review these issues.

- ***Strange parametric responses***: As we discussed during our CPP observed effects discussion, intuitive efficiency solutions do not necessarily result in system-wide effectiveness improvements. We saw this occur for *human resource allocations* (more engineers do *not* necessarily increase system speed), *buffer sizes* (large prototype buffers can be detrimental and small information buffers can even *halt* the system), and *processing efficiencies* (isolated "improvements" in processing may sometimes *reduce* system performance). With CPP, we have the convenience of changing such parameters in isolation, a convenience not generally enjoyed by management. Other than through anecdotal evidence, we do not know if (or how often) these effects are manifested in actual development organizations.

  In the cases where such strange effects *seem* to have been observed, there have notoriously been a host of other changes induced simultaneously. Often, such strange effects are observed during transitory modes in real organizations. For our modeling efforts, we held parameters constant throughout each simulation run. Real-time CPP parameter variation is an exciting area for future investigation.

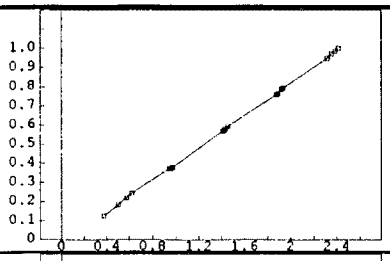
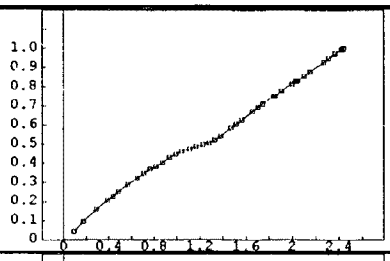
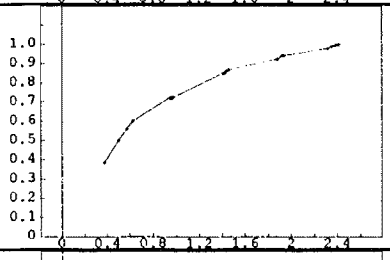
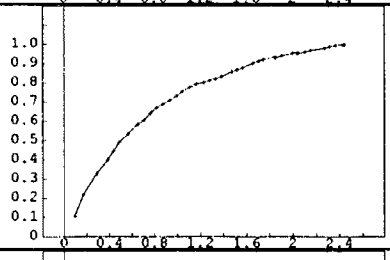
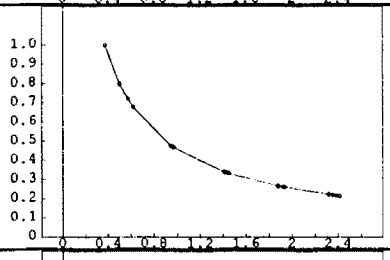
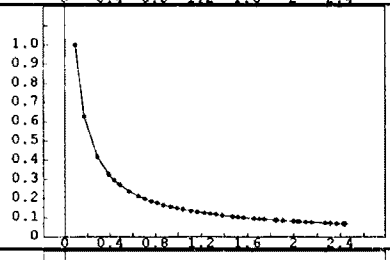
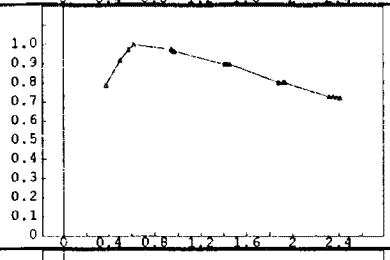
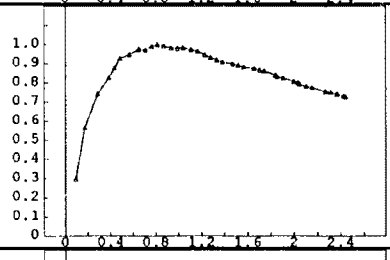
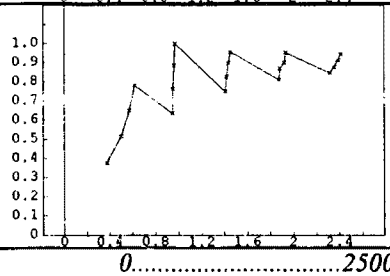
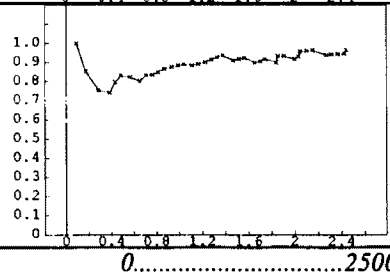
- ***Communication synchronization effects***: Upon visual observation of the CPP Structure in action, it becomes evident that there exist perpetual synchronization problems between any two participants. When participants in one department are ready to process either information or prototype, fellow participants in other departments (i.e., upstream departments) are not ready. The converse of this may be true, as well (i.e., downstream functions are not ready for upstream functions). Such waiting and blocking states of affairs are commonplace throughout the real workplace. However, we have little documented evidence of the *degree* to which

these states occur. At one site, our surveys and interviews revealed that "chasing parts" was the most prevalent single activity among engineers. Thus, an administrative activity was created by engineers to help rectify their own major wait state. If such a problem could be better identified, say through a real-time monitoring model, major wait and block states could be more satisfactorily addressed. In our conservative estimation, such states may occupy up to 50% of engineers' time.

- ***Information-based quality predictors***: As we discussed in Appendix K, we have developed some information-processing based indicators for predicted quality. We developed and utilized five such techniques in our dynamic analysis. Each of these techniques produced different results for a given model. Further, anticipated quality levels could vary widely within a model, even with small deviations in parameters. For example, refer to Exhibit M.1., which demonstrates dynamically changing quality levels for two different runs and the five techniques. As a manager of the development process, which prototype quality profile comes closest to matching your organization? (Data points in the graphs refer to the quality levels for each prototype release. All graphs have been normalized to a maximum level of 1.0.)

EXHIBIT M.1.

5 Quality Indicators

	Run U	Run W
Qualcomp 1		
Qualcomp 2		
Qualcomp 3		
Qualcomp 4		
Qualcomp 5		
Time (x-axis)	*0.....2500*	*0.....2500*

For this initial CPP analysis, quality measures were kept as simple as possible. This was done for several reasons:

> First, there appear to be ***so many different "pet" quality measures*** for so many different products that it is unlikely a specific measure could be generic enough to satisfy any development manager.
>
> Second, evaluation of design quality appears to be ***highly dependent upon the specific situations*** surrounding development personnel. Developing a quality accounting methodology for all possible situations would be unrealistic, tedious, and more than a little pompous.
>
> Third, indicators for ***design quality*** are even ***more difficult to assess than for final product***, as seen by the customer or user. The simultaneously integrative and mis-communicative nature of the design process may behoove one participant to consider a prototype assembly to be over-engineered (having too much quality), while the same assembly may be hopelessly inadequate (low quality) from another participant's perspective. Once assemblies are integrated into a more cohesive design, internal squabbles over detailed design quality may be masked to the end user.

It was observed by several reviewers of and participants in this study that design quality does not necessarily rise with development time. It has been proposed by some that design quality actually rises up to some point in the development process, then falls thereafter. Whether this is a time-dependent or information frequency-dependent phenomenon (or some combination) has yet to be resolved[96].

---

[96] It is interesting to note that similar performance characteristics have been observed in many other disciplines, as diverse as sales, manual labor, even automobile driving (See Evans, 1991).

"Quality" is still a highly subjective area for analysis. Even many quantitative measures are not immune to subjectivity[97]. Thus, the computations presented here are not meant to be robust indicators of design quality for any particular product design. Rather, they are forwarded as potential indicators of the types of quality variation which can be produced during the product development process. Such measures may, in spirit, be applicable to process quality ***at different stages of development*** (e.g., conceptual development. detailed design, prototype build, test/review), as well as ***different subassembly designs*** which exist within a complicated or complex product.

We propose that such diverse measures may occur *simultaneously* among subassemblies of the same product. If true, then this can pose many dilemmas for development managers trying to tune their company's process of development. To better resolve such problems, further detailed research is suggested, with specific orientation towards quality interpretation and dynamics of such interpretation.

Until we posses better understanding of human perceptions of quality, it is advised that development managers gain a solid grasp on *interfacing needs* (i.e., understand how design deviations affect the interfacing assemblies). This interactive approach to design appraisal forces ***higher-frequency communication*** among design personnel and continual ***monitoring of design requirements***, enabling better responsiveness to market wants and evolving technological capabilities.

This brings us into our second major performance validation category: ***field behaviors which do not seem to be incorporated into the CPP Structure.*** Some major candidates for this category include multi-organizational development, prototype looping, non-development responsibilities of engineering organizations, and cultural impacts.

---

[97] It should be well recognized that "subjectivity" and "objectivity" have little or no connection with "qualitative" and "quantitative" analysis methods. In practice, management measures range from highly objective and quantitative to highly subjective and qualitative, with objective-qualitative and subjective-quantitative characteristics seen, as well.

- ***Multi-organizational development***: It is somewhat ironic that the CPP Structure doesn't appear to incorporate the current field condition of decentralized, multi-organizational development, for it was under such conditions that the foundations of the CPP Structure were formed. This apparent inconsistency is easily rectified once one realizes that the CPP Structure is a ***functionally-based*** structure, not organizationally-based. Thus, the locale or formal organizational structure is largely irrelevant to a CPP model. This insight was realized while developing IDEF0 functional models of large-scale development processes in the automobile industry and military organizations.

  To better visualize multi-organizational development using CPP, merely remove the departmental designations[98] in the model and consider stations as functions with no single organizational home. Of course, changes in locale or organization may impact parameter values. How this will affect model performance will depend upon the nature of such parameters. We anticipate that field-developed CPP models will be much larger and more complex than the simple demonstration model in this report. There still remains, however, an acute problem of adequate data collection for detailed activities outside the scope of one's own company.

- ***Prototype looping***: At the end of our discussion on buffer size effects, we alluded to the fact that our CPP model utilizes unidirectional, sequential prototype transfer. In some development organizations we visited, this constraint is too limiting--prototypes may indeed engage in feedback. When such scenarios were observed in the field, development systems could become highly erratic--even more than that seen with just information transfer circulation.

  We may provide for prototype looping behavior in much the same manner as information transfer. In such a case, there will exist ***prototype transfer probabilities*** at each function (hopefully, the dominant likelihood will be towards downstream functions!) and resulting ***prototype processing priorities***. This will offer even more

[98] Alternatively, it can also prove useful to think of "departments" as different organizations, each participating in a large development program.

realistic system dynamics, as prototype "rejection" may occur throughout the process, not just at "testing functions." Such a scenario is expected to make the process even more unpredictable; we can expect to delve far into the realm of Chaos Theory just to describe the behavior we anticipate. As we shall discuss in the next chapter, even our simple CPP model is already taking us there.

- ***Non-developmental responsibilities***: In every development organization visited in our field study, engineers and technicians were asked to perform a variety of tasks which are not directly considered new product development tasks. These types of activities include technical/service manual writing & review, field and plant service support, budget estimations & requests, inventory management, training, creation and dissemination of management reports, supplier negotiations, etc.

  Because our focus in this study has been on new product development, not "running a comprehensive engineering/technical center," we did not explicitly include such tasks. There is a case to be made for the idea that "non-value-added" activities which we characterize as information processing includes such tasks. This accounts for overall time expenditures (in man-hours) of those tasks, but does not address the fact that such tasks may be requested during critical development times. Thus, their schedules may not nest well with development schedules.

  In a strict sense, to avoid a "reductionist" view of this problem, one could create CPP models for each of the major "accessory" processes and merge them with a CPP development model. The "bonds" of such a model merger could be stations (i.e., equipment, facilities, etc.) and/or human resources. Thus, we would consider the performance of a larger model, bounded by capacity constraints. Though this would be appealing from the standpoint of developing a more complete model, there are many more complications and complexities inherent in such an effort, particularly as it grows larger. The tradeoff between such expanded effort[99] and its realizable benefits must be carefully considered.

[99] Recall that model complication levels rise as a function of the square of the number of entities.

- ***Cultural impacts***: As a human-intensive process, new product development is subject to many cultural considerations. During much of our documentation efforts, I had the pleasure of working with a cultural anthropologist. During our work, it was remarkable to see how workplace culture could affect communication channels and patterns, language, equipment/tool use, and decision-making among development participants. I have tried to incorporate some cultural observations into the structure of the CPP model[100]. Nevertheless, there are many such influences which are not documentable by analysts or managers, for they may occur in transient fashions and may be different from participant to participant, and from organization to organization.

  For instance, the mere *suggestion* that development processes may be non-linear, in an environment where managers have been exposed to linear representations and tools for many years, has already been difficult to accept. Acceptance seemed to imply breaking of a well-established, cherished social code. Once such a barrier had been broken, however, further communication was much easier...What was difficult to accept yesterday...is obvious today! Non-acceptance could have the reverse effect: cultural resilience could grow with time.

  Perhaps all we can hope for is that continued, diligent efforts towards investigating the truths about development processes will help reinforce a culture of continuous overall improvement and reduced resistance to new concepts.

[100] Processing priorities, information transfer probabilities, and the basic 3-1 INFO-to-MAIN station relationship were direct reflections of such cultural observations.