Development of a System Dynamics Based Management Flight Simulator for New Product Development

by

Daniel V. MacInnis

M.S. Aerospace Engineering (1986)
University of Maryland

B.S. Aerospace Engineering (1986)
United States Naval Academy

Submitted to the System Design and Management Program
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Engineering and Management

at the
Massachusetts Institute of Technology

February 2004

Signature of Author ______________________________
Daniel V. MacInnis
System Design and Management Program
February 2004

Certified by ______________________________
Nelson P. Repenning
J. Spencer Standish Associate Professor of Management
Thesis Supervisor

Accepted by ______________________________
Thomas J. Allen
Co-Director, LFM/SDM
Howard W. Johnson Professor of Management

Accepted by ______________________________
David Simchi-Levi
Co-Director, LFM/SDM
Professor of Engineering Systems

THIS PAGE INTENTIONALLY LEFT BLANK

Development of a System Dynamics Based Management Flight Simulator for New Product Development

by

Daniel V. MacInnis

Submitted to the System Design and Management Program on January 12, 2004
in Partial Fulfillment of the Requirements for the Degree of Master of Science in Engineering and Management

Abstract

All firms in any mature product development industry are being pressured into performing 'better, faster, and cheaper' by both customers and competitors. In short, firms are being tasked with doing more, with less, faster. This leads to product development organizations being unrealistically tasked to deliver on these programs that often lead to projects falling behind schedule, over budget, and with inadequate quality. While striving to do the right actions to survive, the management of these firms may be leading their firms to disaster through over commitment, and short-term management actions to address the quality, budget, and schedule shortfalls.

An understanding of the system dynamics associated with the program management of new product development (NPD) programs is essential to reversing this trend. Several corporations are instituting system dynamics in their management and executive training curricula to affect *correct* policies, procedures, and behaviors that lead to success. However, because the correct policies, procedures, and behaviors as revealed by system dynamics analysis are counter-intuitive and *opposite* those policies currently employed in program management, a method is needed to drive the *learning* of system dynamics so that it becomes ingrained in the program management thought processes. A management flight simulator (MFS) of the program management of a new product development project based on system dynamics provides the hands on experience that managers can learn the consequences of non systems-thinking policies on project performance and how system dynamics based policies can lead to greater success.

This thesis provides an overview of the system dynamics of project management in new product development and insight into the correct policies, procedures, and behaviors that lead to success. Research on the role of MFSs in driving the learning of system dynamics principles is explored. A single-phase system dynamics model for a new product development program and a MFS is developed to teach the fundamental lessons of system dynamics applied to product development project management and is to be incorporated in the BP Project Academy. Insight from my own experiences in product development is incorporated in this MFS as well as in recommendations for further development.

Thesis Supervisor: Professor Nelson Repenning

THIS PAGE INTENTIONALLY LEFT BLANK

Acknowledgements

I would like to thank my thesis advisor, Nelson Repenning, for his tremendous support in this thesis, help in developing the system dynamics model and management flight simulator, and expertise on project dynamics. It was his lecture on 'Tipping' and fire-fighting in new product development that motivated me to do a thesis on system dynamics in NPD.

I would also like to thank Nick McKenna, a fellow SDM student who with Nelson developed the system dynamics model for which this MFS is based on. I would also like to thank Scott Johnson of the BP Project Academy who provided the funding for the development of this MFS and for his guidance on the requirements of the MFS for incorporation into the BP Project Academy training curriculum. Many thanks also goes to the MIT Sloan School Management Science faculty and PhD candidates who 'flight tested' the MFS and provided valuable feedback for refinement of the MFS.

I also would like to thank James Lyneis, John Sterman, Brad Morrison, Jay Forrester, and Peter Senge who through my SDM studies introduced me to the exciting field of System Dynamics. I now and forever will see things in a different light and as John Sterman told me, it is clear that I am 'looped for life.' I plan to continue my involvement in System Dynamics at my new position at Raytheon Missile Systems and through the System Dynamics Society.

Finally, my deepest thanks go to my wife, JoAnne and my daughter Danielle who provided love, support, and understanding throughout my studies at MIT

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

Chapter **Page**

THIS PAGE INTENTIONALLY LEFT BLANK

List of Figures

Figure **Page**

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

1.1 Background and Motivation

I believe there is a crisis in new product development (NPD) today. While the problems of projects being late to schedule, over budget, and with inferior performance and quality are not new, with the growing complexity of new products and the increasing pressure to deliver new products "better, faster, cheaper" these problems are accelerating, and thereby degenerating the ability of companies to remain competitive in their respective industries. Indeed, these problems and the resultant late to schedule, over budget, and inferior products are becoming the norm. In addition to the undesirable outcomes for the immediate stakeholders, i.e., the customer and the shareholders of the inadequately performing corporations; these problems wreak havoc on the employees who often must work excessively long and stress-filled hours in 'fire-fighting' mode to deliver as close as possible to the often unrealistic schedule, budget and performance targets. The resultant burn-out and suffering morale leads to employee turnover: a loss of knowledge capital making the company even less competitive.

Why are these problems so ubiquitous in new product development environments? Despite the technological advances in product development including computer aided design (CAD), improved communication (internet, e-mail), etc. and advances in project management techniques including PERT, Critical Path Method (CPM), teaming and concurrent engineering, informational technology (IT) based resource loading tools, etc. the problems have gotten worse (Reichfelt and Lyneis, 1999). An understanding of the system dynamics of projects in new product development is essential to explaining these problems and changing policies, procedures and methodologies to prevent and/or overcome them.

I have been involved in several new product development projects for three major aerospace companies for the last ten years. I have witnessed first hand the dynamics of projects unfold as portrayed in the project management system dynamics literature to be covered in chapter three. What is interesting to me is that the policies, procedures, and decisions that are necessary to correct the undesirable behaviors (late to budget and schedule, inadequate performance and quality) are *counter intuitive.* Indeed, although I had been exposed to a comprehensive module on System Dynamics in the System and Project Management class at MIT, I *still* reverted to short-term "better-before-worse" solutions to problems, knowing full well that they were *the wrong thing to do!* Showing senior management Power Point presentations of the system dynamics module and showing that purposely slipping the schedule would allow us to eventually meet the deadline was *not* effective in changing the mental models that have been engrained for decades.

The power of system dynamics as applied to project management carries a double edged sword. While it can answer the question "Why do we continue to fall short", the same dynamics drive the reluctance to adopt its principles as the 'system' strongly reinforces the wrong behaviors (rewarding heroic fire-fighting, etc.) that further exacerbate the long term problems. Because the correct behaviors by management are counter intuitive, there must be a better way to instill the *learning* of these system dynamic principles. Methods such as "management flight simulators" (MFS) have been developed to instill learning of system dynamics principles in other domains such as in supply chain management with "The Beer Game" (Sterman, 1992) and business policy decision making in growing companies with the "People Express Management

Flight Simulator" (Sterman, 1988) have been effective in changing the mental models in those domains.

As mentioned in the literature supporting the latter MFS and in in-class discussions, pilots are not set loose to fly aircraft, nor are physicians allowed to operate on people until they are extensively trained with flight simulators or given repeated practice opportunities on cadavers and medical simulators respectively (Sterman, 2003). Managers of complex new product development projects should have similar simulators to learn the right way of managing to avoid the problems of being late to schedule, over budget, and with inadequate performance and quality. This learning method offers a risk-free learning environment where managers can manage their projects and see the consequences and advantages of different management strategies and decisions.

BP p.l.c. has incorporated a management training curriculum that teaches system dynamics principles for product development called BP Project Academy. The current curriculum includes Power Point presentations and playing of "The Beer Game" by the participating managers. However, the 'power point slide only' approach has been shown to be ineffective for teaching the principles adequately enough to change the managers' mental models and change their policy, procedure, and decision making behaviors (Repenning, 2003). Therefore, BP has funded the development of a MFS to include in their Project Academy training curriculum. Thus, the motivation for this thesis is to improve the way NPD projects are managed by enabling a hands-on learning tool that can drive the system dynamics based management philosophies into the managers' mental models, thereby influencing them to make the right policies, procedures, and decisions.

1.2 Objectives

Due to the impact of management decisions on customers, shareholders, and employees in new product development domains, a MFS should be developed for managers to learn the right way to manage complex product development projects before being 'set loose' to do what comes naturally to them. As the new product development project management domain involves such complex systems, the field of system dynamics is essential to understanding the patterns of behavior prevalent in that domain. Therefore, the management flight simulator (MFS) for NPD must be based on system dynamics and the corresponding causal feedback loops, stock and flow structures, and information flow time delays associated with new product development projects. The purpose of this thesis is to develop a system dynamics based MFS for new product development.

As this is the first MFS on project management for new product development to be incorporated at BP, it is kept relatively simple to not overwhelm the management participants with too much detail. Therefore, the MFS is based on a single-project, single phase system dynamics model of a NPD project incorporating the minimum features required to drive the most fundamental lessons into the project managers' mental models.

The first objective is to learn the system dynamics concepts and principles that govern successful NPD projects. The second objective is to incorporate these concepts and principles into a system dynamics model that accurately portrays the patterns of behavior given the policies, procedures, and decisions set forth in the model. The third objective is to develop a MFS based on this system dynamics model that allows the players to input hiring/firing, schedule, and overtime policies and decisions to learn the consequences of poor policies and the advantages of proper policies and decisions in NPD project management. This MFS should be tailored to the needs of the BP Project Academy, i.e., be associated with developing a new drill site, incorporate a scenario and management metrics that the participating BP managers are familiar with, etc. The MFS should teach the following lessons obtained from a system dynamic perspective on the management of projects for new product development:

- **Managers must have a realistic and attainable budget and schedule for the projects they manage.**

- **Management actions that are intuitively taken to manage new product development projects may show beneficial results in the short term but will yield undesirable long term results.**

- **Proper management actions for new product development projects often yield a 'worse-before-better' result.** Managers must learn that the *right* policy, procedure, or decision may in the short-term look worse than if they had taken the intuitive 'better-before-worse' course of action.

- **It is better to "get it right the first time" and be late than to be on time and have it wrong.**

The final objective is to incorporate lessons learned and experiences from my own NPD project management background in this MFS and make recommendations for further development of the new product development MFS. As I have seen the dynamics first hand that drive the wrong management actions, I have been able to incorporate realistic features in the MFS such as the inclusion of Earned Value Management System (EVMS) metrics. While not part of the BP product development management system, it is prevalent in several large industries (it is mandated for Department of Defense contracts) and is being adopted for many commercial product development projects as well. Recommendations are also made for further development based on past project management experience, i.e., incorporation of schedule pressure through senior management reviews, self-imposed schedule pressure by managers influenced to 'get back on schedule' quicker, and the shifting of personnel from other programs to support fire-fighting to get existing project back on schedule.

1.3. Approach

This section details the approach taken to meet the objectives including the methodology and the structure of the thesis.

1.3.1 Methodology

A literature review was conducted on the system dynamics of projects in the NPD domain. This research included an assessment of the state of project management for NPD and characterizes the primary problems of late to schedule, over budget, and inadequate performance and quality. The system dynamic principles were identified that capture the system causal feedback and stock-flow structure, information flows and time delays that drive poor schedule, budget, and quality performance. The literature review also covered the relevant attributes of system dynamic based simulations that are necessary to drive effective learning of system dynamic principles in organizations.

A system dynamics model was then developed incorporating the causal feedback, stock-flow structure, and information time delays relevant to new product development projects found in the literature review. This model was developed in conjunction with a parallel research effort by Nick McKenna also funded by BP and incorporates elements specific to the petroleum industry. This system dynamics model is a single project, single phase model that incorporates internal process concurrence relationships (Ford and Sterman, 1998). The model is not calibrated, but is a simplified version of a more complicated model that has been calibrated and has been the basis for BP's system dynamic training and management science efforts (Johnson, 2003). The system dynamics model was developed using VENSIM 5.2a, a system dynamics software package made by Ventana Systems Inc.

A management flight simulator was then developed using VENAPP, a simulation tool incorporated in VENSIM that allows user interaction with a system dynamics model similar to that with a game. The MFS was modeled after a previous VENAPP developed by Nelson Repenning and John Sterman for quality improvement (Repenning, Sterman and Leach, 1995). The MFS was developed using product development methodologies taught in MIT System Design and Management (SDM) courses Systems Engineering, System Architecture, and Product Design and Development. The MFS and system dynamics model were then tested and refined based on feedback from BP and MIT Sloan Management Science faculty and PhD candidates.

The MFS was run with different project conditions to show the appropriate behavior patterns associated with different management strategies and decisions to ensure that the objectives set forth in 1.2 were met. Experiences from NPD projects I have managed in the past were related to the MFS results and used to provide additional recommendations for further new product development system dynamic model and MFS development.

1.3.2 Structure of Thesis

This thesis contains the setting in Chapter 2. This chapter includes a brief overview of the field of System Dynamics and how system dynamics is applied to complex systems and in particular to project management of new product development. The NPD process and the BP Project Academy training curriculum are also described. A literature review is presented in Chapter 3 and includes an assessment of the state of NPD project management today, including a description of the problems of NPD projects being late to schedule, over budget, and with inadequate performance to specifications. Explanations for these problems as explained by system dynamics are provided. The "Rework Cycle" is explained as the basis for most system dynamic modeling in new product development project management domains (Reichfelt and Lyneis, 1999). The 'Ford-Sterman Model' is then explained as an improvement to the rework cycle and is the basis for the system dynamics model for which the MFS for this thesis was developed (Ford and Sterman, 1998). Latest theories of new product development dynamics such as Nelson Repenning's "Tipping" and fire-fighting in new product development organizations are also explained. A section on the role of MFSs in improving the learning of system dynamics principles in organizations is also presented in Chapter 3.

Chapter 4 details the methods used to meet the objectives of this thesis. This chapter includes a description of the system dynamics model that the MFS is based on. The chapter also includes the steps taken to develop the MFS including the product development process used to meet the requirements of the MFS. A description of the MFS is also included in this chapter. Chapter 5 presents the results of this MFS including results of test simulations run by MIT Sloan faculty and PhD candidates. These results and lessons learned from development of the MFS are discussed in Chapter 6. Suggestions for future work based on the literature and my own new NPD project management experience is provided in Chapter 7. Conclusions and recommendations are provided in Chapter 8 and references used are provided in Chapter 9. The system dynamics model documentation is provided in Appendix A and the documentation for the MFS is provided in Appendix B.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 2

Setting

This chapter details the relevant background information that is required to understand the work that this thesis is based on. It begins with a brief overview of the field of System Dynamics, and is followed by a description of the NPD process and background information on the BP Project Academy training curriculum.

2.1 System Dynamics[1]

The field of System Dynamics was founded by Jay W. Forrester at the MIT Sloan School of Business and was introduced to the mainstream academic literature in the book "Industrial Dynamics" (Forrester, 1961). It evolved from the application of control theory from electrical engineering to the behavior of non-technical *dynamic* social systems (supply chain distribution, business management, etc.) Its premise is that the behavior of a dynamic system is the result of the *structure* (causal relationships, feedback relationships, and time delays), not necessarily the cause-effect actions of individual parts in a system. System Dynamics is particularly well suited for the understanding of complex systems where multiple feedback effects, time delays, and unknown system property attributes are unseen. Because of our inability to manage complexity, a systematic process is required for us to understand the complex relationships that drive complex system behavior. It was for this need that the field of System Dynamics was developed.

[1] The information outlined in this section was obtained primarily from material learned in ESD.36J System and Project Management and 15.874 Business Dynamics at MIT in 2002 and 2003 respectively.

There are several concepts and tools used in the field of system dynamics. The central concept is the concept of "systems thinking". Peter Senge popularized the concept of "Systems Thinking" based on Forester's work in his seminal book "The Fifth Discipline: The Art and Science of the Learning Organization" (Senge, 1990) in which he defined systems thinking (the "fifth" and most important of the five disciplines described in the book) as a framework for understanding, through a body of knowledge and tools, that the behavior of a system is driven by the relationships and interactions of all parts of a system, not any individual part of the system (Senge, 1990, p. 7). Another concept central to system dynamics is that all decisions are made based on models. A corollary to this concept is that all models are wrong (Sterman, 2003). A model is any representation of reality, whether it is a physical prototype of a future product or a *mental model* which is a representation of reality in someone's mind. Since all models are wrong (some more "wrong" than others), "bad" decisions are made due to incorrect mental models. Thus, good, well intentioned people doing what they think is right based on incorrect mental models may be leading the system to failure despite their well intentions. The field of system dynamics is therefore a framework for improving mental models about complex systems to enact change that will improve the behavior of the system.

Another concept central to system dynamics is that of *causal relationship*. Assume a system is made up of a number of components that interact to determine system behavior. An increase in value of one component of a system will enact on another component value either a likewise increase (a *positive* causal relationship), or it will enact a decrease (a *negative* relationship). How these causal relationships relate back to the behavior of the original component is represented by the concept *causal feedback loops*. Therefore if the increase in value of the original component of a system described above leads to a *further* increase in that component as the causal relationships are traced around the system, then the causal feedback *loop* is called a *reinforcing* (or positive) causal loop. If instead the component value decreases when the causal relationships are traced back to the original component in the system, the causal feedback is called a *balancing* (or negative) causal loop. *Causal Loop Diagramming* (CLD) is a tool used in System Dynamics to trace the causal relationships in a system and determine the primary causal loops of a system. An example of a CLD, adapted from "Business Dynamics: Systems Thinking and Modeling" (Sterman, 2000) which shows the causal feedback loops for elements of production system is shown in Figure 1. Arrows are marked with a "+" sign to indicate a positive causal relationship, and a "-"sign to indicate a negative causal relationship. Loops are marked with either a "R" to reflect a reinforcing (positive) feedback loop or a "B" to mark a balancing (negative) feedback loop. The loops are named appropriately to help the system dynamicist recognize the meaning and behavior attributes of that loop on the system. The double slash marks on some of the arrows indicate time delays for those causal relationships. The CLD in Figure 1 is very similar to the representation of the model that drives undesirable behavior in product development organizations. Causal loops are important in system dynamics because they predict the behavior of the system. Typical behavior patterns prevalent in systems throughout nature are associated with combinations of negative and positive causal loops. A thorough explanation of all these behavior patterns is beyond the scope of this thesis.

Another tool used in system dynamics is the *stock-flow structure*. This modeling concept incorporates the accumulation of *stuff* (whether a physical quantity like products in a warehouse, or an intangible quantity like the perception of lateness) which is modeled as a *stock*. The flow of stuff that either accumulates (feeds) or depletes a stock measured as quantity per unit time is called a flow. Stock-flow structures are used with CLD to model the *dynamic* nature of system behavior by incorporating time into the model. An example of a stock-flow structure that models the adoption of a new product is shown in Figure 2.

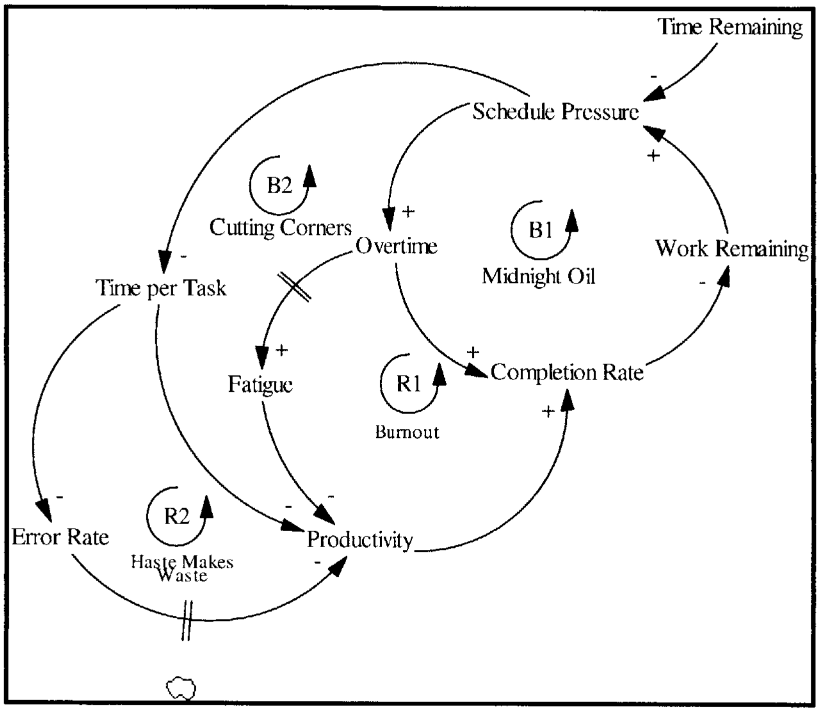

Figure 1. Causal Loop Diagram (CLD) of Production System, adapted from (Sterman, 2000, p. 149)

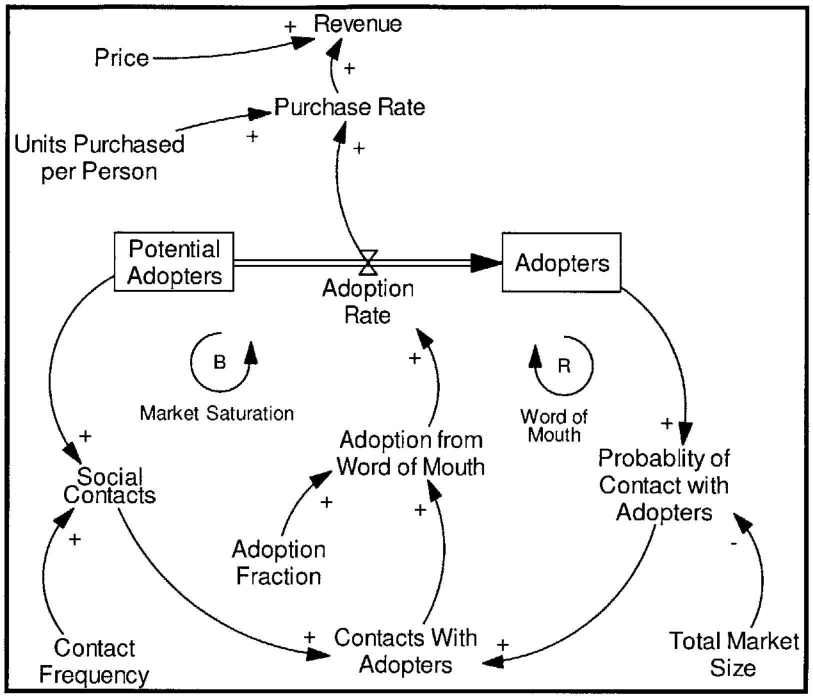

Figure 2. Stock Flow Structure of New Product Adoption

The behavior pattern of this system is typical of a two stock structure that incorporates a reinforcing and balancing causal loops. Initially, product adoption is slow but rises exponentially as the reinforcing loop R of "word of mouth" (WOM) dominates the system behavior. As the stock of *Potential Adopters* is depleted, however, the number of *social contacts* is limited and the balancing loop "market saturation" dominates the system. The overall system behavior, or "reference mode" (another concept in system dynamics where the behavior of a parameter over time is predicted and/or evaluated to guess which causal relationship is dominating the system) of potential adopters is S-shaped, as it is limited by the number of potential adopters that feeds the adoption rate.

Through systems thinking, CLD, and stock-flow structure, the system dynamicist can identify the causal loops that ultimately determine system behavior. They can identify the loops that drive undesirable system behavior and enact policies to weaken those loops while strengthening the loops that drive desirable system behavior. The final tool used by system dynamics to be discussed in this thesis is simulation. While systems thinking, CLD, and stock-flow structure models are helpful in understanding the dynamics of complex systems, the most successful applications of system dynamics to changing the mental models of key enablers that can impact system structure and performance (i.e., senior management) are where detailed, *calibrated* models were developed that simulate real-world performance (Lyneis, 2003). While the level of detail required is debatable, the more accurate the model, the more likely management will believe the reasoning behind the system behavior caused by system dynamic principles and enact change in their policies. This is important because most of the time the correct policies are counter-intuitive: the impacts of undesirable loop effects from existing policies are not seen until long after they are initiated due to time delays.

2.2 Product Development Process[2]

Since the MFS is to model the new product development (NPD) process, a short description of the NPD process is described in this section. The NPD process occurs in a complex system combining multiple disciplines and multiple phases to develop a product to meet a customer's functional requirements. The NPD process is broken down into different stages called "phases" that transform customer functional requirements into finished product. While NPD processes vary slightly from organization to organization, the typical NPD process incorporates the following sequential phases (Ulrich and Eppinger, 2000, p. 9)

- **Phase 0: Planning Phase:** This phase includes corporate strategy and assessment of technology developments and market objectives. Key output of this phase is a project

[2] The information for this section was obtained primarily from (Ulrich and Eppinger, 2000).

mission statement that specifies the market targeted, key assumptions, and constraints.

- **Phase 1: Concept Development Phase:** The needs of the target market are identified and concepts are developed, evaluated, and selected for further development. Outputs from this phase include a description of the form, function, and features of potential products to meet target market needs, a competitor product assessment, and an economic justification to go forward with the product.

- **Phase 2: System-Level Design:** This is where decomposition of product functionality and architecture to subsystems is made to manage complexity. The end product is a layout of the entire product assembly with functional requirement definition (specifications) for each of the subsystems and integration of these subsystem designs to ensure product functionality.

- **Phase 3: Detail Design:** In this phase, functional specifications identified in Phase 2 are transformed into detailed designs including definition of geometrical form, material, and tolerances for all components making up each of the subsystems. This output definition make up the *control documentation*: the drawings and/or computer files describing these parts and the associated production tooling required to manufacture them.

- **Phase 4: Testing and Refinement:** This phase evaluates the performance of pre-production prototypes to ensure that the functionality of the product meets the customer's requirements. Results from these tests are used to refine or iterate the design process(es) in Phase 2 and Phase 3.

- **Phase 5: Production Ramp-Up:** Once a product meeting the customer and economic requirements is developed through phases 1 to 4 has been finalized, the product is developed to be manufactured to the production processes. In this phase, the *production* work force (as opposed to development work force) is trained and any problems in production processes are worked out.

There are several factors about the NPD process that make it hard to manage successfully. First, several disciplines interact (internally and externally) as the project progresses through the 5 phases. These disciplines include marketing, design, manufacturing, finance, sales, drafting, procurement, and quality. Additionally, the outputs of each of the respective *upstream* stages have a profound impact on the *downstream* phases. For example, the more accurate (i.e., the higher quality) that the outputs in Phases 2 and 3 have, the less iteration results from Phase 4. Phases 0 and 1 are typically referred to as the "front end" of the NPD process, and have the highest impact on the individual project and the entire NPD organization's success (to be shown in later section in Chapter 3). However, because of the time delay associated with actions in those phases not being discovered until the later stages (as long as 2 years), managers do not associate the effects with those actions. Finally, because of the inherent complexity of managing so many different processes, disciplines, and interrelationship dependencies (i.e., 'concurrence'), managers tend to simplify their mental models of the process to deal with the complexity. For example, they view the processes as purely sequential, not recognizing the importance of dependencies, and under-estimating the amount of iteration that will be required.

2.3 BP Project Academy[3]

This section provides a brief description of the BP Project Academy. The BP Project Academy was established in conjunction with the BP Engineering Project Technology Group (EPGT) Project Excellence Team (PET) to help improve BP project perform by addressing project dynamics. It incorporates a 3 Term training curriculum to teach system dynamic principles to managers as shown in Figure 3. The objectives are to instill systems thinking, teach how CLD can be used to reframe project issues, provide a safe environment through the use of a MFS to deepen their knowledge and skill, and explain how participants can use the EPGT PET system dynamics resources when they return to their projects. Each term is taught by an MIT facilitator (Nelson Repenning) in 2 day sessions. The MFS is being developed for Terms 2 and 3.

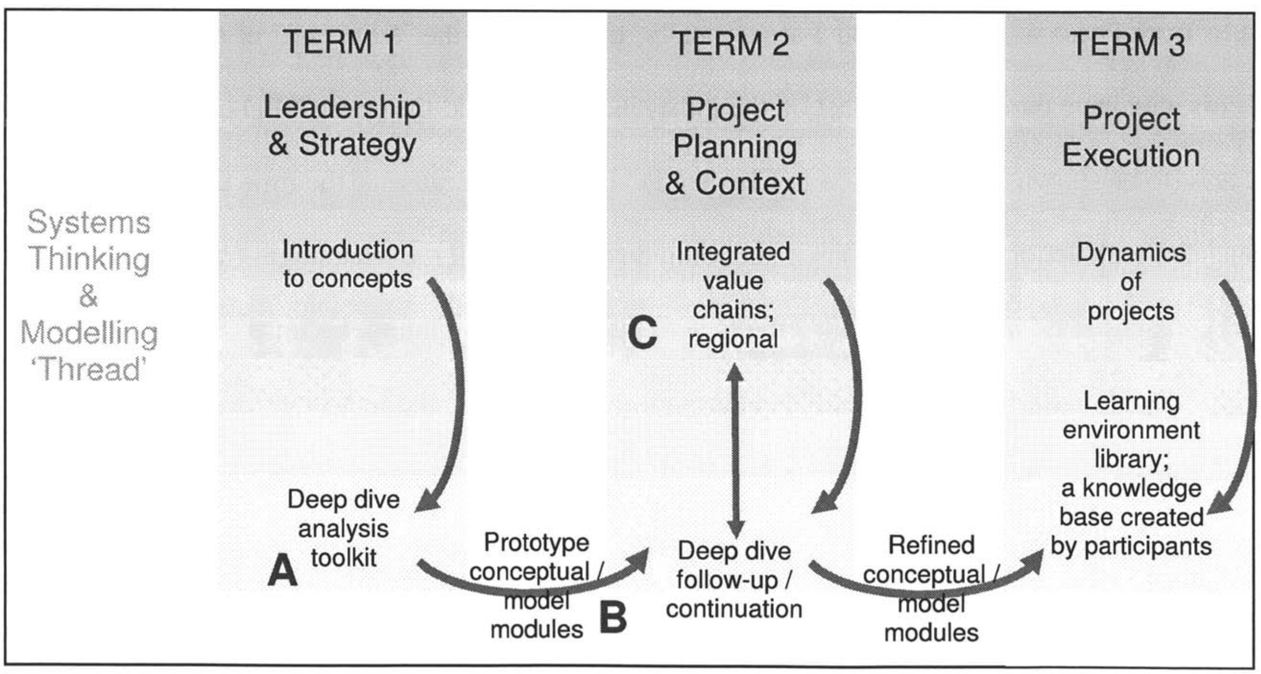

Figure 3. BP Project Academy Plan, taken from (Johnson, 2003)

[3] Information for this section was obtained from (Johnson, 2003)

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 3

Literature Review

This chapter details the relevant research on system dynamics as applied to new product development project management. It begins with a description of the problems prevalent in new product development projects. The "Rework Cycle" from which almost all new product development system dynamic modeling is based on (Reichfelt and Lyneis, 1999) is explained and is followed by descriptions of later developments in the field. A section is also included on the role of simulation in improving the learning of system dynamics in organizations.

3.1 System Dynamics and New Product Development

This section begins with a description of the problems found in NPD projects and why system dynamics is an appropriate method for understanding why these problems exist. Several modeling concepts are discussed including the latest theories of project dynamics affecting performance of NPD new product development organizations.

3.1.1 Problems of New Product Development Projects

Anyone who has been involved with managing new product development projects has experienced the problems of keeping their projects on schedule, on budget, and with the required performance attributes required by the customer. These problems are also well documented in the literature. A study of 3500 new product development projects by Morris and Hough in 1987 found that schedule overruns were typical, normally between 40 and 200 percent (Reichfelt and Lyneis, 1999) and a similar study by Roberts in 1992 found that less than half of corporate R & D projects were able to achieve their schedule and budget goals (Lyneis, Cooper and Els, 2001).

A sample of 10 large projects used for a system dynamic assessment of new product development performance by Pugh-Roberts Consultants, the premiere system dynamics consulting group now of PA Consulting in 1999 found that of the 10 projects, average budget overruns were 86% and the average schedule overruns were 55% (Reichfelt and Lyneis, 1999). An additional study of World Bank Projects in 1992 found that only 70% of projects were rated as 'satisfactory' by stakeholders and only one third achieved their goals, with completion delays averaging 50% (Lyneis, Cooper and Els, 2001).

The problems of projects completing late to schedule, over budget, and with inferior quality and performance are betting worse. One reason to explain this is the change in customer expectations in mature industries. When in a new industry, customer expectations are primarily set by performance and improvement in the performance features that define the functional value of the product. As the industry matures, however, customer expectations shift to reducing cost (Utterback, 1994) *and* delivering newer products faster in addition to meeting the performance improvements. Thus, the mantra "Better, Faster, Cheaper" dominates and epitomizes the customer expectations in a mature industry. In addition to impeding the ability of these firms to attain these objectives (since the expectations have increased), the mental models of managers have not changed to accommodate these changed expectations and thus their policy, procedure, and decision making practices that were previously successful in a 'better only' environment are ineffective in the new 'better, faster, cheaper' paradigm.

The situation gets even worse, as Al McQuarrie describes in his 2003 MIT SDM thesis, because for firms to remain competitive in their respective industry, they must aggressively bid even more 'better, faster, and cheaper' than competitors (even in excess of customer expectations) to win new business to stay in business (McQuarrie, 2003). The following causal loop diagram in Figure 4., taken from McQuarrie's thesis, illustrates how these undesirable dynamics form to make the problems of new product development projects get worse over time in a 'better, faster, cheaper' environment which is indicative of the current Department of Defense's (DoD) customer expectation environment:

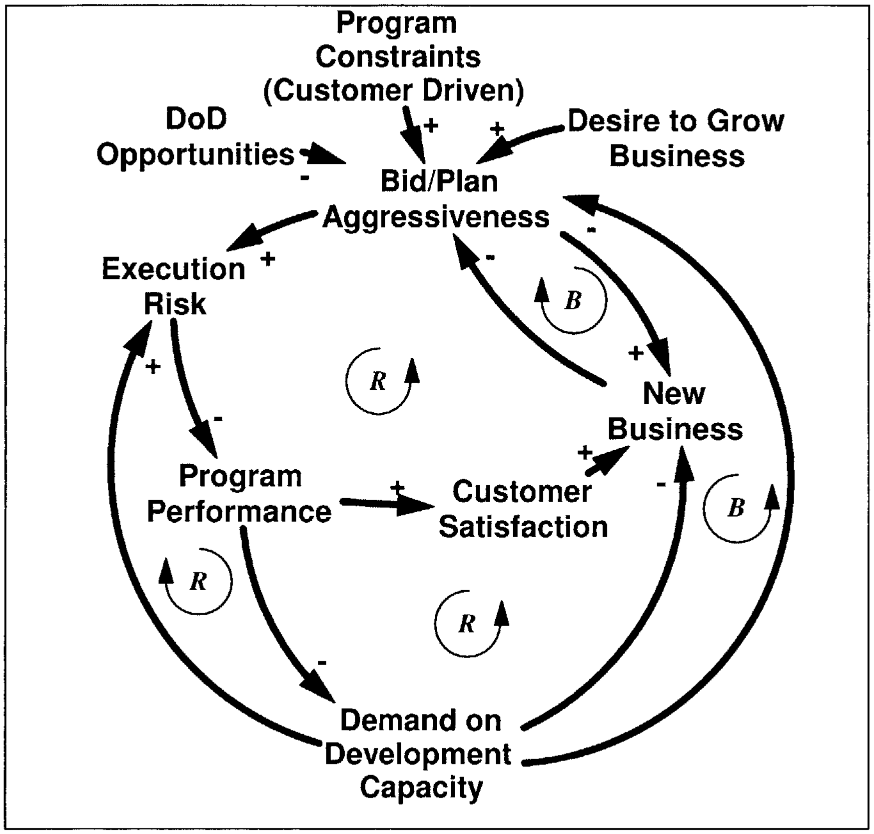

Figure 4. Business Capture Causal Loop Diagram – Relationship Between Business Goals, Bid/Planning, Resource Demands, and Customer Satisfaction (taken from Al McQuarrie 2003 MIT SDM Thesis).

As Figure 4 illustrates, as customer expectations increase and/or desire to grow business increases, bidding aggressiveness increases which in turn increases program execution risk, reduces program performance and customer satisfaction, and thus reduces new business. The reduction in new business drives the company to bid even more aggressively, further perpetuating the vicious cycle of poor performance and customer dissatisfaction. The other two reinforcing (positive) causal loops further contribute to the undesirable project dynamics. As program performance is reduced, demand on the organization's product development capacity is reduced through the dedication of additional resources to address the faltering program, which further reduces the ability to support new business. Additionally, as the demand on development capacity increases, program execution risk is also increased further reinforcing the undesirable dynamics. The two balancing loops work to counter these undesirable dynamics, but because they are counter to current management mental models, i.e., reduce bid aggressiveness; they usually do not dominate to reverse the vicious cycles to virtuous cycles (McQuarrie, 2003). In summary, as a NPD industry matures, the ability of a company to achieve the 'better, faster, cheaper' expectations decreases, and is further exacerbated by an organization's actions to bid more aggressively in the maturing market to stay in business.

Another reason why the problems of being late, over-budget, and with inadequate performance and quality are getting worse is because of the increased complexity of product development projects (Ford and Sterman, 1998). With the increase in product development project complexity, organizations have shifted from sequential and functional product development processes and organizations to team-based and concurrent engineering PD processes and organizations to manage the complexity and meet the 'better, faster, cheaper' expectations (these PD methods were introduced to reduce staffing levels in support of the 'cheaper' criteria as well as improve the communication and understanding required of more complex products). These newer PD processes and organizations dramatically increase the dynamic complexity of the new product development project, and like that seen with the change in customer expectations, the mental models of managers leading these product development projects have not changed to accommodate this increased complexity. Thus the estimates of manpower and budget required and decisions made to manage these projects have not taken into account the increased complexity and system dynamics of these projects. Thus, poor decision making based on lack of understanding of system dynamics of complex product development has contributed to even more project failures (Ford and Sterman, 1998). As complexity increases, the problems of new product development projects get worse.

The problems of NPD projects described in the literature are indicative of my own experiences. I was a project manager for a major aircraft gas turbine engine manufacturer that initiated development of a new aircraft engine that was to be developed in about one half the normal development time-period, an unprecedented development period in the industry. The final product was completed 4 months later than originally planned (late), with a development cost more than double that originally budgeted (over budget), and with a performance shortfall to specification an order of magnitude off what was acceptable to the customer (inadequate performance). A major redesign effort was initiated to eventually deliver an adequate product, and when it will be finished will result in a product development program that will exceed nominal development time and budget *still* deliver less performance than the customer's specification requirements. Thus, by striving to meet unprecedented schedule, budget, and performance targets, the organization will end up with an eventual project that will have taken longer, cost more, and with less performance than previously experienced. I have also repeatedly heard senior managers question their respective leadership team members "Why do we continually fall short?" and attribute the failures to poor leadership and management. It is this domain where good, well intentioned people doing what they think is the right thing yet yielding undesirable results that the field of System Dynamics is most appropriate for gaining understanding and improving the situation.

3.1.2 Dynamic Perspective on the Problems of New Product Development Projects

As seen in Chapter 2, the field of System Dynamics is useful for understanding the behavior of complex systems. The reason that new product development projects continue to have the problems cited in 3.1.1 is that these projects are not treated as complex dynamic systems (Reichfelt and Lyneis, 1999). First, Reichfelt and Lyneis point out in particular, managers tend to view projects 'statically' (further reinforced by project scheduling/management techniques and tools such as Critical Path Method (CPM) and Microsoft Project) that implies that given tasks in the product development project are completed in their entirety and sequentially before the next task. Second, because of the complex nature of these projects, managers must intentionally limit their scope of understanding to avoid being overwhelmed by details. Thirdly, managers tend to view their individual project separately with no systematic method to learn across projects. Thus the same mistakes are repeated over and over again (Reichfelt and Lyneis, 1999).

Although traditional non-system based techniques, tools, and mental models were once effective in managing simpler less complex projects, as system complexity of NPD projects has increased (in part due to the 'better, faster, cheaper' dynamic, team-based and concurrent engineering trend, and overall increase in system and product complexity discussed in section 3.1.1), they are no longer capable of dealing with system complexity and therefore the same mistakes continue to be made (Reichfelt and Lyneis, 1999). James Lyneis, considered to be possibly the most experienced system dynamic modeler today (Morrison, 2003) has contributed significantly to the literature of system dynamic modeling an analysis of new product development. Lyneis offers a characteristic behavior pattern of most new product development projects. Typical behavior patterns for project staffing and project completion are illustrated in

Figures 5 and 6, respectively.

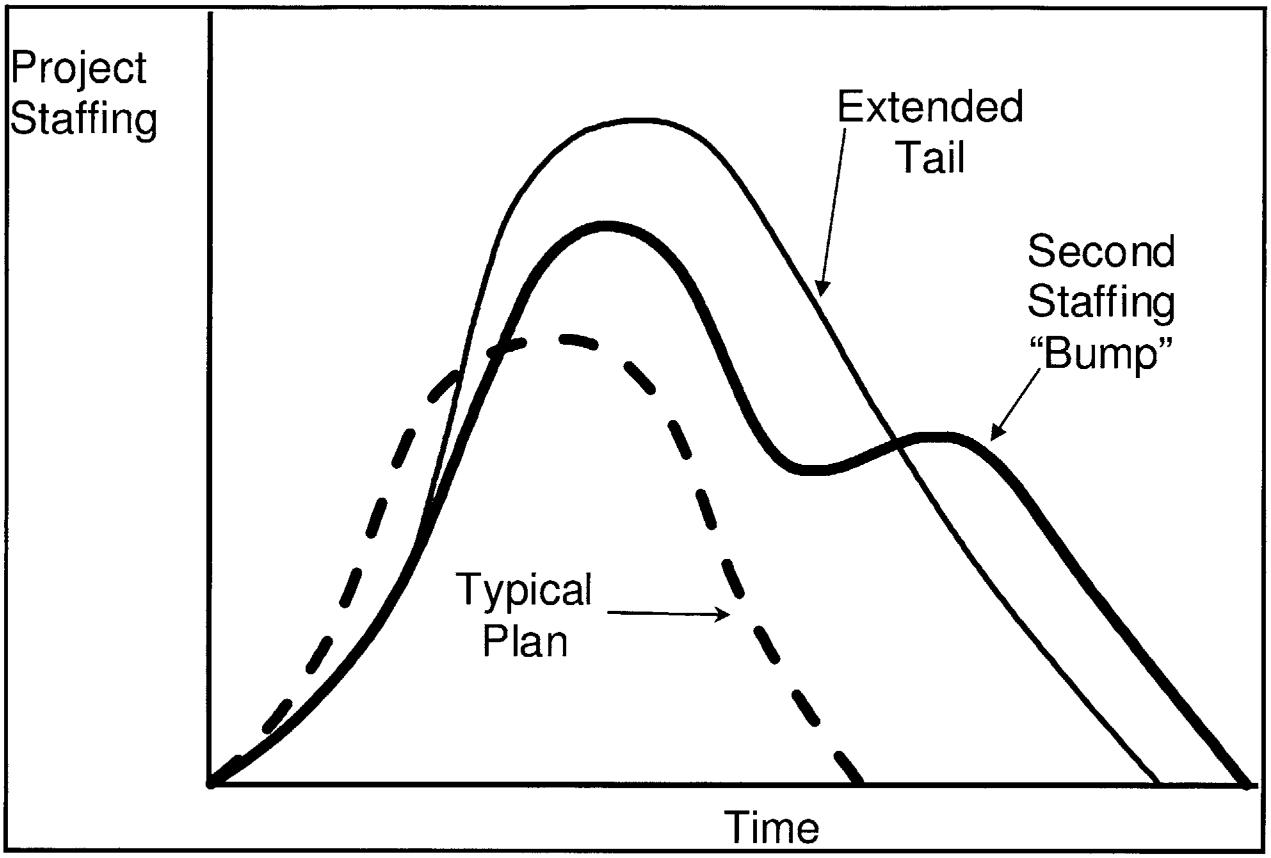

Figure 5. Typical Staffing Behavior Patterns for Projects (adapted from ESD.36J Course Lecture Notes from James Lyneis and in literature (Reichfelt and Lyneis, 1999))

These behavior patterns are characteristic of the problems of NPD projects and have been articulated in the literature by other names such as "the 'lost' year" and the "90% Syndrome" due to the apparent loss of project completion progress near the end of the project (Reichfelt and Lyneis, 1999). The dashed lines in Figures 5 and 6 reflect the typical project plan characterized by a slow ramp up in personnel resources as project work is initiated with a likewise decrease in staffing as project tasks are accomplished and the total amount of work required for the project is finished. Actual projects, however, are characterized by a delay in staffing compared to plan at the onset of the project, followed by staffing levels higher than planned with either an extended staffing profile (the 'extended tail' in Figures 5 and 6), or a temporary decrease in staffing followed by a second increase in staffing (the 'second staffing "bump"' in Figures 5 and 6). These excessive staffing profile and project completion patterns of behavior are fundamental to explaining the reasons why project problems of being late to schedule and over budget are so common (Reichfelt and Lyneis, 1999).

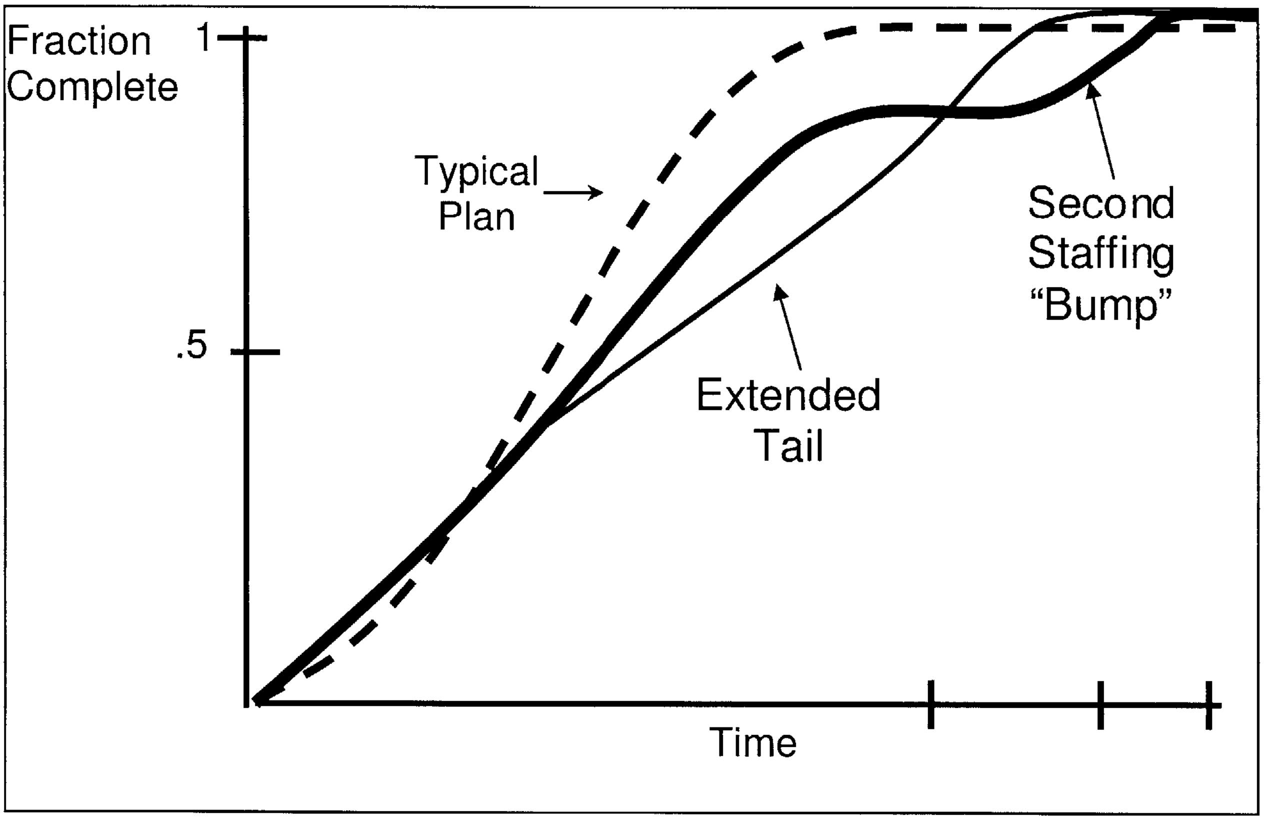

Figure 6. Typical Project Completion Behavior Patterns for Projects (adapted from ESD.36J Course Lecture Notes from James Lyneis and in literature (Reichfelt and Lyneis, 1999))

An understanding of the system dynamics associated with NPD projects is essential to understanding the patterns of behavior found in Figures 5 and 6. As stated in section 3.1.1., the mental models, techniques, and tools used by managers of new product development projects are inadequate for dealing with the project dynamics that drive these undesirable behavior patterns. In particular, two shortfalls of traditional project planning methodologies contribute to the behavior. First, managers tend to assume that staffing will follow the plan and then reduce as work is completed. Actually, staffing usually always occurs slower than planned due to delays associated with getting personnel assigned to the project and adequately trained enough to contribute. Secondly, traditional methodologies assume that productivity is constant throughout the project. Actually, productivity itself is dynamic, usually decreasing initially at the start and through the middle of the project and then rising significantly higher near the end of the project. There is usually a two times difference between maximum and minimum productivity in a project, as illustrated in Figure 7, another pattern of behavior described by Lyneis in the literature (Reichfelt and Lyneis, 1999).

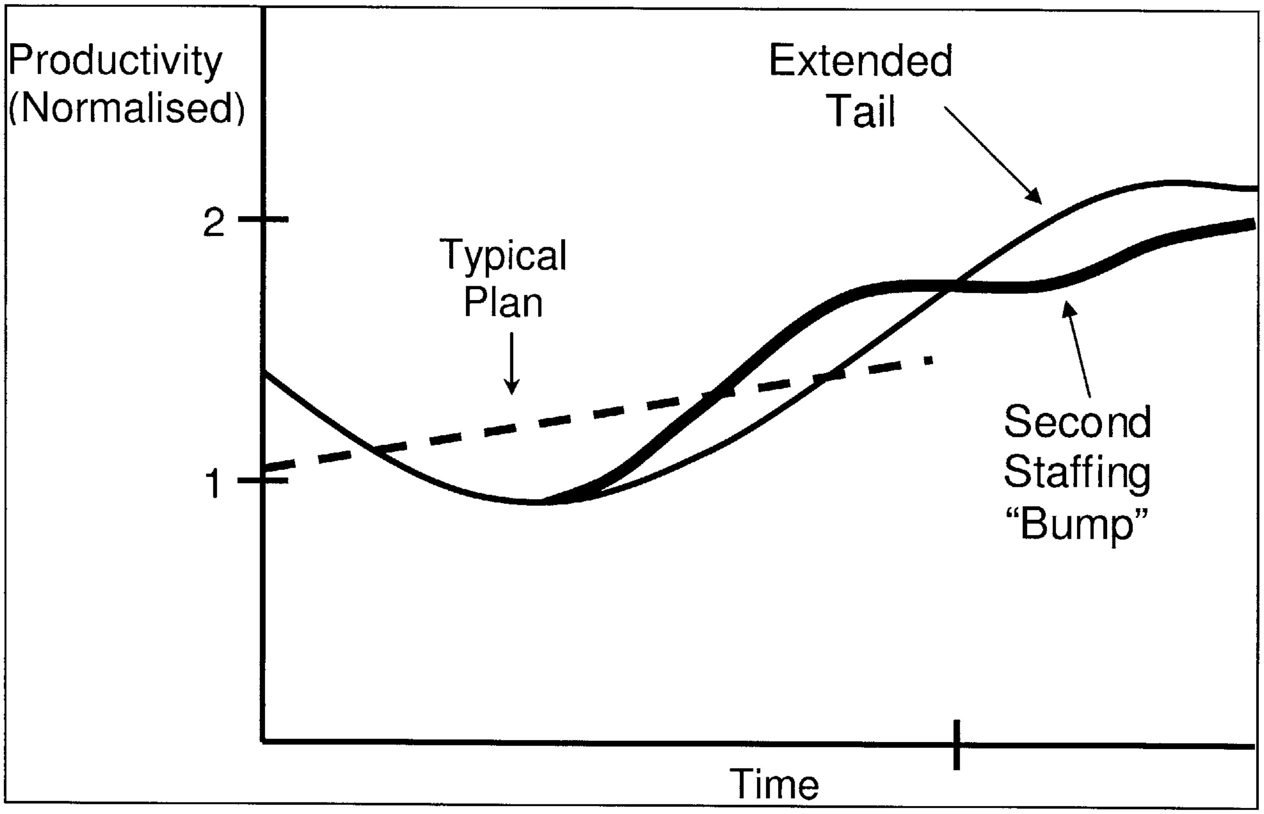

Figure 7. Typical Productivity Behavior Patterns for Projects (adapted from ESD.36J Course Lecture Notes from James Lyneis and in literature (Reichfelt and Lyneis, 1999))

Contrary to the beliefs of most managers, the reasons behind the extended staffing or secondary staffing bumps is not caused by poor planning, lack of productivity or additional work scope identified late in the project, but rather because of *rework* discovered late in the project and then corrected to complete the project. Almost *all* cases of projects characterized by behavior patterns shown in Figures 5 through 7 can be attributed to excessive rework (Reichfelt and Lyneis, 1999). Thus, to fully understand the project dynamics of new product development, a discussion of the "Rework Cycle" is needed[4]

[4] The "Rework Cycle" concept was developed by Pugh-Roberts Associates originally under the name "Work Accomplishment Structure" but has been adopted as a standard system dynamic modeling construct in the literature –[see Cooper, Kenneth G., "Naval Ship Production: A Claim Settled and a Framework Built," Interfaces, Vol. 10, No. 6, December 1980] (Lyneis, 2002).

3.1.3. The Rework Cycle, Feedback Effects, and "Knock-On" Effect[5]

The "Work Accomplishment Structure", or 'rework cycle' is the key system dynamic modeling structure used to understand the dynamics of projects in the new product development domain. The rework cycle is used in all dynamic models of projects in some form (Reichfelt and Lyneis, 1999). The main advantage to the rework cycle is the importance and visibility it gives to the role of known and undiscovered rework on project dynamics. An understanding of these impacts can greatly affect the way projects are viewed and managed. Figure 8 illustrates a stock-flow structure of work accomplishment with associated behavior modes associated with traditional project management techniques and mental models. Because there is no accounting for rework, the staffing and project completion profiles are indicative of the 'typical plan' profiles shown in Figures 5 through 7 (Lyneis, 2002).

A more realistic stock flow structure and associated behavior modes that accounts for both known and undiscovered rework (and with the time delays associated with the discovery of rework) is illustrated in Figure 9. The four stocks in the structure are Work to be Done, Work Really Done, Undiscovered Rework, and Known Rework. The flow *Work Being Done* is the rate at which tasks in the stock *Work to be Done* are depleted by task accomplishment (represented by people times the productivity (task accomplishment rate per person)). Tasks then flow to either the stocks *Work Really Done* or *Undiscovered Rework* based on the *Quality* (percentage of tasks completed with no error and hence do no need to be reworked). The stock of *Undiscovered Rework* is important as the delays of discovering the need to rework tasks that were initially thought to be completed free from errors can be months or years later in a complex project.

[5] The information in this section was obtained primarily from (Reichfelt and Lyneis, 1999) and (Lyneis, 2002). The concepts are substantiated with highly complex, calibrated system dynamics models on numerous case studies.

Thus the length of a project's 'extended staffing tail' or magnitude of the 'second staffing bump' is dependent not only on the *Quality,* but the *Rework Discovery* rate that depletes the *Undiscovered Rework* stock and fills the *Known Rework* stock. Additionally, if the stocks of *Undiscovered Rework and Known Rework* are not recognized by management this will yield a difference between *perceived* and actual progress, another key driver in the "90% Syndrome" phenomenon of thinking the project is 'almost done' when there is still a lot more work (i.e., rework) to be done.

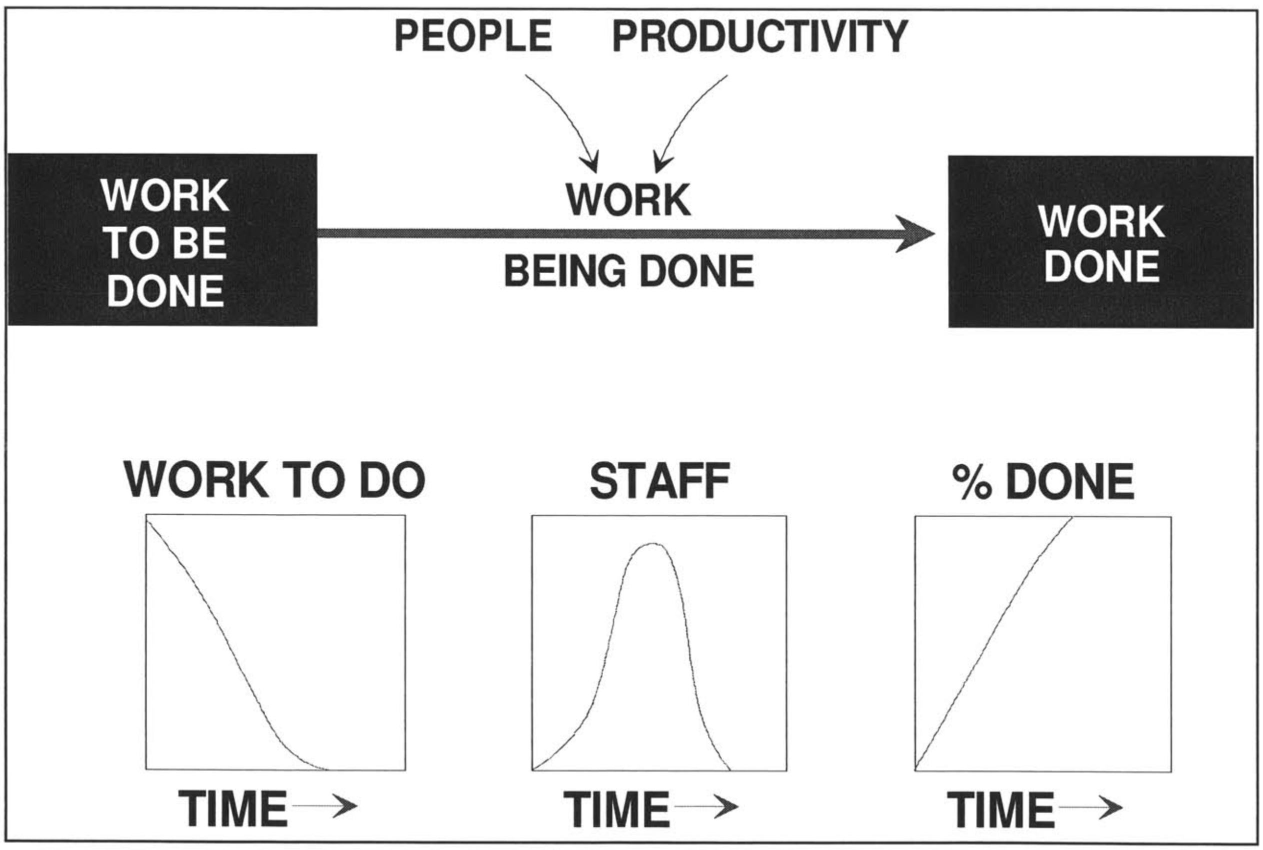

Figure 8. Traditional Project Management Work Accomplishment Stock Flow Structure (taken from ESD.36J Course Lecture Notes from James Lyneis (Lyneis, 2002))

Tasks can go through the rework cycle repeatedly until they are completed without defect. It is this rework that causes the extended staffing tails and secondary staffing bumps.

Thus, while traditional management focuses on improving *productivity* to address its project performance shortfalls, much higher leverage can be attained by mastering the *Quality* and *Rework Discovery* processes that ultimately drive the dynamics of project performance (Reichfelt and Lyneis, 1999).

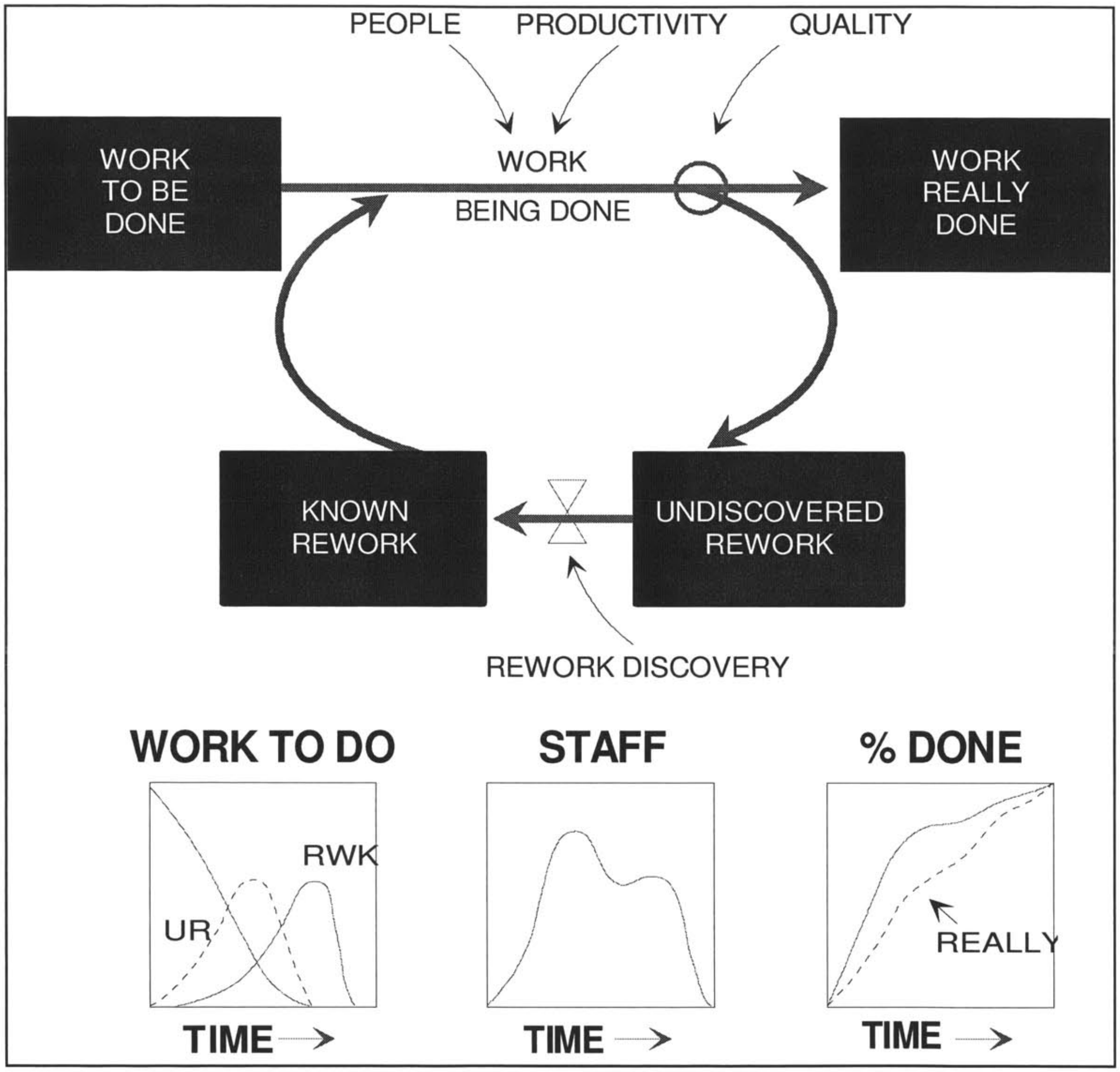

Figure 9. Traditional Project Management Work Accomplishment Stock Flow Structure (taken from ESD.36J Course Lecture Notes from James Lyneis (Lyneis, 2002))

While insight into the dynamics of the rework cycle are helpful in understanding why projects continue to be plagued with late to schedule, over budget, and with inferior performance; two other principles are important to understanding project dynamics. The first principle is the impact of *feedback effects* of the project management system on productivity and quality. A more comprehensive representation of the rework cycle is shown in Figure 10 with the addition of a *Staff* stock with *Hiring* and *Turnover* flows and the major feedback effects of the system structure on *Productivity* and *Quality*.

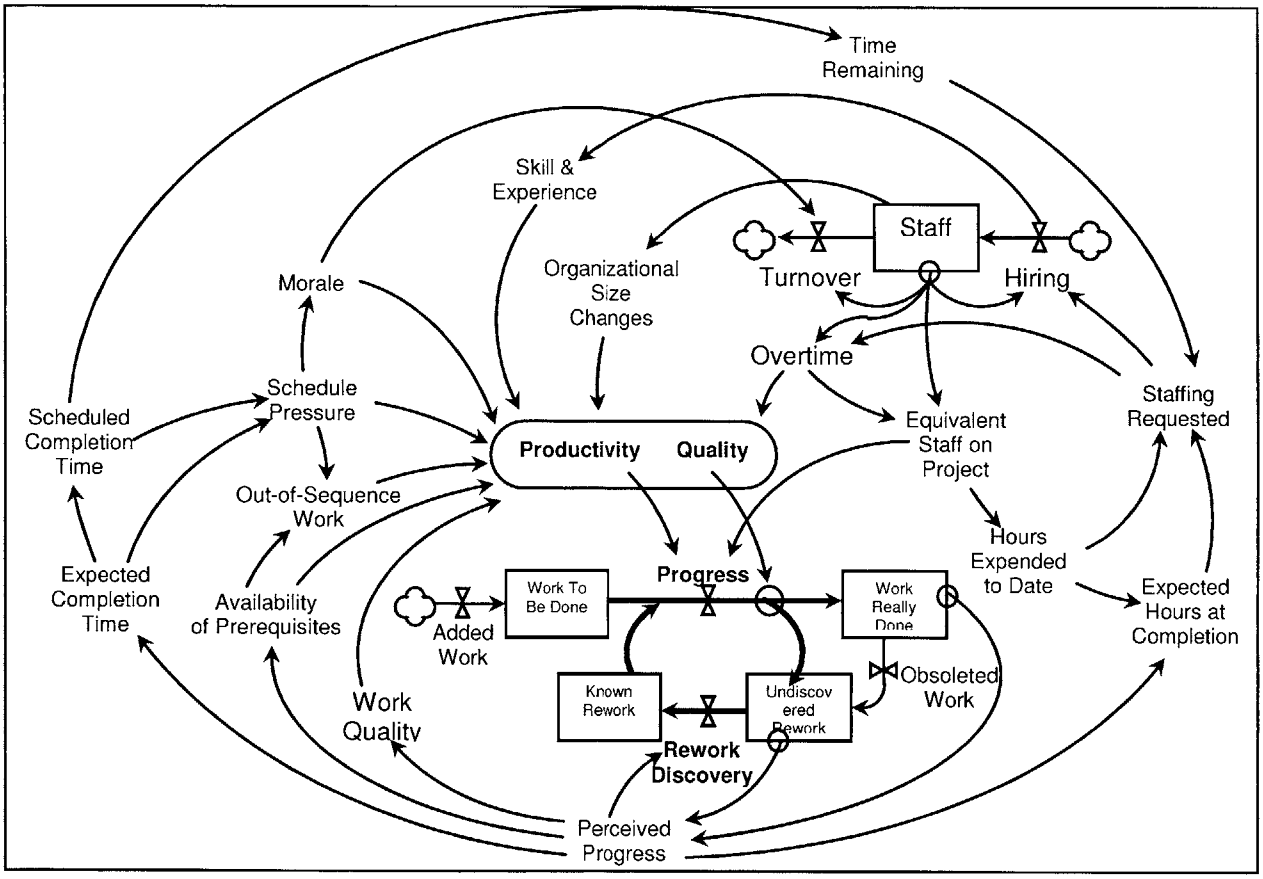

Figure 10. Rework Cycle with Feedback Effects on Productivity and Quality (taken from ESD.36J Course Lecture Notes from James Lyneis (Lyneis, 2002))

As Figure 10 illustrates, many aspects of the project management structure affect the productivity and quality as well as the hiring and turnover dynamics that impact the performance of the project. For example, *perceived progress* which has been shown to be lagging actual progress due to the rework cycle, impacts hiring and overtime policies to address project completion. The hiring process has significant delays, so frequently overtime is opted over hiring to address staffing shortfalls. However, even with the addition of staff through hiring there is a detrimental impact on productivity due to the decrease in average experience level (results in slower task accomplishment with higher defects) and time required for existing employees to assist their new workers. Additionally as the organizational size increases, there is a decrease in productivity due to increased communication requirements and acclimation to new reporting channels (Reichfelt and Lyneis, 1999).

Other significant factors that are prevalent in new product development projects that strongly impact project performance through feedback effects are *schedule pressure, out of sequence work,* and *overtime.* Schedule pressure arises when *perceived project progress* reflects that the project will be late based on known resources and time remaining until completion. Schedule pressure manifests itself in management actions to increase productivity through 'visibility' and 'focus' usually by frequent and high level management reviews. Because of the dynamics outlined in section 3.1.1, the deadline for a project is usually unrealistically attainable and the causal relationships shown in Figure 4 preclude slipping of a deadline as a viable option to relieve schedule pressure. Indeed, schedule pressure has a reinforcing effect in that productivity usually increases in the short-term as a project team increases productivity to meet the requirements of a high level management review. The detriments of schedule pressure can be devastating, however, as it leads to reduced morale (leading to reduced productivity and turnover), more defects, and *out of sequence work* as project teams do 'whatever it takes' to get the job done (Reichfelt and Lyneis, 1999). Schedule pressure is also a strong driver of project "tipping" and 'fire-fighting' to be discussed in section 3.1.5.

Out of sequence work occurs when tasks are accomplished prior to the required prerequisites necessary to adequately perform the tasks are accomplished. Also described as 'cutting corners' this activity is prevalent in projects that are unrealistically planned and/or where schedule pressure is pervasive. Like schedule pressure, it can give the temporary perception that the project is 'getting back on track' through short term productivity improvements. However, because tasks are completed without prerequisites, out of sequence work always leads to increased rework (Reichfelt and Lyneis, 1999). The use of overtime also has a 'better before worse' effect in that productivity increases due to more work hours being performed. Additionally, due to the delays associated with hiring and budgetary constraints that can limit ability to hire, overtime is the preferred method to augment staffing shortfalls in project crisis situations. The temporary boost in productivity is short-lived, however, as sustained overtime leads to fatigue, lower morale, and reduced quality and productivity (Reichfelt and Lyneis, 1999).

In addition to feedback effects on productivity and quality, the second principle that drives project dynamics is the "Knock-On" effect. This effect recognizes that the product development project system is a multi-phased system where the performance of individual phases drives the performance of other phases. The rework cycle shown in Figure 10 can be viewed as the project structure for any given phase in a multi-phased product development project. A multi-phased project structure is shown in Figure 11, where completed tasks of *upstream* phases flow to later *downstream* phases. The individual phase rework cycle structures can be viewed as 'building blocks' that add up together to form the entire system structure that drives product development project behavior (Reichfelt and Lyneis, 1999).

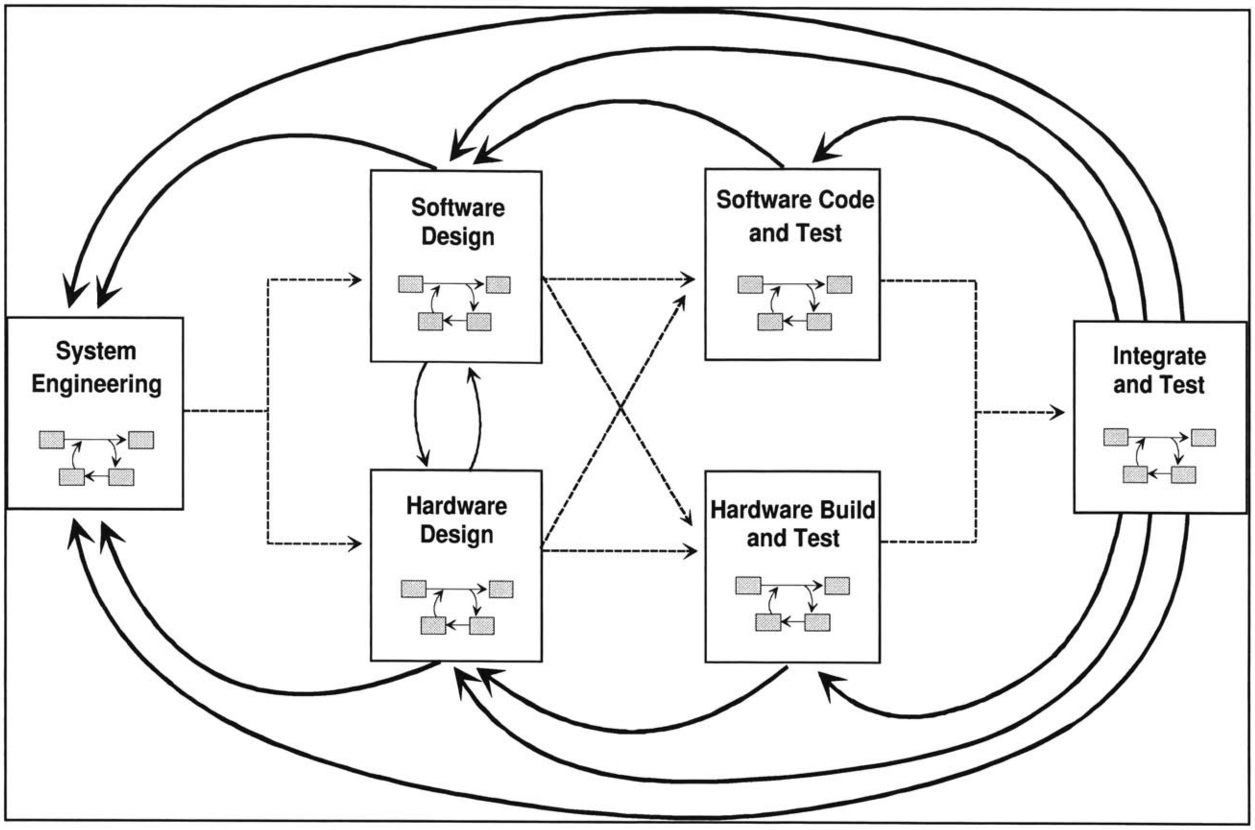

Figure 11. Rework Cycle Representation of Multi-Phase Product Development Project to Represent "Knock-On" Effect (taken from ESD.36J Course Lecture Notes from James Lyneis (Lyneis, 2002))

The knock-on effect amplifies the overall product development project behaviors characterized by the rework cycle and feedback effects described for individual phases described above. This is because behaviors in upstream phases affect the behaviors in downstream phases, and visa versa, as seen by the solid arrows in Figure 11. For example, late task accomplishment in the system engineering phase affects availability of designs to be worked in the software and hardware design phases. Quality of the tasks accomplished in the system engineering phase affects the quality of tasks to be accomplished in the design phases which starts the vicious rework cycle dynamics for that stage. As successive downstream phases are impacted by the errors from upstream phases, they are often 'already behind' and the system responds with the schedule pressure and out-of-sequence responses described above. Additionally, errors found in downstream phases may be attributed to errors in upstream phases that were not previously discovered, thereby contributing to the upstream phase's rework discovery which further impacts all the other downstream phases (Reichfelt and Lyneis, 1999).

In conclusion, the project dynamics of NPD projects are characterized by the rework cycle, feedback effects of the system structure on productivity and quality, and knock-on effects of multiple phases. These project dynamics work to hamper project completion when projects start to go wrong by initiating vicious cycles that ultimately slow productivity, yield poor quality, and slow the discovery of defects. These dynamics result in the extended staffing tails or secondary staffing bumps that drive late schedule and over budget project performance. Management actions of late hiring, schedule pressure, overtime, and cutting corners through out-of-sequence work while in the short term appear to better the situation actually reinforce these vicious cycles and yield even worse results due to the project dynamics (Lyneis, 2002). However, because these management actions are intuitive and positively reinforced by traditional project management systems (rewards for heroic fire-fighting, accolades from management for 'pulling it off', etc.) and because by nature, managers tend to opt for 'better before worse' policy and decision making actions (Repenning, 2000).

The lessons to be learned from the work of Lyneis and others in the literature of new product development project management can be summarized as follows (Reichfelt and Lyneis, 1999):

- **Get it right the first time** – Since rework is the primary driver of late to schedule and over budget project performance, it is more important to get it right the first time and be late than it is to be on time with defects that will impact further in-phase and downstream phase work. Thus management must make a rigorous effort to identify and resolve rework as far upstream as possible and ensure that deliverables from those phases meet the requirements of downstream errors with no defects.

- **Start with a realistically attainable schedule and budget –** The undesirable project dynamics described above are set in motion and exacerbated when there is an unattainable schedule and/or budget. This is the most important lesson to be learned from a system dynamic perspective on project management (Lyneis, 2002).
- **Start with an experienced team and hire early** – Because of the delays associated with hiring new staff and the importance of staff experience on productivity and quality, managers should aggressively hire staff early in a project and ensure they have an experienced team. While this practice may impact budget negatively initially, this 'worse-before-better' approach will help to avoid the vicious cycles of rework.
- **Execute work in an optimal sequence –** Avoid 'cutting corners' to meet interim milestones and ensure that all work performed in each phase is correct before passing to downstream phase.
- **Avoid the use of sustained overtime** – The benefits of overtime are short-lived and always lead to reduced productivity and quality in addition to fatigue, lower morale, and higher employee turnover.
- **Avoid the use of schedule pressure** – Management actions to improve 'visibility' and 'focus' through frequent and high level management reviews exacerbate the vicious cycles that lead to poor project performance.

- **Slip interim milestones** – Milestone due dates that are kept in place when a project gets in trouble exacerbate the vicious cycles that lead to poor project performance.

3.1.4. Process Concurrence and the Ford-Sterman Model[6]

While the rework cycle, feedback effects on productivity and quality, and knock-on effects described above are well substantiated in the literature to explain the behavior of NPD projects, further work has been conducted to better understand the project dynamics of these systems. David Ford and John Sterman published a significant work in the literature in 1998 that included a separate modeling structure for *process concurrence* (Ford and Sterman, 1998). Because the system dynamics work being conducted by BP is based on the work of Ford and Sterman, BP required that the system dynamics model that the MFS use be based on a Ford-Sterman model. Therefore, a discussion of process concurrence and the Ford-Sterman model is included here.

Previous system dynamics models of project management were based on stock flow models that incorporated staffing resources, project scope (tasks and deadlines), and productivity and quality relationships as described in section 3.1.3. The term 'process concurrence' recognizes that task accomplishment must be represented by the accomplishment of tasks previously completed. *Internal process concurrence* refers to the ability of work to be accomplished based on how much work has progressed so far *within* a given phase of the product development project. For example, in the phase of assembling a jet engine, the compressor cannot be assembled onto the engine until the compressor itself is built. Thus the engine assembly phase is constrained by the progress of the compressor assembly task. *External process concurrence* refers to the ability of work to be accomplished based upon the completion

[6] The information presented in this section was obtained primarily from Dynamic Modeling of Product Development Processes (Ford and Sterman, 1998) in which the concepts are substantiated through calibrated models and cases.

of work in an *upstream* phase. For example, a jet engine could not be tested in the testing phase until the engine was assembled in the assembly phase.

Previous system dynamic models addressed this concurrency implicitly through the productivity relationships (see *Availability of Prerequisites* in Figure 10) and knock-on effect modeling through the building block approach (Figure 11). However, research revealed that processes frequently constrained the ability to accomplish work, not resources. Hence without *explicitly* modeling these concurrencies, previous models could predict that tasks could be completed instantly if resources were available (Ford and Sterman, 1998)[7]. Ford and Sterman's contribution to the NPD project management literature is the incorporation NPD process structure to system dynamics modeling.

The Ford-Sterman model, which recognizes known and undiscovered rework (and hence is a form of the rework cycle), is modeled differently than described in section 3.1.3. Figure 12, adapted from (Ford and Sterman, 1998) shows the product development process structure for a single project phase. It incorporates four stocks of *Tasks Completed not Checked, Tasks to be Iterated, Tasks Approved,* and *Tasks Released.* The flow *Initial Completion Rate* based on internal process concurrence feeds the *Tasks Completed not Checked* stock. The flow *Discover Defective Task Rate* depletes the *Tasks Completed not Checked* stock and feeds the *Tasks to be Iterated* stock. Tasks in this stock are completed through the *Iterate Task Rate* flow. The *Approve Task Rate* flow depletes the *Tasks Completed not Checked* stock and feeds the *Tasks Approved* stock. The *Release Task Rate* flow feeds the *Task Released* stock. This is the stock that feeds the *Initial Completion Rate* flows for upstream processes (Ford and Sterman, 1998).

[7] A convenient expression for understanding the importance of process concurrence is "Nine women can't make a baby in one month" (Sterman, 2003). Often when told of the time it takes to accomplish a task, management will respond with "That's unacceptable – what do you need to make it X time?" This response reflects the lack of

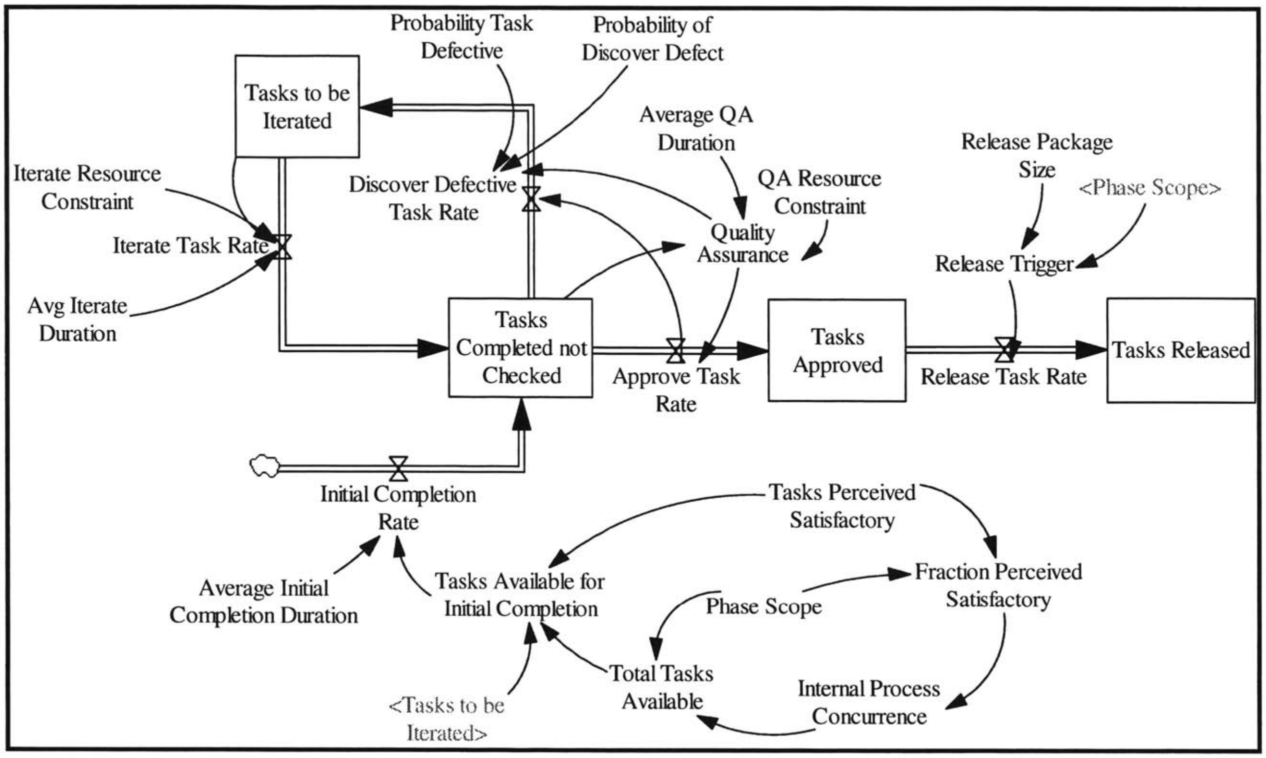

Figure 12. New Product Development Process Structure for a Single Project Phase (adapted from Ford and Sterman, 1998).

Ford and Sterman showed that the relationship of internal process concurrence was important to capturing the dynamics of projects in the NPD domain. Frequently, this relationship is non linear, and changes as the project approaches completion. As a project approaches completion, tasks become more concurrent and more integration activity becomes 'less parallel' requiring more tasks to be completed before subsequent tasks in the phase can be completed thereby constraining initial completion rate (Ford and Sterman, 1998).

A similar concurrence relationship was identified between phases, called *external process concurrence* (EPC). Ford and Sterman's approach is more adequate to describe this relationship than traditional methods devised to capture the same dependency such as Critical Path Method

understanding of complex system tasks and the importance of concurrence and the non systems thinking is reinforced when heroic efforts are made to meet the revised deadline.

(CPM) and PERT. First, EPC describes the dependency relationship throughout the project duration, not just at the beginning and end of the phases. Second, EPC recognizes non-linear relationships that change as the project progresses. Third, EPC allows a dynamic interaction between phases that regulates the throughput of tasks through the system based on conditions of the project through the life of the project (Ford and Sterman, 1998). An interesting finding of this research is the differences in mental models amongst upstream and downstream contributors in the NPD system of the amount of external process concurrence present between phases. Upstream contributors assume smaller amounts of concurrence are present then the perceptions of downstream contributors. The actual amount EPC present can significantly drive project performance through constrained task completion (Ford and Sterman, 1998).

The importance of managing EPC relationships is recognized by the NPD process of coordination which describes the management actions taken to ensure that upstream tasks are completed to satisfy the needs of upstream phases free of defects. The Ford-Sterman modeling of coordination allows distinction between two important types of defects associated with NPD processes: 1) defects that occur due to factors internal to a development phase and 2) defects that occur in upstream phases due to defects inherited from upstream phases. Figure 13 shows the Ford-Sterman representation of a multiple-phase NPD process and coordination structures. In this representation, the coordination activity (in the form of meetings, reviews, communication via e-mail, etc.) depicted by the *Coordinate Task Rate* flow depletes the *Tasks to be Coordinated* stock. The *Tasks to be Coordinated* stock is thus fed by both the *Discover Inter-Phase Defective Task Rate* (the flow of tasks not yet released but need coordination to ensure no downstream defects are detected later) and by the *Coordination due to Downstream Quality Assurance* (the flow of tasks previously released but through inherited defects to downstream process must be iterated to satisfy downstream requirements) (Ford and Sterman, 1998).

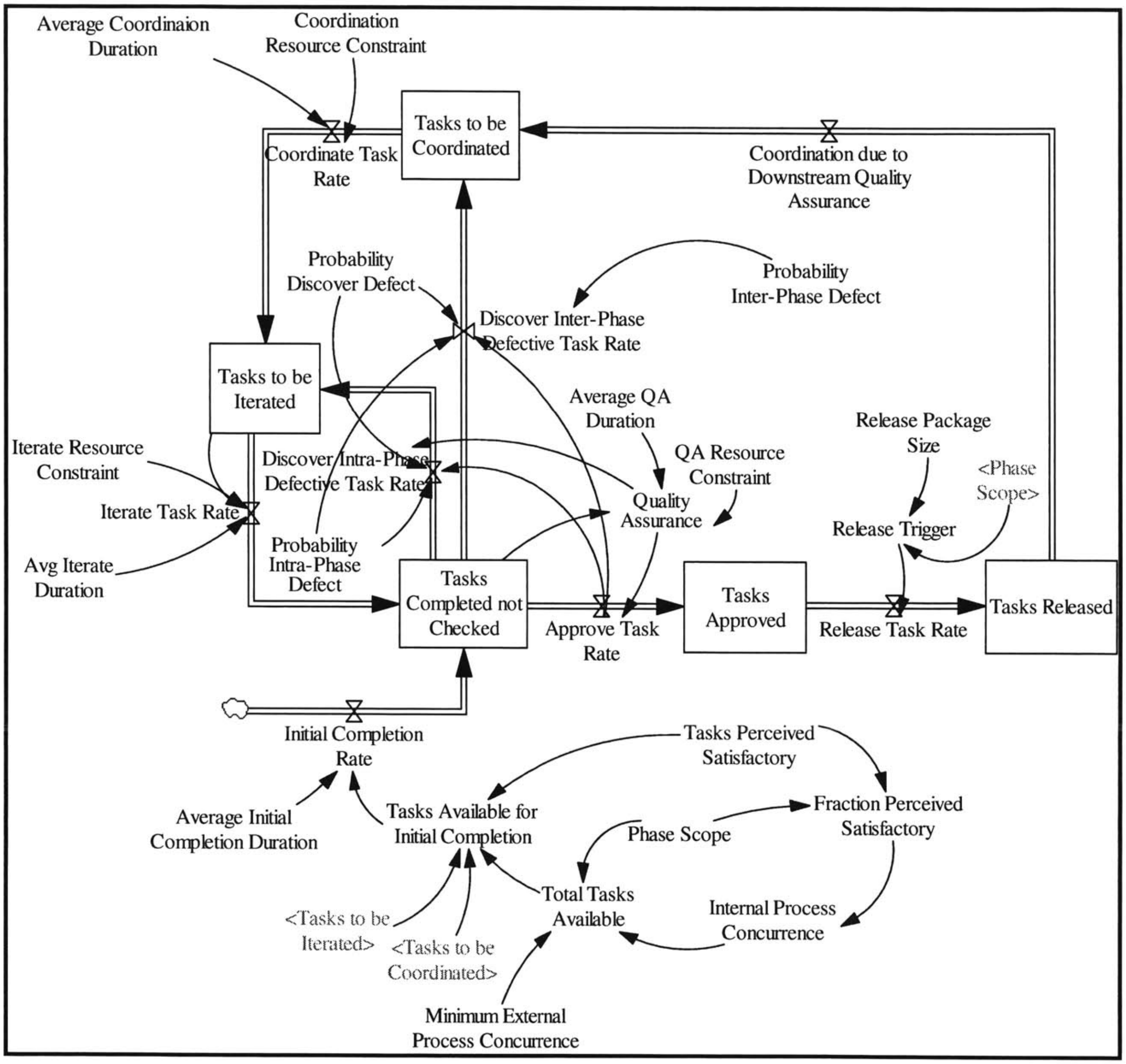

Figure 13. New Product Development Multi-Phase Process and Coordination Structures (adapted from Ford and Sterman, 1998).

The importance of concurrence and coordination are being recognized as crucial to dealing with system complexity in NPD. Concurrent Engineering, team-based organizations and processes have been developed to address this. The Design Structures Matrix (DSM) method developed at MIT and being further enhanced by Stephen Eppinger et al. is another method that includes concurrence and in particular iteration where traditional methods do not to address this need. However, DSM does not address the dynamic relationship that these dependencies have on the system (Ford and Sterman, 1998).

In conclusion, the Ford-Sterman model recognizes the importance of internal and external process concurrence on the dynamics of projects and models them separately from the other processes. Through this approach, several balancing loops not related to resources, targets, and scope frequently missed by managers such as the constraint on available work at the beginning of a project phase and the amount of work available to be completed when upstream work is completed are captured and are key to understanding the dynamics of projects (Ford and Sterman, 1998).

3.1.5. Fire-Fighting and Project "Tipping" in New Product Development[8]

Nelson Repenning has furthered the field of NPD project dynamics research by introducing the concept of modeling the dynamics associated with 'fire-fighting' in NPD organizations. Fire-fighting is defined as the practice of allocating resources to unforeseen problems late in a product's development cycle in an attempt to deliver the product to market on schedule. Fire-fighting is a pervasive phenomenon in NPD, so much that it is considered standard practice and is expected in any NPD project. Repenning theorizes that fire-fighting is a system phenomenon that affects not only the project that it is applied to, but undermines the ability of an organization to sustain its new product development capability. Using the term "tipping point" from epidemiology where in infectious diseases the tipping point represents the point of susceptibility and infectivity where beyond it the disease becomes epidemic, Repenning

[8] The information obtained for this section was obtained primarily from (Repenning, Goncalves and Black, 2001) and (Repenning, 2002).

contends that the practice of fire-fighting in an organization can likewise spread and contaminate all projects in an NPD organization until all projects are in continual fire-fighting mode.

The basic premise of the tipping point concept is that as projects that are in the latter phases of development (near launch or production ramp-up) draw resources from other projects in the 'front-end' phases to address unforeseen problems (i.e., 'fire-fighting'), the projects with reduced front-end staffing as a result occur with reduced quality (usually due to a 'cutting-corner' dynamic in response to getting the project to the next development phase). This reduction in quality leads to further development problems for *that* project in later year(s) (due to front-end phase dependencies that impact success of downstream phases). Therefore, additional fire-fighting will be needed which will impact the front-end of the next project. This is a perpetuating phenomenon. What is interesting is that because of the importance given to projects in latter stage, fire-fighting is actually *rewarded* due to usually heroic efforts to bring the later-phase project 'home'. Again, the time delay associated with the impact of reducing staff on front-end phase projects is not seen until the next year, so unless the dynamics are understood, managers are unable to learn the consequences of their actions. Without learning, they continue to make the same mistakes that perpetuate fire-fighting.

3.2 The Role of Simulation in Improving Organizational Learning[9]

While the principles learned from understanding the dynamics of projects using system dynamics are well established in the literature, the transformation of these principles to working practice in NPD organizations is lacking. While there have been successful applications of system dynamics in NPD organizations to reverse the trend and yield successful projects[10],

[9] The information for this section was obtained primarily from (Senge, 1990) and (Morecroft and Sterman, 1994).
[10] See (Lyneis, Cooper and Els, 2001) for details of the Peace Shield project undertaken by the Raytheon Corporation.

widespread application of system dynamic principles to change the mental models, policies, procedures, and decisions to exploit these principles has not occurred. Reluctance to adopt these principles is characteristic of the "policy resistance" phenomenon rampant in all domains. In this section, the literature on the role of simulations in improving organizational learning and affecting changes in policies is explored.

Several principles of learning need to be understood. First, humans learn best by 'doing' or first hand experience. This is evident in learning to walk, ride a bike, drive a car, play the piano, etc. (Senge, 1990). The learning process can be described by the following process steps: act -> observe -> adjust. However, to learn from doing, one must have feedback from their actions that is timely and clear. The problem with actions in complex systems is that often the feedback is neither timely nor clear, i.e., it is ambiguous. Senge describes this as the "dilemma of learning from experience" in which "we learn best from experience, but we never experience the consequences of our most important decisions" (Senge, 1990, p. 313). Management Flight Simulator's (MFS) provide a means where time is compressed so that the consequences of actions can be observed immediately after the decisions are made, thereby contributing to learning. MFSs also provide a way of compressing space. A MFS based on a system dynamics model can reveal the impact of actions on elements of an organization that otherwise would be inaccessible due to location, cost, or risk. MFSs therefore provide a risk free, inexpensive way for participants to experiment. In addition to 'learning by doing', they are able to observe the feedback (consequences) of their actions that are timely and relevant. (Senge, 1990; Bakken etal., 1990)

In addition to improving learning, MFSs also have additional benefits. Proper incorporation of MFSs in training curricula have been shown to initiate productive discussion amongst organization members on issues that would otherwise not be discussed (Senge, 1990). Through this interaction and dialogue comes even greater learning above that originally intended in the original simulation. This dialogue comes from questioning the assumptions of the model, which often reveals misconceptions amongst the mental models of the participants. MFSs are thus very effective in 'shaking up' the perceptions that form mental models, thereby improving their ability to be changed. Another benefit is giving visibility to factors and concepts that are not normally visible in a real-world situation, a factor that leads to a phenomenon of realization that Senge characterizes as "*discovering that you're not really as good as you thought you were*". This realization also reveals that the metrics commonly used to measure (and thereby drive) performance while tangible is not really driving good system behavior (Senge, 1990).

Graham and others have shown that traditional methods of teaching (case studies, power point presentations, etc.) are not as effective as simulations in learning because participants are not able to test hypotheses of the effects of alternate actions. The ability of humans to comprehend these scenarios in complex systems is not adequate, and they frequently jump to (wrong) conclusions also referred to as judgment biases. (Graham et al, 1994).

However, the role of a MFS must be part of an overall "learning system" which includes *investigation, conceptualization or framing,* and *transfer* of the principles to other application domains, i.e., problems. In *investigation,* the participant considers the outcome of their decisions and develops a *strategy* of policy decisions to meet system behavior goals. They then transform through *conceptualizations or framing* these ideas by first 'playing' the simulation and then analyzing the results of their actions by reflections. Finally, the ideas must be adequately internalized to be able to *transfer* these principles to other contexts. Incorporation of a MFS without following this 3 step process results in a phenomenon called "video gaming" where participants simply play the MFS like a video game to try to 'beat the game' by getting better scores – no learning takes place (Grahama et al, 1994)

In conclusion, MFSs are effective methods when used as part of an overall investigation/conceptualization/transfer learning system to teach system dynamic principles and change mental models. They are also valuable in instilling productive dialogue amongst participants in an organization to reveal insight on processes and uncover misconceptions of the assumptions that underlie the mental models in the organization. The next chapter will build upon the information learned in this chapter to achieve the thesis objectives.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 4

Methods

This chapter details the methods used to achieve the thesis objectives. It includes a description of the system dynamics model that the MFS is based on (here fore referred to as the BP NPD MFS model). The chapter also includes the steps taken to develop the MFS including the product development process used to meet the requirements of the MFS. A description of the MFS is included in also this chapter.

4.1 System Dynamic Model of Single-Phase, Single Project NPD Project

This section details the development of the system dynamics model on which the MFS is based upon. The model was developed through a team effort by Nelson Repenning, Nick McKenna, and the author in conjunction with parallel research on improving the NPD process for the petroleum industry for the BP Corporation. The original model was based upon a modification of the Lyneis rework cycle described in section 3.1.2, however at the request of BP was reconfigured to that based on a Ford-Sterman model described in section 3.1.3. This was in part due to past BP system dynamics efforts and management training that had been based on a BP customized version of the Ford-Sterman model and also due to the substantiation of the Ford-Sterman model in the literature (Johnson, 2003). Since the ultimate objective of the MFS is *learning* for BP management, introduction of models and concepts in the MFS that were different then those previously introduced to BP managers would impede learning. The model was developed to capture the dynamics of the rework cycle and internal concurrence relationships described in sections 3.1.2 and 3.1.3.

4.1.1. Work Flow Structure

The work flow structure is based on a single-phase Ford-Sterman model. As this MFS is intended to be updated to reflect a multiple-phase project, some attributes of the multiple-phase structure are included for ease of updating. The work flow structure is shown in Figure 14.

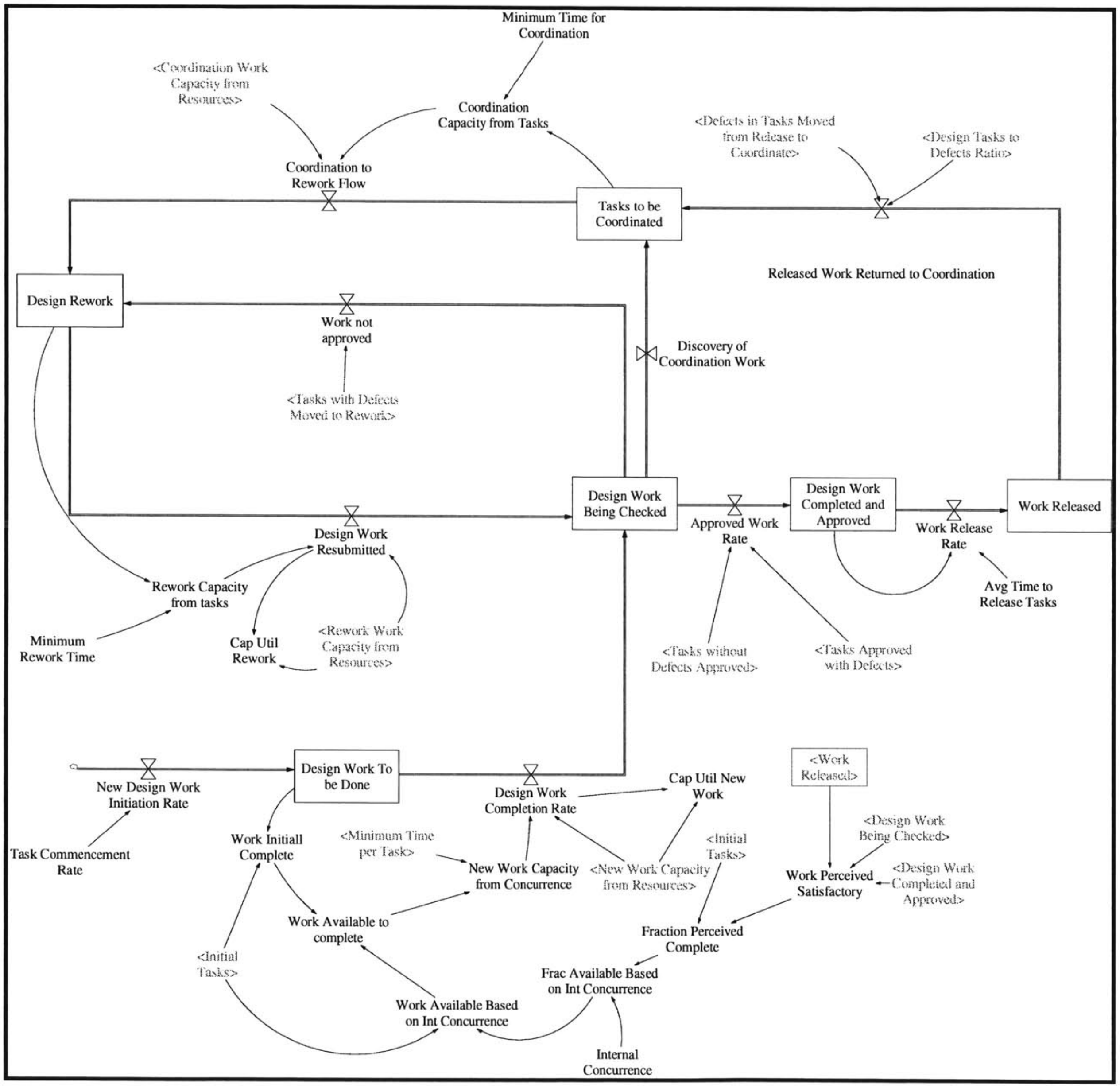

Figure 14. Work Flow Structure of BP NPD MFS Model

The model is a simplified version of a more complex version of the Ford-Sterman Model. The stock flow structure incorporates five stocks to represent the flow of accumulations of tasks in respective stages of the single phase. Unless otherwise specified, the definition of a stock is defined by the initial value of the stock plus the integral of the sum of the in-flowing flows minus the out-flowing flows. The stock *Design Work to be Done* is fed by the *New Design Work Initiation Rate* that is included to model project scope creep (the Design Work to be Done is initialized by the value *Initial Tasks*). The flow *Design Work Completion Rate* represents the completion of design tasks and feeds the stock *Design Work Being Checked.* The Design Completion Rate is equal to the work available to complete (based on internal concurrence) divided by the minimum task per task. The internal concurrence relationship is a simple linear function based on the perceived fraction of work complete (table function of internal concurrence is shown in Figure 15). This structure precludes the unrealistic achievement of tasks with infinite resources.

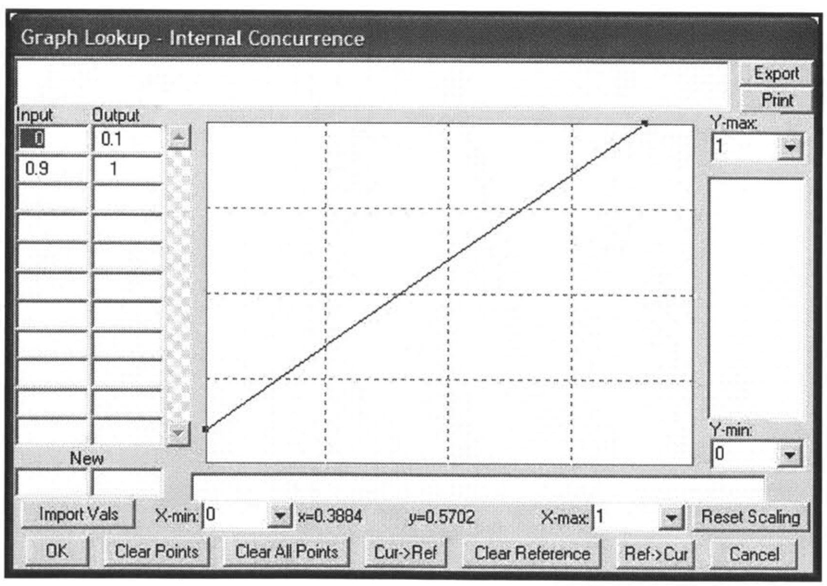

Figure 15. Internal Concurrence Relationship Table Function

From the Design Work Being Checked stock, tasks flow to either the *Design Work Completed and Approved* via the *Approved Work Rate* flow or the *Design Rework* stock via the *Work Not Approved* flow. The flow *Discover of Coordination Work* is included in the structure for later updating for multiple-phases, but is not defined in this single-phase version of the model. The prior flows are defined by a defect flow structure to be described later. Tasks move from the Design Work Completed and Approved stock to the *Work* Released stock through the *Work Release Rate* flow which is simply the ratio of tasks in the Design Work Completed and Approved stock divided by the average time to release tasks (*Avg Time to Release Tasks).* From the Work Released stock, some tasks are returned ala the rework cycle to the *Tasks to be Coordinated* stock via the *Released Work Returned to Coordination* flow (to be defined later in section on defect flow structure). Thus, while this is a single-phase model, it *does* account for defects found in tasks released and defects found prior to release. From the Tasks to be Coordinated stock, work flows to the *Design Rework* stock via the *Coordination to Rework Flow* flow. This flow is defined as the minimum of the coordination capacity based on resources (tasks to be coordinated divided by the minimum time for coordination) and the *Coordination Work Capacity from Resources* (to be defined later in staffing flow structure).

Work accumulated in the *Design Rework* stock from the Coordination to Rework Flow and Work Not Approved flows then flows back to the Design Work Being Checked via the *Design Work Resubmitted* flow. The Design Work Resubmitted flow is defined as the minimum of the rework capacity based on rework tasks (Design Rework divided by minimum rework time) or the rework capacity based on resources (to be defined later in staffing flow structure).

The work flow structure for the BP NPD MFS model contains all the key rework cycle dynamic structure elements shown to be crucial in modeling the project dynamics of new product development projects described in Chapter three. It includes the internal process concurrence relationships, distinction in defects found during the development phase and after completion within the phase, and both resource and concurrence constraints shown to be important in section 3.1.3. Additional structures will add the feedback effects that are also fundamental to modeling the dynamics of projects.

4.1.2. Defect Flow Structure

This model incorporates a defect co-flow structure to more easily capture the feedback effects on productivity and quality described in Chapter 3. The defect flow structure is shown in Figure 16. The structure consists of five stocks that parallel the stocks in the work flow

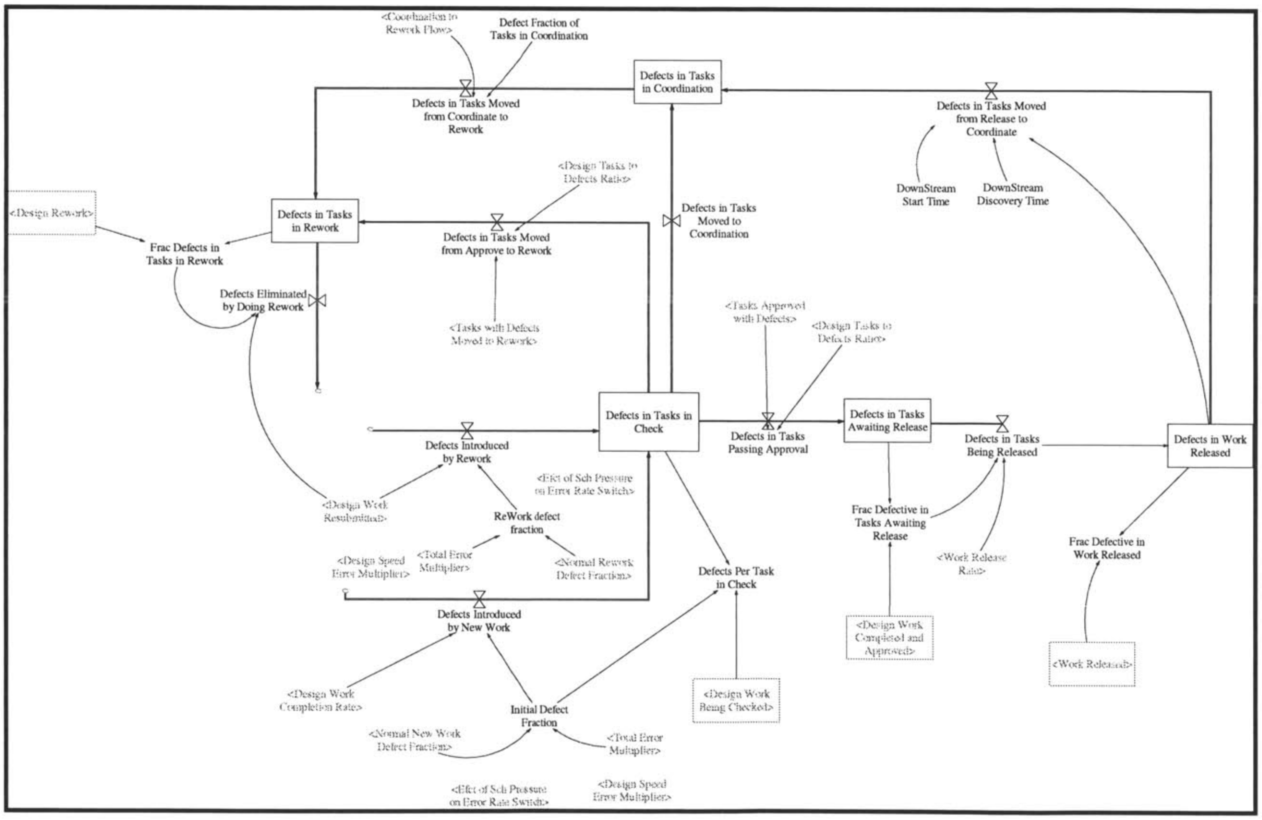

Figure 16. Defect Flow Structure of BP NPD MFS Model

structure. The names of the five stocks are *Defects in Tasks in Check, Defects in Tasks Awaiting Release, Defects in Work Released, Defects in Tasks in Coordination*, and *Defects in Tasks in*

Rework. The flow *Defects Introduced by New Work* represents the defects generated in the introduction of new work (phase scope) and is equal to the *Design Work Completion Rate* (previously defined in section 4.1.1.) times an *Initial Defect Fraction.* Defects generated in this flow accumulate in the Defects *in Tasks in Check* stock. Defects then flow to either the *Defects in Tasks in Rework* stock via the *Defects in Tasks Moved from Approve to Rework* flow, or to the *Defects in Tasks Awaiting Release* via the *Defects in Tasks Passing Approval* flow. This latter flow is defined by *Tasks Approved with Defects* (Approval Work Rate*Defects Per Task in Check*(1-Frac Defects Identified)*Design Tasks to Defects Ratio) divided by the *Design Tasks to Defects Ratio* (set to 1 in this model for simplification by assuming that when a defect exists, it is unique to that design task).

The Defects in Tasks Awaiting Release stock is depleted by the *Defects in Tasks Being Released* flow which is defined by the Work Release Rate (defined in section 4.1.1) times the *Frac Defective in Tasks Awaiting Release.* This flow feeds the *Defects in Work Released* stock. Defects flow from this stock to the *Defects in Tasks in Coordination* via the *Defects in Tasks Moved from Release to Coordinate* flow (defined by *DownStream Start Time* times *Defects in Work Released* divided by *DownStream Discovery Time*). Since this is a single-phase model, there are no downstream phases; however, the *DownStream Start Time* and *DownStream Discovery Time* are set as constants to simulate the discovery of defects in work already released. A flow *Defects in Tasks Moved to Coordination* structure is also included for later inclusion of multiple phases, but as with the work flow structure is not defined for this model. Defects flow from the *Defects in Tasks in Coordination* stock to the *Defects in Tasks in Rework* via the *Defects in Tasks Moved from Coordinate to Rework* which is simply the Coordination to Rework

Flow times Defect Fraction of Tasks in Coordination (assumed to be equal to one as for this case all tasks flowing through coordination are defects discovered after work has been released).

The stock *Defects in Tasks in Rework* is fed also by the flow *Defects in Tasks Moved from Approve to Rework* which represents the defects found in checking and is equal to *Tasks with Defects Moved to Rework* divided by *Design Tasks to Defects Ratio.* Finally, defects that are eliminated due to rework are represented by the flow *Defects Eliminated by Doing Rework* which is equal to *Design Work Resubmitted* divided by *Frac Defects in Tasks in Rework* (Defects in Tasks in Rework divided by Design Rework). A complete equation listing for the entire model is included in Appendix A, but the following figure is also included to further illustrate the relationships that further define the task defects in the model.

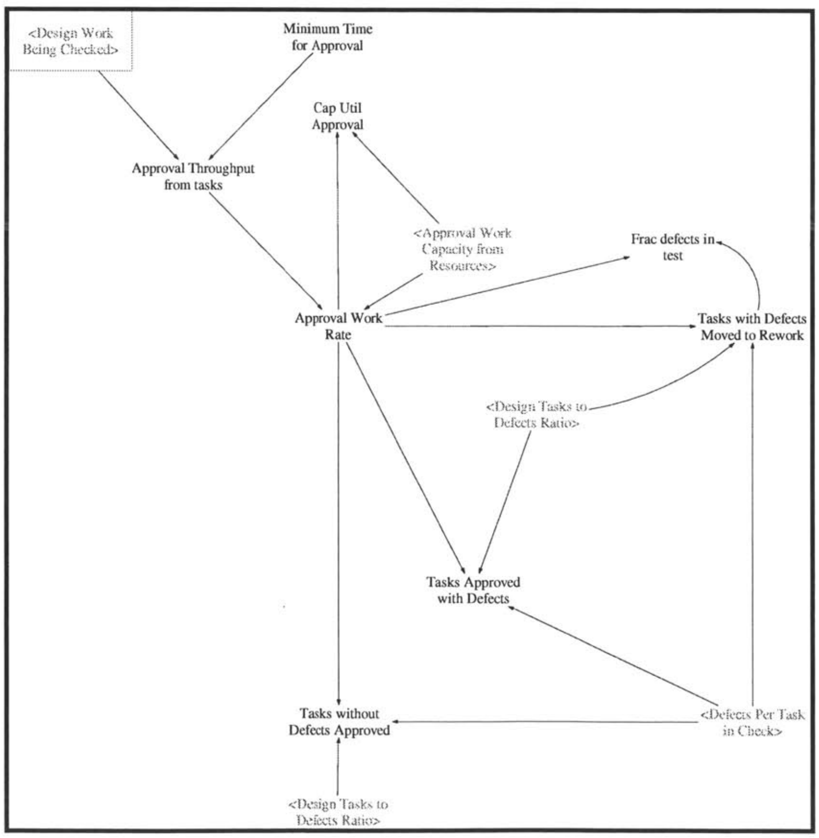

Figure 17. Approved Task Defect and Tasks with Defects Moved to Rework Definition Relationships

Figure 17 shows the relationships between task approval and design tasks defect ratio to determine tasks approved with defects and tasks with defects moved to rework. The relationships of feedback effects on quality (defects and defect discovery) will be discussed in a later section.

4.1.3. Staffing Structure

The model incorporates a staffing structure that models hiring based on desired project staffing from project status, turnover through attrition, and experience level to account for the feedback effects of experience to be discussed in later sections. Figure 18 shows the Staffing Structure of the BP NPD MFS model. This structure consists of three stocks: Desired Project Staff, New Project Staff, and Experience Project Staff. The stock *New Project Staff* is fed by the flow *Hiring Rate* which is equal to the *Desired Gross Increase in Staff* (*Desired Net Increase in Staff* divided by the *Time to Hire New Staff* plus *Total Attrition*). From this stock, new project staff flow to the stock *Experienced Project Staff* through the flow *Experience Gain Rate* which is equal to the New Project Staff divided by the *The Time to Gain Experience* (set as a constant 52 weeks in this model). A flow *New Staff Attrition* also depletes the New Project Staff stock, but is zeroed out in this model for simplicity. Personnel in the Experienced Project Staff leave this stock either through the *Experienced Attrition* flow (defined as Experienced Project Staff times the *Frac Exp Leaving Project* (1/week)) or the *Exp Staff Transferred* flow (defined by maximum of zero or the negative of Desired Gross Increase in Staff to account for reduction in staff when project is ahead of schedule).

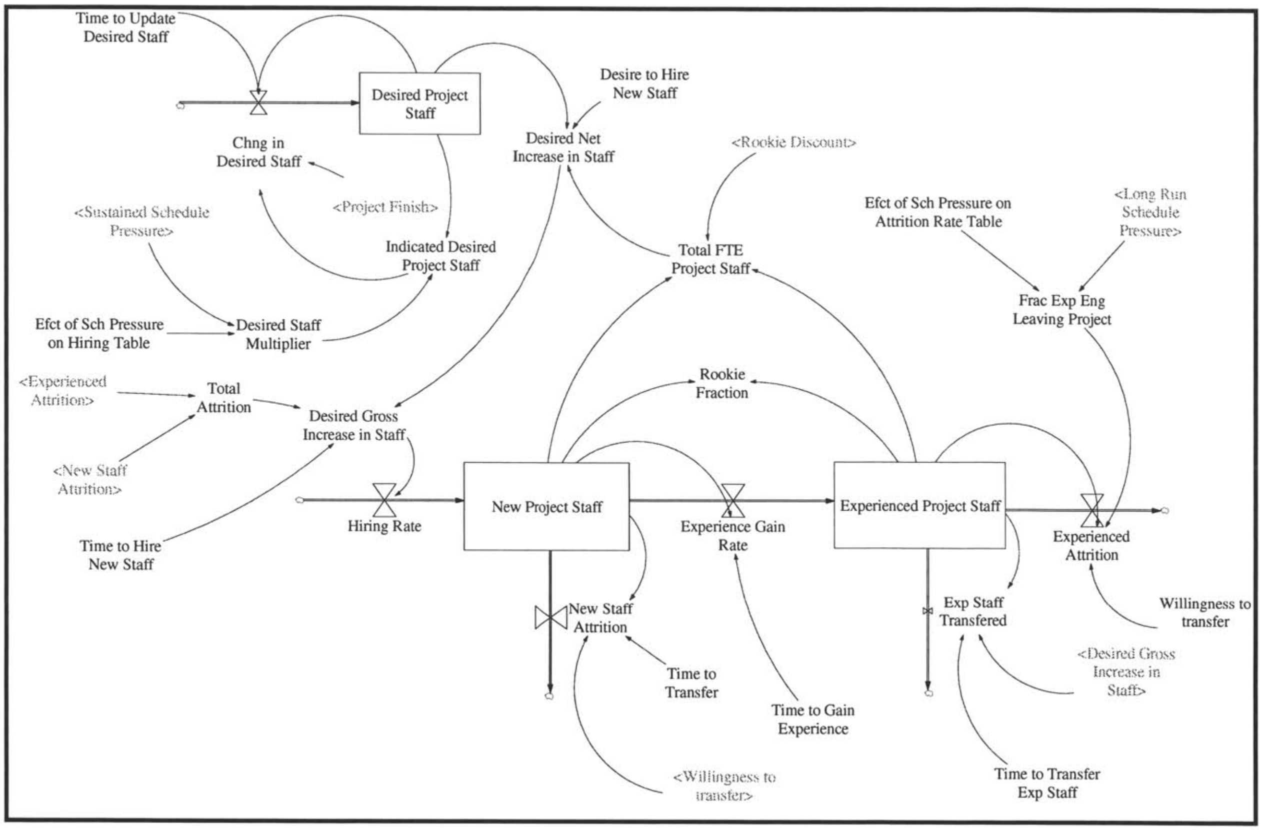

Figure 18. Staffing Structure of BP NPD MFS Model

The structure for the stock *Desired Project Staff* incorporates an informational flow *Chg in Desired Staff* which is equal to (*Indicated Desired Project Staff-Desired Project Staff*) divided by *Time to Update Desired Staff* (modeled as a constant 12 weeks to account for information delay of management realizing they have to hire new staff). Indicated Desired Project Staff is determined by a multiplier based on Schedule Pressure, a feedback effect that will be discussed in a later section. The auxiliary variables *Willingness to Transfer* and *Desire to Hire New Staff* are switches that turn on the hiring and attrition functions of the model and *Project Finish* is used to reflect when the project is finished. In the MFS, the Desired Project Staff structure is not used as hiring is input into the model through Game functions in the VENAPP, however the structure is included to verify that the hiring and firing dynamics are working in the NPD model.

Two additional variables are defined in the staffing structure that are of use in other structures of the model. *Rookie Fraction* is defined as the ratio of New Project Staff divided by (New Project Staff plus Experienced Project Staff) to represent the experience level of the total project staff. *Total FTE*[11] *Project Staff* is defined by New Project Staff times the *Rookie Discount* (a variable representing the productivity of a new employee compared to that of an experienced employee: set as a constant 0.5 in this model, a new employee will only be half as productive as an experienced employee) plus the Experienced Project Staff. Additional structure for staffing estimates to be used in the cost structure to be explained later is illustrated in Figure 19. Complete definitions of the variables in this structure can be found in the equation listing in Appendix A. Details on overtime structure will be covered in the section on resource allocation and productivity.

[11] FTE is an acronym for "Full Time Equivalent", a common term in characterizing staff levels in product development organizations.

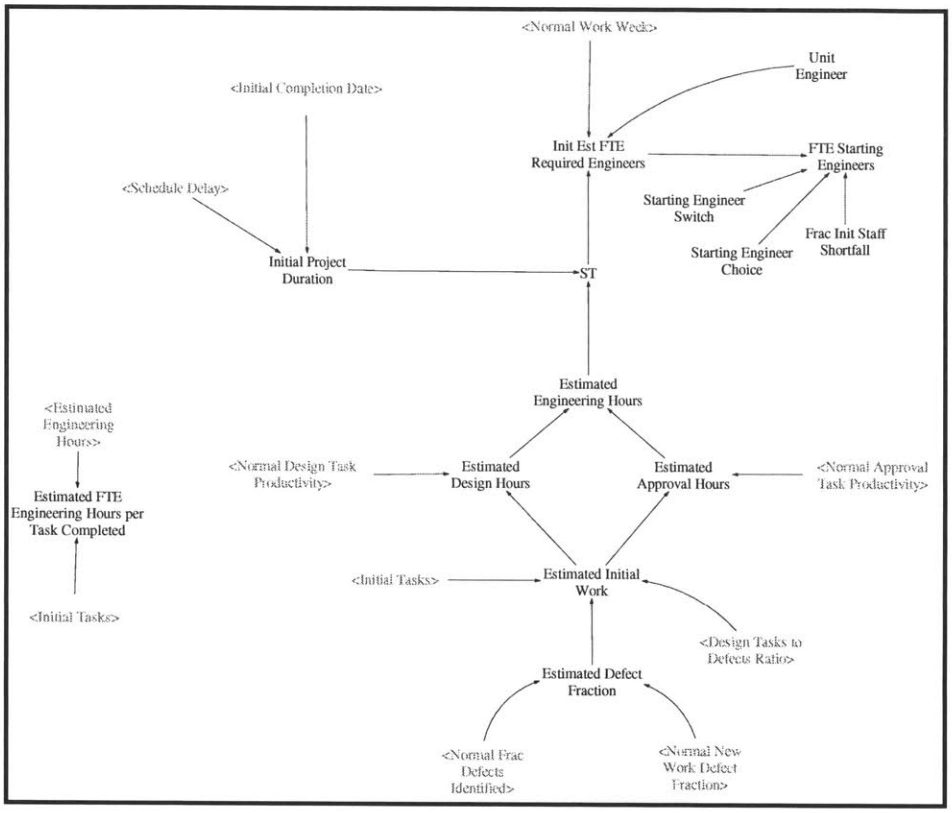

Figure 19. Staffing Estimate Definition of BP NPD MFS Model

4.1.4. Completion Date Structure

The model incorporates a single stock structure to determine project completion date that will be important in modeling project cost and the schedule pressure feedback effects that were described in chapter three as being so important to modeling project dynamics. The Completion Date Structure is shown in Figure 20. The stock *Calculated Completion date* is fed by the flow *Changes in Completion Date* which is equal to *Completion Date Gap* divided by *Adjustment Delay*. The Completion Date Gap is the difference between the *Anticipated Finish Date* and *Calculated Completion date. Willingness to Slip Deadline* is a switch that enables the flow Changes in Completion date to be active and allows schedule to slip to alleviate schedule pressure effects. *Anticipated Finish Date* is simply the time elapsed plus the estimated time to

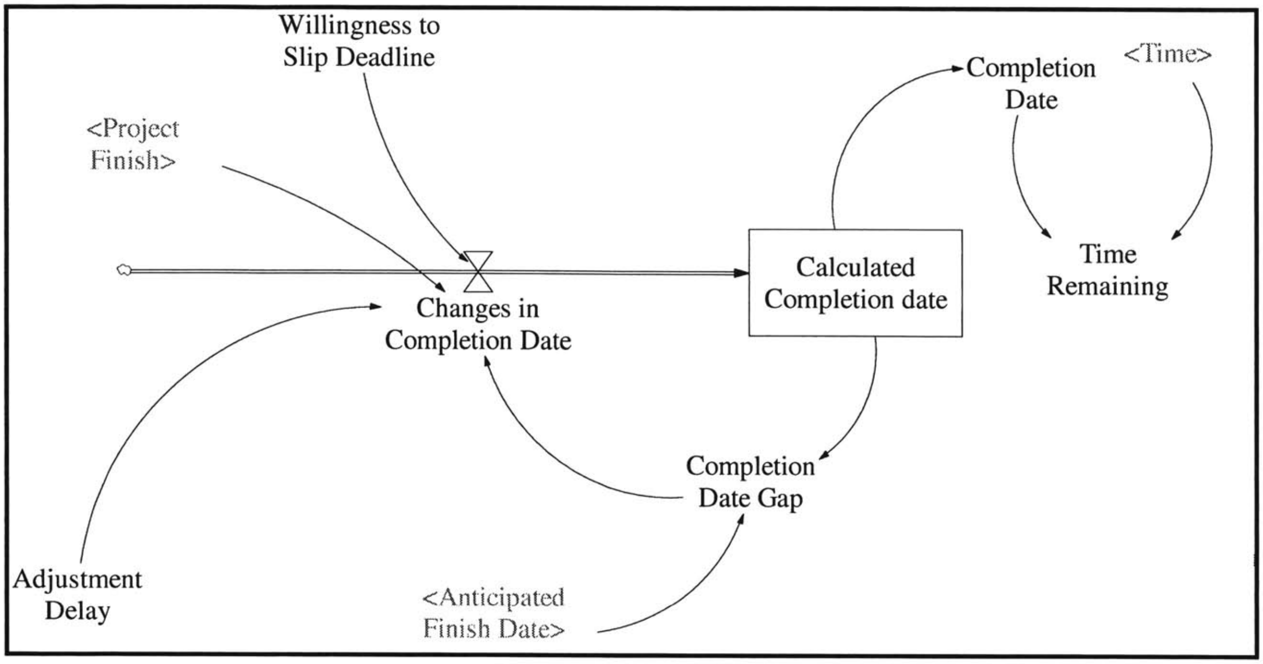

Figure 20. Completion Date Structure of the BP NPD MFS Model

complete the project based on current work accomplishment rates (to be discussed in section on schedule pressure). *Completion Date* is the value of the stock *Calculated Completion date* and is used to calculate *Time Remaining* by subtracting the value of *Time* from *Completion Date*. When the model is used for the MFS, *Completion Date* is input by the player of the MFS so this structure is not used, however it is useful in verifying that the model is accurately reflecting schedule slip dynamics.

4.1.5. Cost Structure

Because cost is a critical parameter used to manage new product development projects, a cost structure is included in the BP NPD MFS Model. The cost structure for the model is illustrated in Figure 21. The stock *Cumulative Cost of Project* is fed by the flow *Inc in Cum Cost*

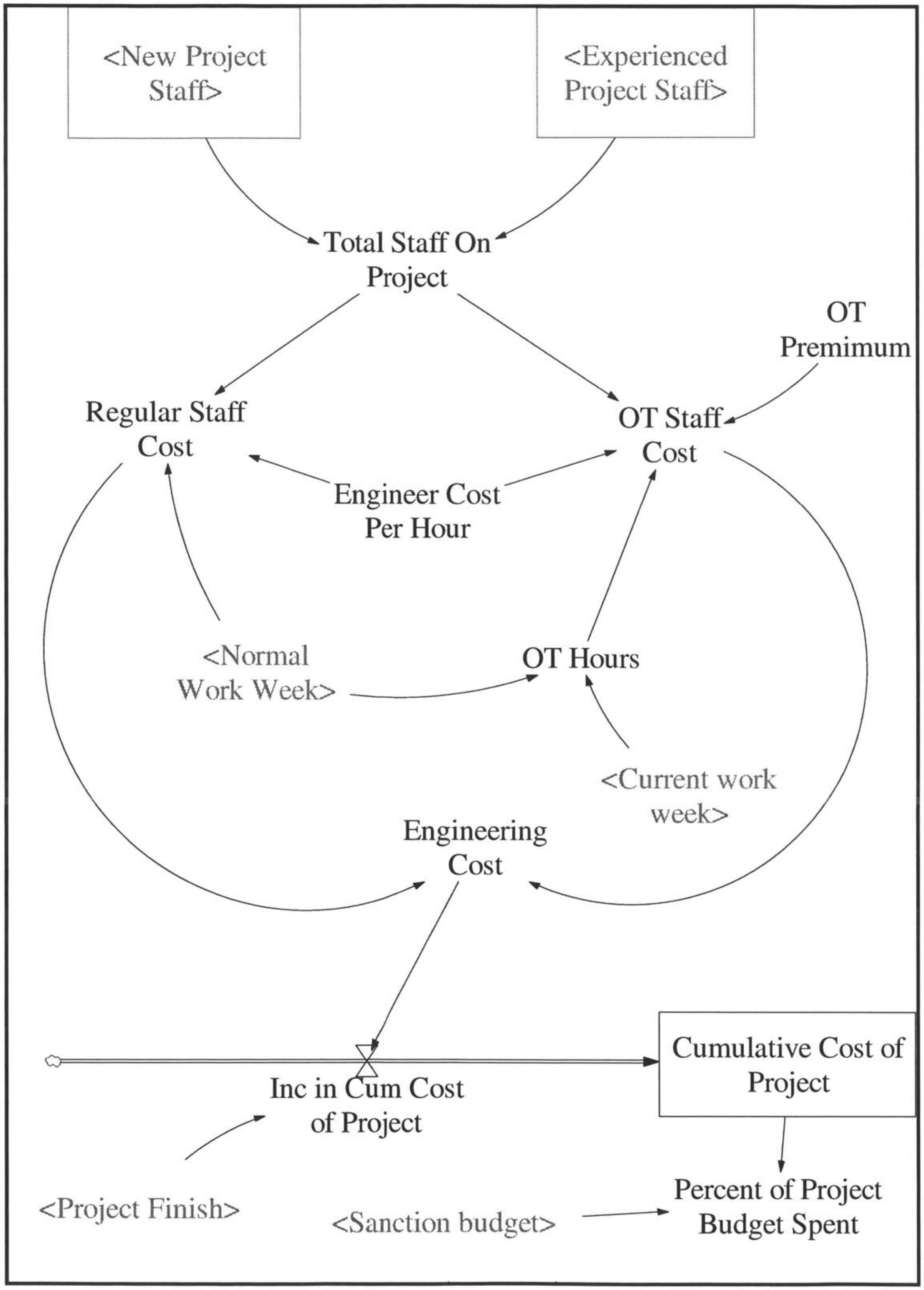

Figure 21. Cost Structure of BP NPD MFS Model

of Project which is equal to the *Engineering Cost* in units of $ dollars per week. Engineering Cost is defined by the sum of *Regular Staff Cost* and *OT Staff Cost. Regular Staff Cost* is defined by the *Total Staff on Project* (which is the sum of New Project Staff and Experienced Project Staff defined in section 4.1.3)[12] times the *Engineer Cost per Hour* (120$/hour for this model) times the *Normal Work Week* (set as a constant 40 hours per week in this model). *OT Staff Cost* is defined by *OT Hours* (Current Work Rate minus Normal Work Rate) times the *OT Premium* (fractional increase in overtime pay compared with normal pay – is set equal to 1 in this model since design staff in petroleum industry are not paid a higher rate for overtime). *Sanction Budget* is defined by *Scheduled Engineering Cost* ($/week) times *Initial Completion Date* (weeks-which is initialized in the MFS simulation to be discussed in Section 4.2). Scheduled Engineering Cost is determined by initial staff estimates that are calculated to start the simulation at a realistic staffing level determined by the structure shown in Figure 19. A complete equation listing is included in Appendix A.

4.1.6. Resource Allocation to Work Flow and Productivity Structure

Several of the flows identified in the work flow structure in section 4.1.1 were constrained by resource capacity at that flow. This section illustrates the model structure which determines the allocation of resources and productivity to the various flows in the work flow structure. Key variables that are used in other structures are highlighted in red in the following figures. The resource allocation structure of the model that determines the fractional amounts of total work allocated to rework, new work, and coordination is shown in Figure 22. The structure for determining the approval work flow resource capacity is shown in Figure 23. The structure

[12] Note that for cost purposes, actual staffing headcounts is used to estimate the total cost, where for productivity purposes, FTE staff is used which takes into account the Rookie Fraction where new personnel are not as productive as experienced personnel.

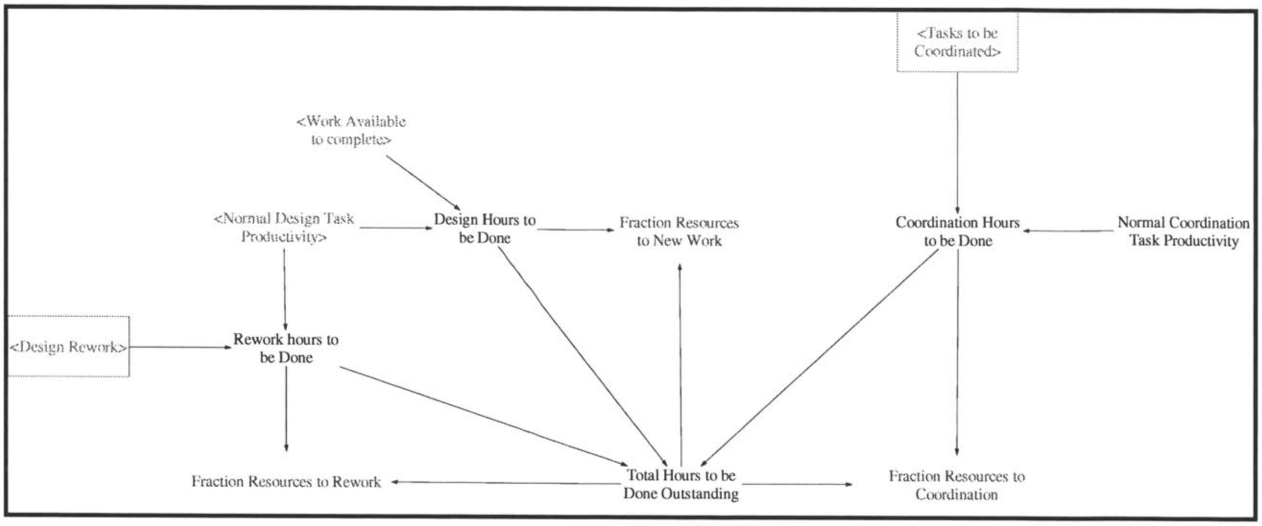

Figure 22. New Work, Rework, and Coordination Resource Allocation Structure

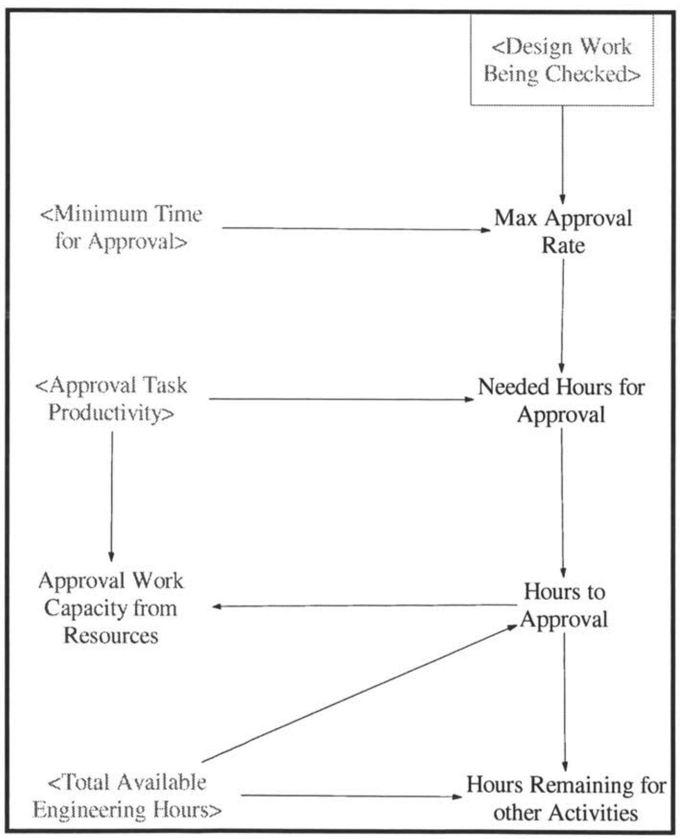

Figure 23. Approval Work Capacity Structure

The structure for determining resource capacity for rework, new work, and coordination is illustrated in Figure 24. The use of overtime to augment productivity is modeled in Figure 25.

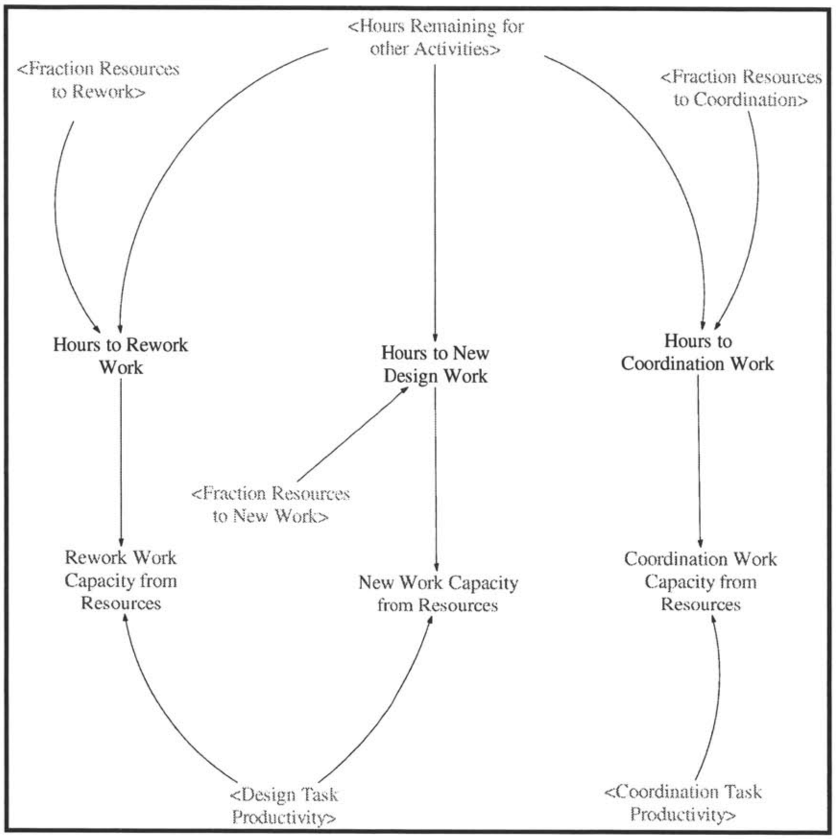

Figure 24. Resource Capacity for Rework, New Work, and Coordination Structure

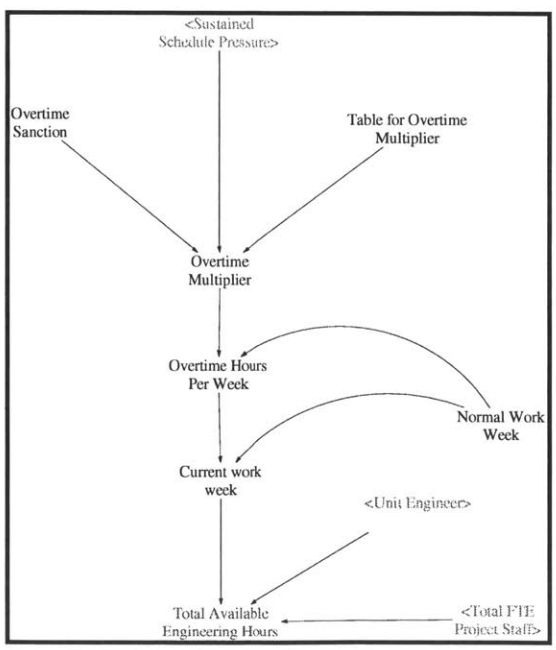

Figure 25 Overtime Structure

Overtime Hours per Week is a variable representing overtime hours to extend the work week to increase task accomplishment that is calculated by the model through a table function based on *Sustained Schedule Pressure* (shown in Figure 26). As sustained schedule pressure increases, overtime is increased to quicken work accomplishment to achieve more tasks, which results in an increase in the variable *Current Work Week* which is compared to *Normal Work Week* to create a normalized parameter *Task per Week Ratio* that is the overtime parameter used to address feedback effects based on overtime. In the MFS, overtime is an inputted decision variable so the overtime structure to increase overtime based on schedule pressure is not used; however, it is incorporated in the model to confirm the overtime feedback effects when verifying the model.

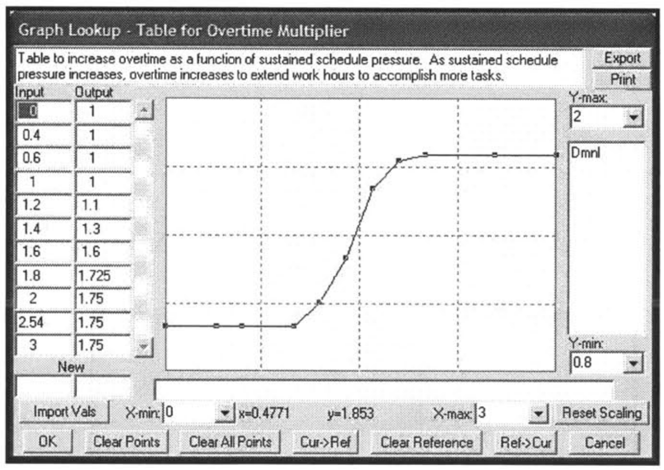

Figure 26. Table Function to Increase Overtime as Sustained Schedule Function Increases

The structure showing productivity based upon resource capacities and effects of schedule pressure and fatigue to be discussed in the feedback effects sections of this chapter is illustrated in Figure 27. A complete equation listing for the resource allocation, capacity and productivity structure is included in Appendix A.

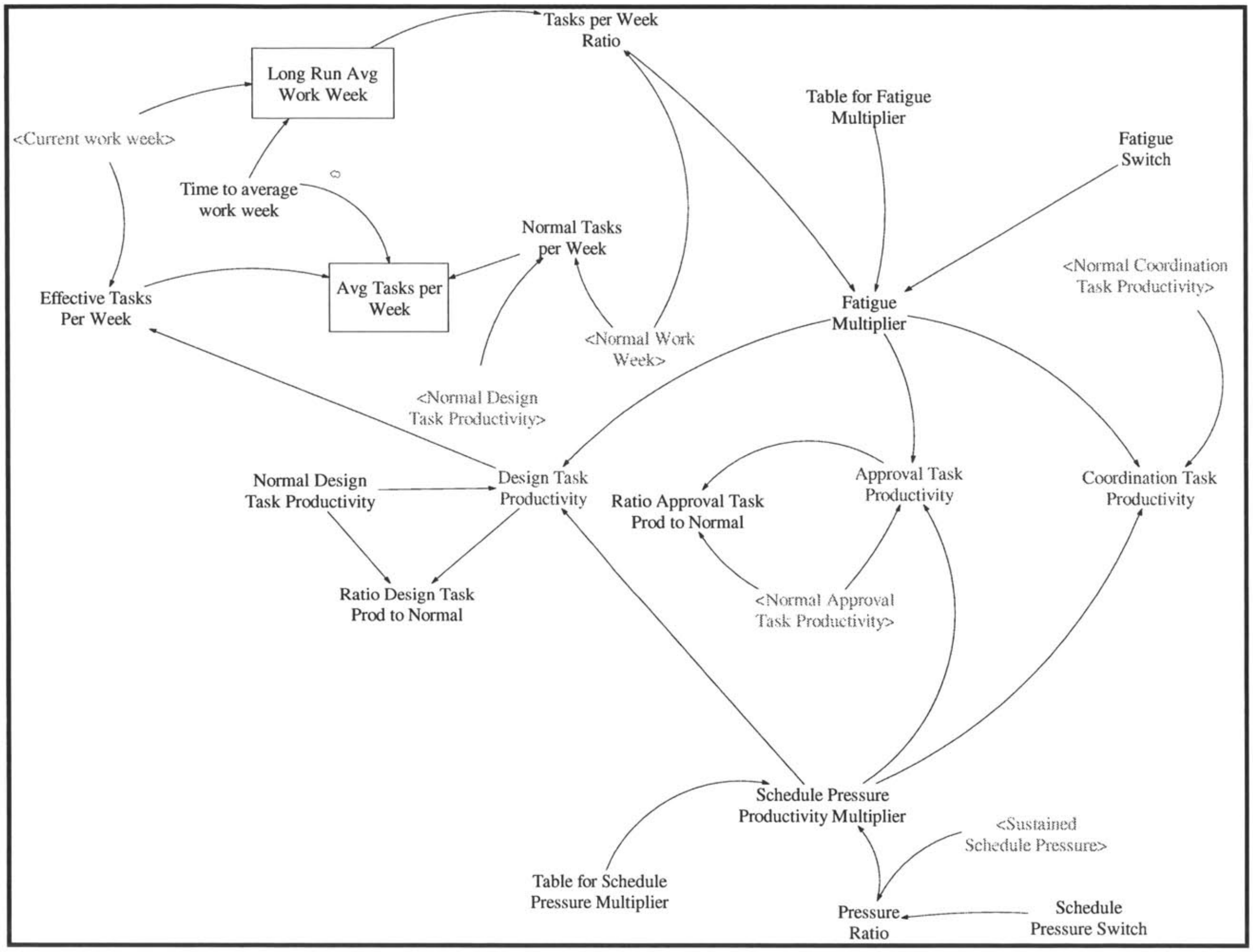

Figure 27. Design, Approval, and Coordination Task Productivity

4.1.7. Schedule Pressure Structure

Many of the feedback effects on productivity and quality that were described in section 3.1.2 are initiated by schedule pressure. The BP NPD MFS model incorporates a structure to calculate schedule pressure to model these effects. Schedule pressure is an informational variable that calculates the relative 'lateness' that the project will have at the end of the project based on The feedback effects of schedule pressure on productivity and quality will be described in later sections, but a description of the schedule pressure structure is presented here. The basic structure for the accumulation of schedule pressure is shown in Figure 28.

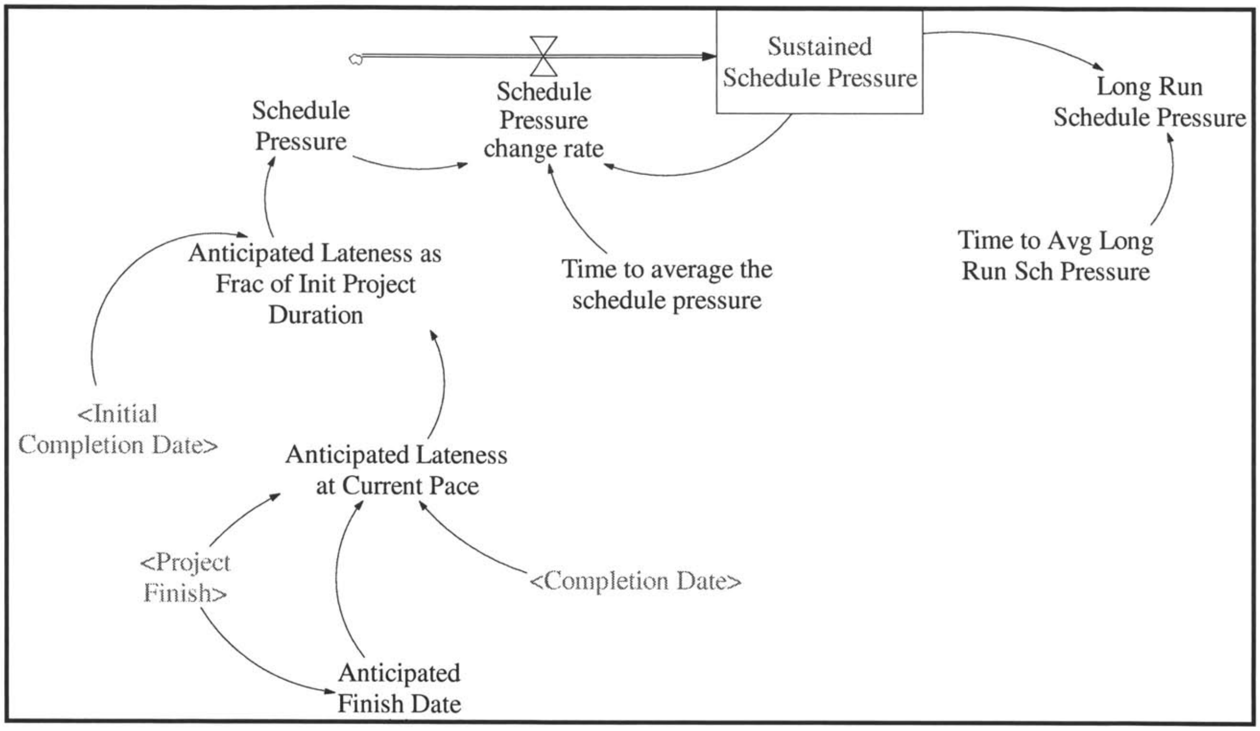

Figure 28. Schedule Pressure Structure of BP NPD MFS Model

Sustained Schedule Pressure is modeled as it is the experience of the author and the other two modelers that level of schedule pressure based on project lateness grows the further the project goes along. Therefore, while initial project progress (up to around the first quarter of the total project durations) may indicate that it will be very late, because the project has just started, there is not the same level of schedule pressure that would be present if the same lateness came during the last quarter of the project. The stock *Sustained Schedule Pressure* is fed by the flow *Schedule Pressure change rate* which is equal to *Schedule Pressure* divided by *Time to average the schedule pressure* (which is set as a constant 4 weeks in this model to account for the information delay in accumulating schedule pressure). *Schedule Pressure* is developed as a function of the initial schedule pressure and the emergent pressure during the project and is defined as 1 plus *Anticipated Lateness as Frac of Init Project Duration. Anticipated Lateness as Frac of Init Project Duration* represents the fraction late as a ratio to initial project duration and is defined by *Anticipated Lateness at Current Pace* divided by the *Initial Completion Date. Anticipated Lateness at Current Pace* represents the weeks late the project will finish and is defined by the *Anticipated Finish Date* minus the completion date (Project Finish is included as an indicator to set value to zero when the project is completed). The structure defining *Anticipated Finish Date* is shown in Figure 29.

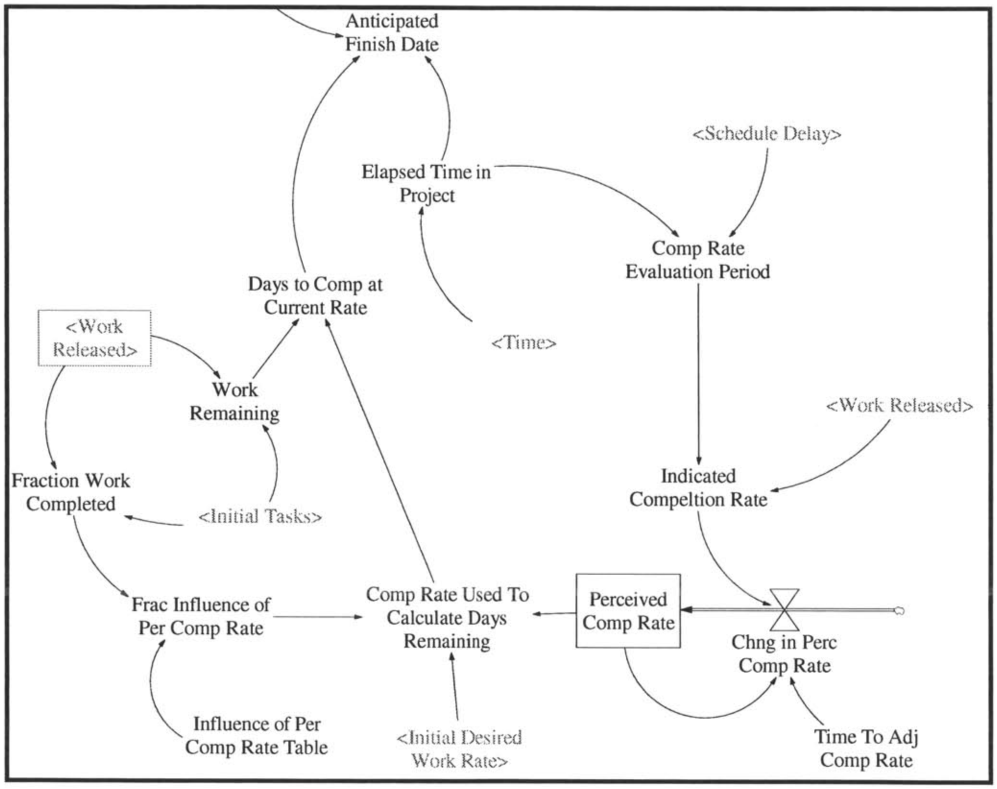

Figure 29. Structure to Determine Anticipated Completion Date

The stock *Perceived Comp Rate* accounts for changes in managements expectations of project completion rate as the project progresses, i.e., the completion rate as perceived over time. It accounts for management re-calculating the rate at which work is actually done in comparison with the initially projected work rate. As the project progresses, more attention is given to *Indicated Completion Rate* and less on the initially expected (*Initial Desired Work Rate).* This non linear relationship between project completion and perceived completion rate is illustrated in the table function *Influence of Per Comp Rate Table* shown in Figure 30. Additional structure related to schedule pressure is shown in Figure 31 which shows the definition of Initial Desired Work Rate and other variables that will be displayed in the MFS to indicate schedule progress.

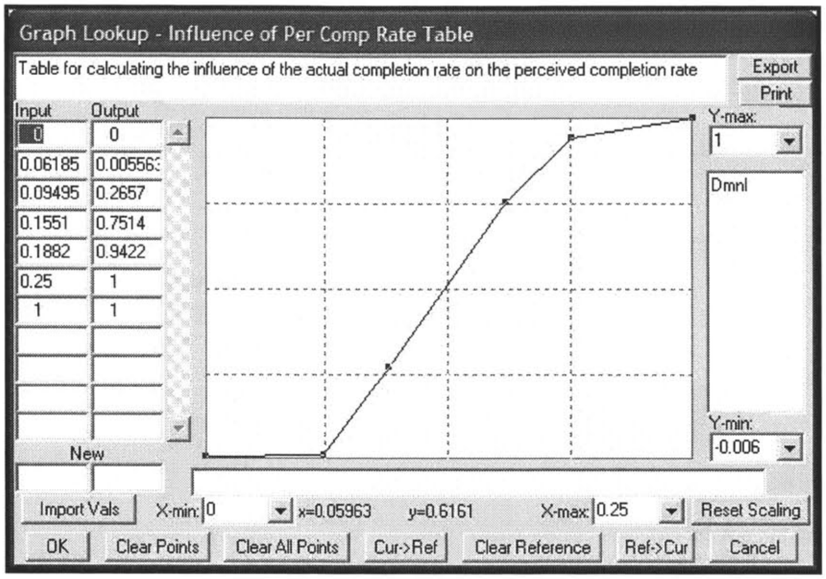

Figure 30. Influence of Project Completion on Perceived Completion Rate Table Function

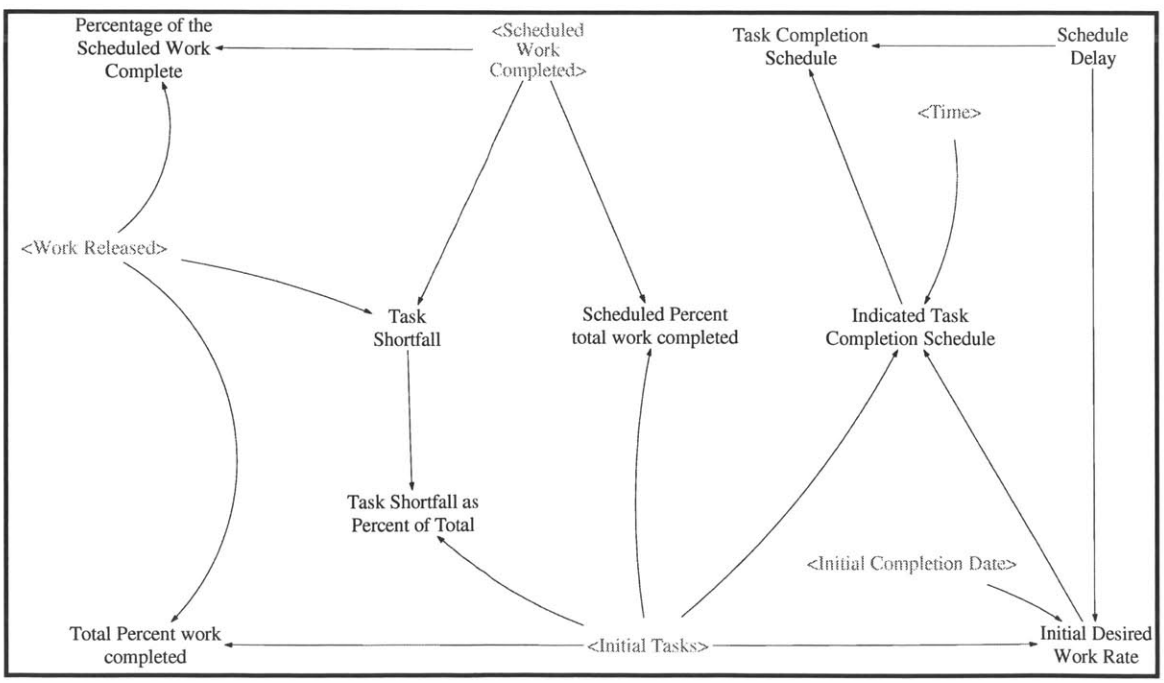

Figure 31. Additional Schedule Pressure Definition

A simple two stock structure that calculates *Scheduled Completion Date* is shown in Figure 32. Scheduled Completion Date is important as many new product development project management techniques use metrics based on progress compared to scheduled completion date, as will be seen in the section that discusses the Earned Value Management System (EVMS) structure.

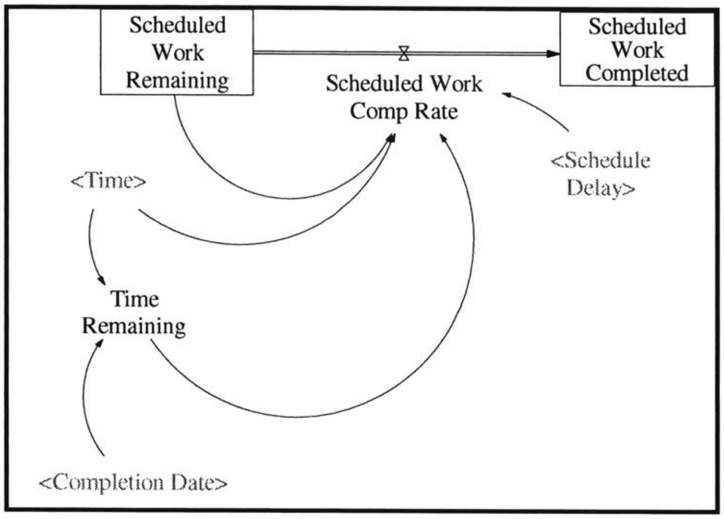

Figure 32. Scheduled Work Completed Structure

4.1.8. Earned Value Management System Reporting Structure[13],[14]

Because there is an increasing trend in the management of new product development projects to use the Earned Value Management System (EVMS), structure in the model defining EVMS variables are included. EVMS has been a standard objective method of cost and schedule performance mandated by Department of Defense (DoD) contracts for over a decade, and is being adopted by more and more commercial product development organizations due to the systematic and objective way it addresses cost and schedule performance. EVMS allows project managers to determine if a project is progressing per scheduled rather than just by task

[13] Information on EVMS included in this section comes from a pocket EVMS reference card provided by the Acquisition Management Institute, 7432 Alban Station Boulevard, Suite B252, Springfield, VA 22150 (703) 440-5000.

[14] See also: Q. Flemming and J. Koppelman, "Earned Value: Project Management", Project Management Institute – September 2000

accomplishment and budget expended. By comparing the progress to what was originally *scheduled* to be spent and accomplished it can give managers an indication of whether the project is ahead or behind schedule, not necessarily just against what it was planned and/or budgeted for. For example, a project may have planned to spend a quarter of its budget by a certain date. With non EVMS methods, a project that was showing that it had spent a quarter of its budget by that date would be seen as being 'on budget', without recognizing whether it had accomplished what it was supposed to have by that time, i.e., the "value earned'. EVMS provides a set of metrics that can give visibility to management on whether the project is on schedule and/or budget or not. While EVMS is not being used by BP, BP desired to have the EVMS system included in the MFS.

A list of key EVMS variables and their definitions are provided in Table 1. Project status based on EVMS is frequently displayed by a graph of Schedule Performance Index (SPI) versus Cost Performance Index commonly known as the "Bulls-eye Chart". An example of a bulls-eye chart is shown in Figure 33. Managers in NPD organizations that use EVMS frequently refer to the CPI and SPI of a given project's status both at an aggregate level (for the entire program, a 'phase' of the program, or at a module level, i.e., the compressor group's project status for an engine development program) and also for individual project tasks in a project. Because BP does not use EVMS, only the CPI, SPI, and the bulls-eye chart were incorporated in the MFS.

Earned Value Management System (EVMS) Terminology		
Term	**Name**	**Definition**
BCWS	Budgeted Cost of Work Scheduled	The $ amount of budget for tasks **scheduled** up to current date
BCWP	Budgeted Cost of Work Performed	The $ amount of budget for tasks **performed** up to current date
ACWP	Actual Cost of Work Performed	The $ amount actually spent for tasks **performed** up to current date
MR	Management Reserve	The $ amount above BAC that adds up to Target Cost - additional budget held in reserves by management to account for unexpected problems
EAC	Estimate at Completion	The total $ amount expected to be spent by the project at project completion based on progress to date and scheduled tasks remaining until completion. It is calculated by the either of the following expressions EAC = BAC/CPI(Cum) EAC = ACWP + (BAC-BCWP)/(CPI*SPI) EAC = ACWP + (BAC - BCWP)/(.8CPI + .2SPI)
TC	Target Cost	The $ amount not to be exceeded at project completion, also called the "Budget Base".
BAC	Budget at Completion	The $ amount budgeted for completion of the project if everything went "as planned" . Does not change unless PMB is 're-baselined'
PMB	Performance Measurement Baseline	The $ amount vs. Time relationship for spending of the project per scheduled if everything goes "as planned". Does not change unless 're-baselined' by management.
CPI	Cost Performance Index (Efficiency)	A measure of **cost** management **efficiency**. It is equal to BCWP/ACWP. It is a measure of how cost-effectively tasks were accomplished as performed regardless of whether they were performed as scheduled. A project is "Green" if CPI is > 0.95, "Yellow" if 0.90 < CPI < 0.95, and "Red" if CPI < 0.90
SPI	Schedule Performance Index (Efficiency)	A measure of schedule management efficiency. It is equal to BCWP/BCWS. It is a measure of how much work was actually accomplished compared to how much work was supposed to be accomplished to date, regardless of how much was actually spent accomplishing the work. A project is "Green" if CPI is > 0.95, "Yellow" if 0.90 < CPI < 0.95, and "Red" if CPI < 0.90
CV	Cost Variance	The difference between the budgeted cost of work performed and the actual cost of work performed (BCWP - ACWP). Positive CV means CPI is greater than 1.0 and the project team is spending less than planned for work accomplished (cost effective). Negative CV is the amount the project has 'over spent' on tasks to date.
SV	Schedule Variance	The difference between budgeted cost of work performed and budgeted cost of work scheduled (BCWP - BCWS). Positive SV means that SPI is greater than 1.0 and the project team has accomplishing more than originally scheduled. A negative SV indicates the amount a project is behind schedule expressed as the $ amount of work it did not accomplish that it was scheduled to accomplish

Table 1. Terminology Used in Earned Value Management System (EVMS)

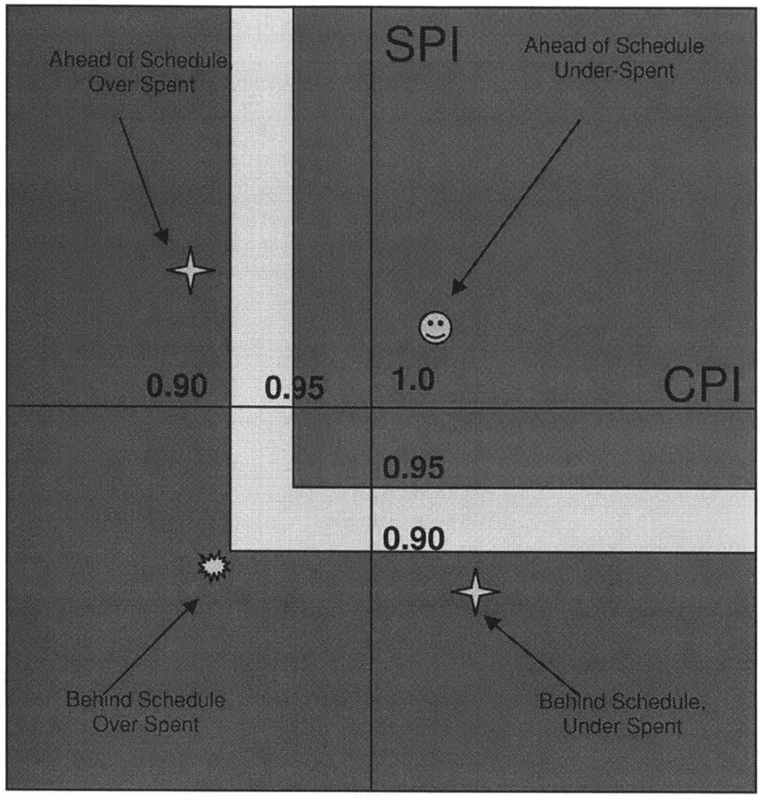

Figure 33. Example of SPI vs. SPI "Bulls-Eye" Chart Used in EVMS

The structure used in the BP NPD MFS model to 'report' CPI and SPI is shown in Figure 34 and incorporates the same equations to calculate the necessary parameters as is shown in Table 1.

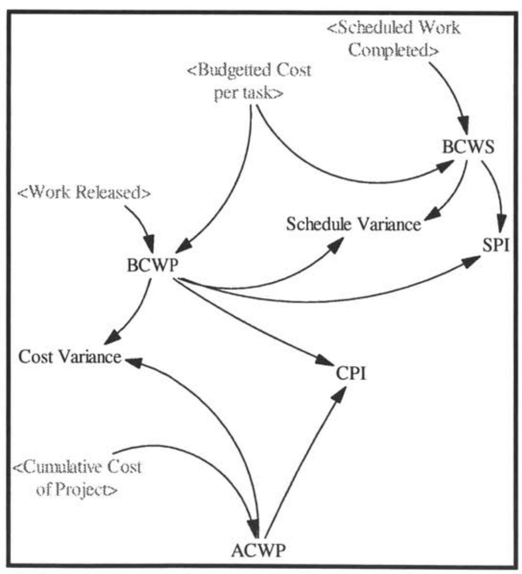

Figure 34. SPI and CPI Definition in BP NPD FMS Model

4.1.9. Modeling Feedback Effects on Productivity and Quality

Now that the key work flow, defect, resource, cost, and schedule pressure structures that are necessary for the rework cycle and Ford-Sterman modeling of NPD Projects have been established, this section describes how these structures are related to model the vicious project dynamics that drive poor NPD performance. As described in section 3.1.2, these effects are usually overlooked by managers and it is the intent of the MFS to simulate these effects. The primary feedback effects are reduced productivity and quality attributed to experience level as indicated by a high rookie fraction, and reduced productivity and quality attributed to work accomplishment 'speed' and fatigue that occurs as the result of schedule pressure. A summary of how these feedback effects are introduced into the Defect Structure is shown in Figure 35.

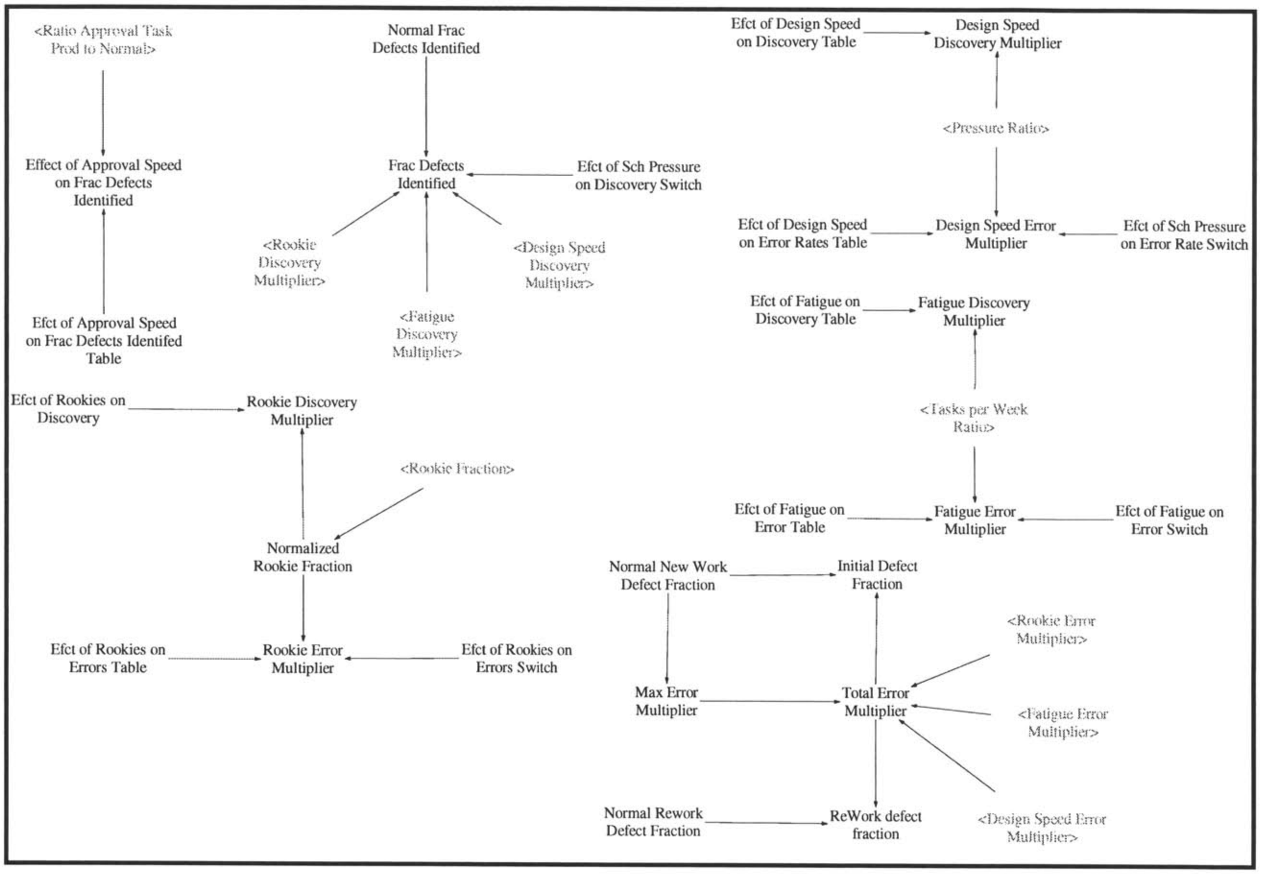

Figure 35 Feedback Effects on Defect and Defect Discovery

Effects on quality impact the defect structure not only through generation of defects but also through the *discovery of defects* (recall from section 3.1.2 the importance of rework discovery to the dynamics of projects). The effect of inexperience on defect generation is introduced into the model through the *Total Error Multiplier* by the *Rookie Error Multiplier* which utilizes a table function called *Efct of Rookies on Errors Table* (shown in Figure 36). As the *Rookie Fraction* increases, a higher rookie multiplier increases the total number of errors. Inexperience also leads to fewer discoveries of defects as new personnel are not knowledgeable enough to know defects are present until they have gained experience.

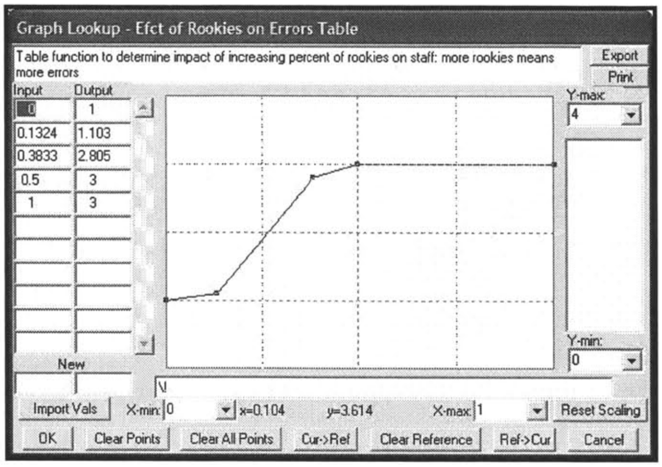

Figure 36. Effect of Inexperience (Rookie Fraction) on Errors Generated Table Function

The effect of inexperience on rework discovery is introduced per Figure 35 through *Frac Defects Identified* by the *Rookie Discovery Multiplier* which utilizes a table function called *Efct of Rookies on Discovery* (shown in Figure 37.). The feedback effect of inexperience on productivity was already described by definition of the *Rookie Discount* which discounted a new employee's productivity in section 4.1.3.

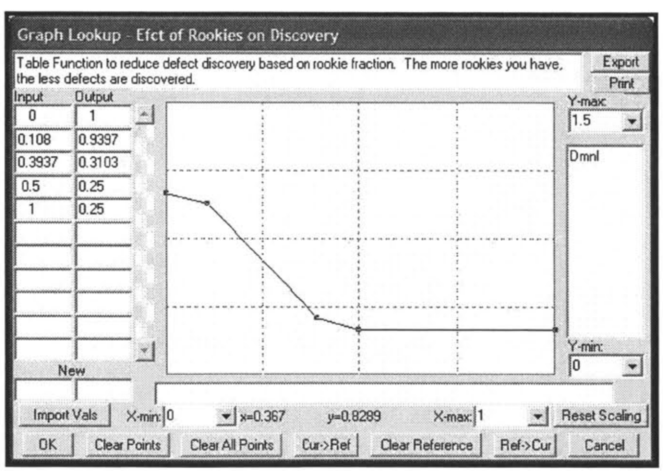

Figure 37. Effect of Inexperience (Rookie Fraction) on Defect Discovery Table Function

The feedback effects of schedule pressure and overtime are likewise introduced to defect generation discovery through similar table functions. Schedule pressure impacts quality *speed* (to simulate the "cutting corners" dynamic described in section 3.1.2) and overtime impacts quality through *fatigue*. The effect of schedule pressure manifested as speed is modeled through the table function *Efct of Design Speed on Error Rates Table* (shown in Figure 38). The faster the team works due to schedule pressure, the more errors are made. Likewise, the faster the team works, the fewer errors are *discovered* which is captured in the table function *Efct of Design Speed on Discovery Table* (shown in Figure 39). Similar effects that overtime have on defect generation and discovery are shown in Figures 40 and 41, respectively.

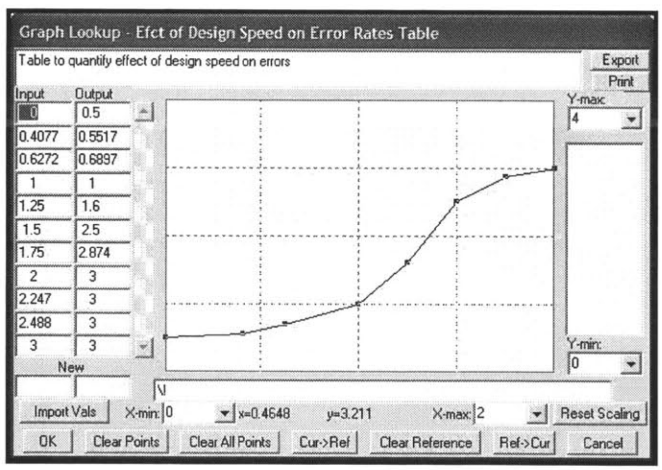

Figure 38. Effect of Design Speed (due to Schedule Pressure) on Error Rates Table

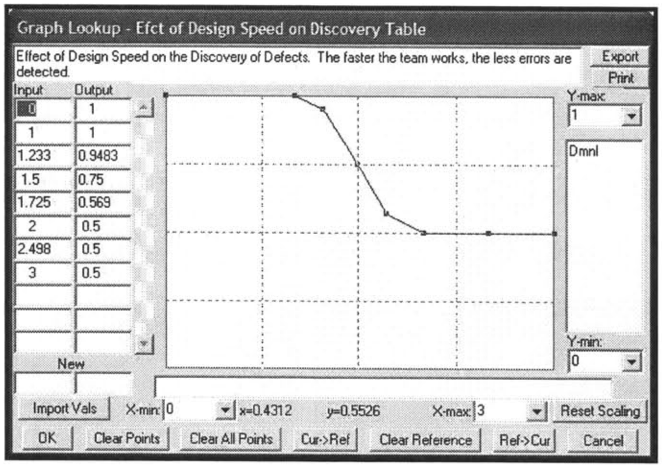

Figure 39. Effect of Design Speed (due to Schedule Pressure) on Error Discovery Table Function

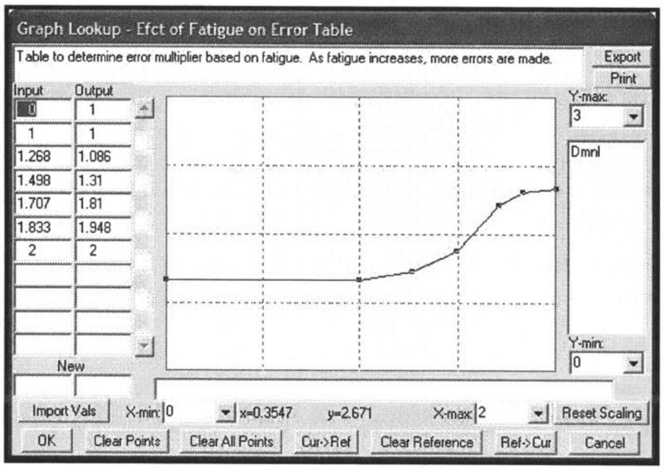

Figure 40. Effect of Overtime (Tasks per Week Ratio) on Error Generation Table Function

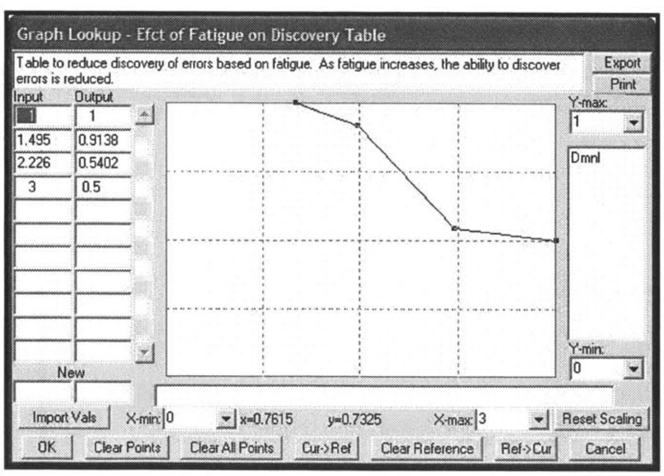

Figure 41 Effect of Overtime (Tasks per Week Ratio) on Error Generation Table Function

There are additional feedback effects in the structure of the model. Schedule pressure also affects attrition as shown in Figure 15. The table function *Efct of Sch Pressure on Attrition Rate Table* (shown in Figure 42) which is a function of *Long Run Schedule Pressure* described in section 4.1.7 and shown in Figure 25. Schedule Pressure also impacts productivity as engineers

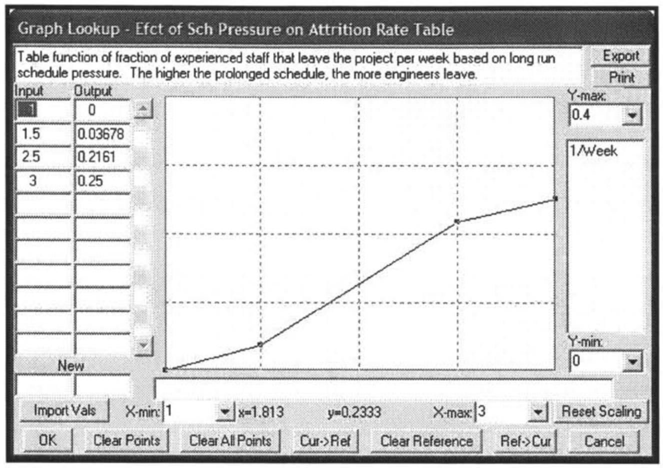

Figure 42 Effect of Long Run Schedule Pressure on Experienced Staff Attrition Table Function

who are behind tend to work harder to catch up. The multiplier on productivity based on schedule pressure is incorporated in the model through the table function *Table for Schedule Pressure Multiplier* (shown in Figure 43) but is based on *Sustained Schedule Pressure.*

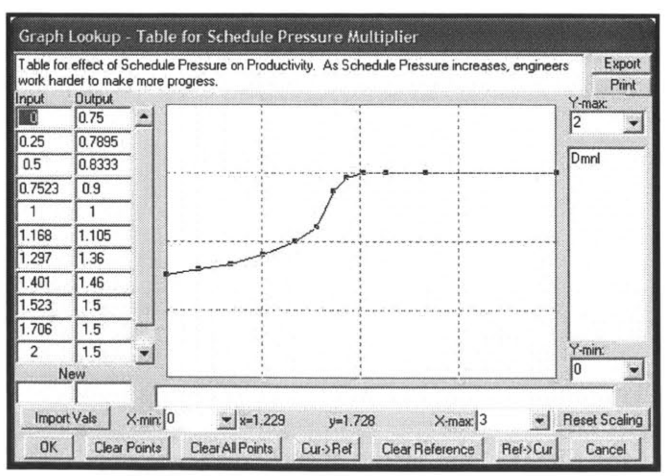

Figure 43. Effect of Sustained Schedule Pressure on Productivity Table Function

The effects of fatigue from overtime also impact productivity. As engineers work longer hours (as reflected in *Tasks per Week Ratio*), they tend to be less productive from burn-out. This relationship is incorporated in the table function *Table for Fatigue Multiplier* shown in figure 44. As fatigue increases (measured by *Task per Week ratio,* defined as *Long Run Avg Work Week/Normal Work Week*), *Design Task Productivity, Approval Task Productivity*, and *Coordination Task Productivity* is reduced by the *Fatigue Multiplier*.

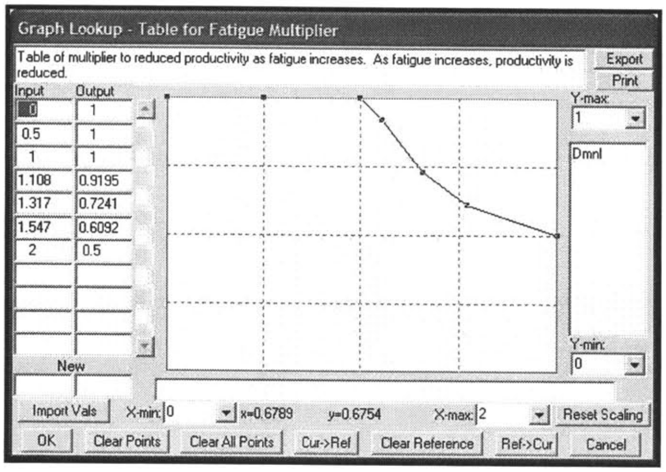

Figure 44. Effect of Fatigue on Productivity Table Function

The feedback effects described in this section were based on common-sense judgment from the modelers from product development experience and effects that have been documented in the literature. They are not intended to establish the accuracy required of a calibrated system dynamics model, but rather to capture the essential project dynamics described in Chapter 3 so as to be used for the MFS. A section on validation of the model is not included in this thesis, although the model was continually tested while being validated to ensure the correct project dynamics were capture. An illustration of the model's results for a simple project are shown in Figure 45 and 46 which show the effects of different feedback features turned on to verify that the feedback effect structures were working as intended.

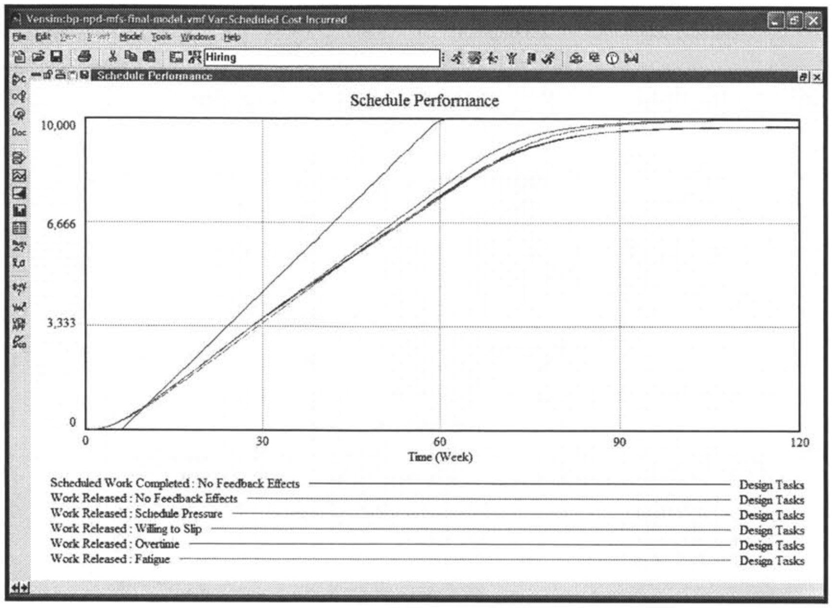

Figure 45. Schedule Performance with Different Feedback Effects Activated

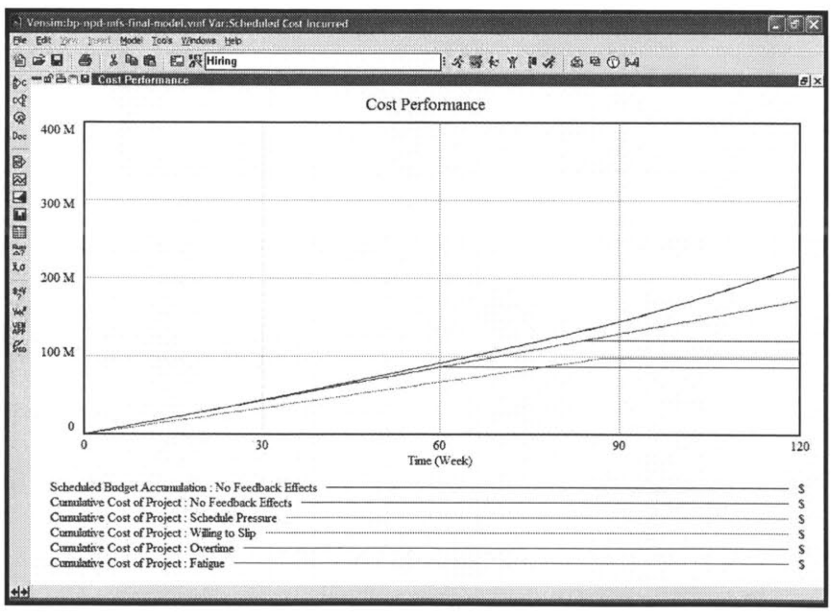

Figure 46. Cost Performance with Different Feedback Effects Activated

4.2 Development of Management Flight Simulator (MFS) for New Product Development (NPD)

Now that the project dynamics of new product development projects have been understood and modeled with a system dynamics model, the next step in meeting the thesis objectives is to develop the management flight simulator (MFS) itself. This section describes the product development (PD) process used to develop the MFS. The structure of the MFS is described showing features for each screen in the MFS.

4.2.1. Principles and Requirements of BP NPD MFS Learning System[15]

The BP NPD MFS was developed to enhance the learning of system dynamics principles that when applied to product development projects can improve the cost and schedule performance of these projects. These principles were described in Chapter 3. *Playing* or *doing* is an established principle of learning as shown in Chapter 2. The principle of improving learning by playing or doing project management is the central principle of this MFS, which will be referred to as a 'game' for the remainder of this thesis, and the participants will be referred to as the 'players'. The most effective way shown to instill a change in mental models is to have the players play the game with their existing decision making policies and see the results of their decisions (Repenning, 2003 and Sterman 2003). This is the second principle that the MFS is based on. The third principle is that the game must portray behavior that is similar to that seen in real life so that the players will believe the results of the simulation. The game scenario must also be similar to what the managers' are familiar with so that the scenario is 'relevant' to their environment.

[15] The information in this section was obtained primarily from correspondence with the BP Project Academy contact who sponsored this development work, Scott Johnson.

BP Project Academy was also involved in the development of the MFS. Their requirements were defined as:

- Teach how causal loop diagramming (CLD) can be used to reframe project issues
- Provide a safe environment through the use of Management Flight Simulators to deepen their knowledge and skill.
- Explain how the participants can best utilize the Engineering Projects Technology Group (EPTG) Project Excellent Team (PET) to deepen their knowledge and skill.
- Explain how the participants can best utilize the EPTG PET system dynamics resources when they return to their projects.
- Introduce PA Project Academy participants to important project dynamics lessons through the use of a MFS consisting of a model, user interface, and supporting process.
- Have a series of MFSs o address a range of lessons
    - Develop and validate a plausible story line with knowledgeable BP personnel
    - Develop and validate a VENSIM model with knowledgeable BP personnel
    - Select user interface technology to create the MFS
    - Create a MFS by combining the model with user interface technology
    - Develop and test-run a teaching approach with knowledgeable BP personnel
- Include a presentation of BP case studies by EPTG PET staff to link curriculum with real work and introduce potential resources.
- Participants should identify and recognize business issues and ways to apply principles to their projects.
- Desire for Web-based accessibility

    - BP PA participants can return to their home-base and immediately access the MFSs to influence their projects.
    - Nothing to download – avoids installation time and errors (unexpected software conflicts).
    - Less version control effort – don't have to send updates to multiple users
    - Broadcast tracking features associated with web-based applications gather data on how people actually interact with the MFS to provide feedback knowledge for continuous improvement.
    - Avoids "Video Game" phenomenon: If users know that every click is being recorded they will be less likely to simply start pressing buttons without taking accountability for their decisions – *extremely important for learning* (emphasis added by author).
- Incorporate features relevant to BP management:
    - Use appropriate product development phase
    - Use relevant information that a BP project manager reviews often.
    - Recognize distinction between BP employees and contractors and address relationship appropriately in MFS.
- Ensure content is linked to "No Wrecks".
- Integrate significance of "Front End Launch" (FEL) activities into the mental models of BP project managers.
    - Teach that setting up projects correctly in FEL activities is critical to the success of the project.

    - Introduce important FEL activities that take BP project managers out of their "comfort zone" of managing later stages of a project that they are comfortable with.
    - Incorporate model structure that allows BP Project Managers to understand how FEL decisions impact performance in later stages of a project.
- Create an appropriate system dynamics model and simulation story line that incorporates the following:
    - Classic project management issues relating to the effects of schedule pressure, overtime, new hires and skill distribution, rework/testing resources, scope creep, etc.
    - Risk management / uncertainty reduction and dealing with internal changes.
    - Issues around management of new technology.
    - Tradeoffs associated with allocation and focus of project management time to deal with problem of arbitration and team integration issues.
    - How Appraise phase (FEL work) impacts project performance through downstream effects.
    - Strategies for allocating resources for technical analysis, rework (changes), integration.
- Integrate FEL work and all other work to recognize significance of FEL decisions and impacts on downstream phases.
- Curriculum should focus the participants on how value is created and destroyed in projects by incorporating a "full life-cycle" perspective: FEL through 5 years operations

- Include D-cost ($/BOE), Start-Up Efficiency (SUE), Schedule, quality, etc.

As pointed out in Chapter 2, a MFS is only effective when used as a part of an overall *learning system.* The intent of this thesis is to develop the MFS only; the entire system is being developed through an ongoing project to support the BP Project Academy through collaboration between BP and MIT. Therefore, not all the requirements specified by BP are included in this version of the MFS. A schematic of the entire learning system represented in the Objective -> Goal -> Function format learned in System Engineering and System Architecture is shown in Figure 47. The development of the training concepts for the Investigate and Transfer goals

OBJECTIVE: Teach Principles of Project Dynamics to Improve Project Performance

GOAL:	**INVESTIGATE**	**CONCEPTUALIZE & FRAME**	**TRANSFER**
FUNCTION:	Develop Strategy	Test Strategies	Reflect on Results
	Clear Expectations	Experiment	Understand Why
	Learn Fundamentals	Learn by “Doing”	Apply to Other Scenarios
	Predict Outcomes	Play	Seek Guidance
CONCEPT:	Strategy Sheets	Management Flight Simulator	Debrief Players on Results
	Behavior Predictions	Use Decisions to Test Strategies	Show applicability to other scenarios
		Monitor Project Progress	

Figure 47. Objective, Goals, Functions, and Concepts of BP NPD MFS Learning System

of the BP NPD MFS learning system are not discussed here, other than to say that a strategy sheet was developed for the *investigate* stage for players to develop a strategy before playing the MFS, and debrief material was prepared for the *transfer* stage of the learning system.

4.2.2. Requirements for the MFS

The requirements listed in section 4.2.1. were tailored to the development of the first MFS for BP. This first MFS is intended to be expanded upon to incorporate more lessons in later terms of the BP Project Academy. Additional requirements were added to those in 4.2.1 based on suggestions from the literature and project management experience from the author. A list of requirements specific for this initial MFS is shown in Table 2. The remaining elements for the overall learning system (i.e., strategy sheets, debrief session, etc. are not included in this thesis).

4.2.3. Architecture of the MFS

The architecture of the MFS was adapted from suggestions provided by Ventana Systems, Inc. in their Vensim DSS Reference Supplement. Figure 48 shows the architecture

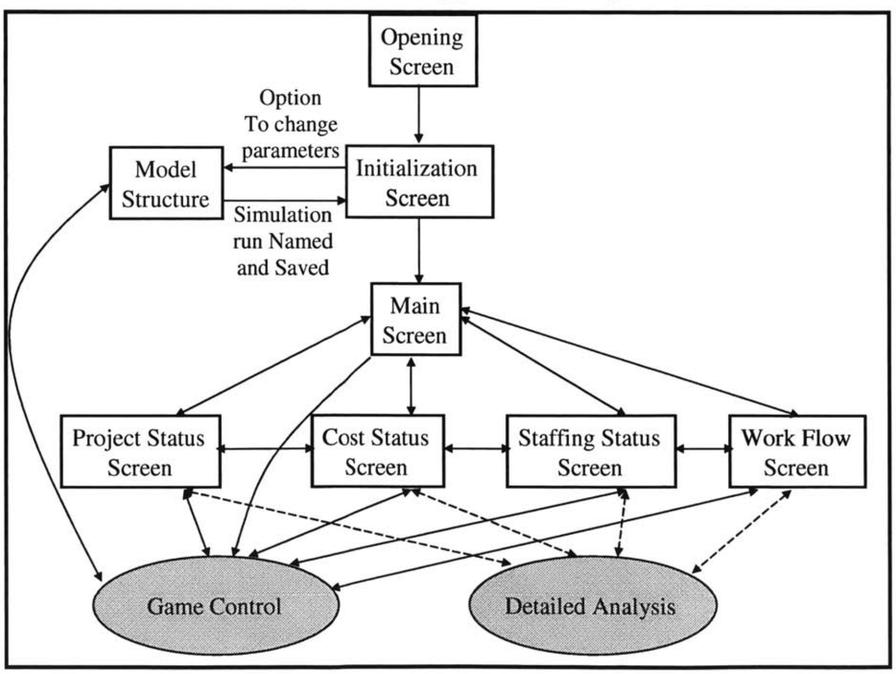

Figure 48. Architecture of BP NPD MFS

Need Category	Requirement	Functional Feature
Realism and Applicability to Petroleum Industry	1. Develop and validate a plausible story line with knowledgeable BP personnel	Design tasks and productivity profiles selected per BP design phase
	2. Use appropriate product development phase	Design Phase selected
	3. Use relevant information that a BP project manager reviews often	1. Schedule Completion: Feature % Project Complete, % Complete of Schedule Work 2. Budget: % Project $ Budget Expended, % of Scheduled $ Budget Expended 3. Staff: New Engineers, Experienced Engineers
Learning of System Dynamic Principles as applied to Program Management	1. Incorporate effects of schedule pressure, overtime, new hires and skill distribution, rework/testing resources, and scope creep	Incorporate System Dynamics Model based on Rework Cycle
	2. Incorporate effects of internal process concurrence	Model System Dynamics Model using Ford-Sterman Model
	3. Incorporate effects of external process concurrence	Simulate effect of upstream rework discovery in single-phase Ford-Sterman Model
User Interface and Playing	1. Users must be able to operate MFS without prior computer application knowledge (i.e., Vensim)	Stand-alone MFS with workings of model 'hidden' to user
	2. Incorporate "work-in-process" (WIP) graphs so players can see the effects of their decisions without looking for it	Incorporate WIP graphs for schedule, budget, staff, and EVMS
	2. Users must be able to access multiple feedback screens at one time	Use VENAPP MFS software that allows multiple screens
	3. Users must not be overwhelmed with detail in making decisions	1. Limit decisions to hiring, schedule, and overtime 2. Incorporate 'even numbers' in tasks, time, and personnel (10,000 tasks,100 weeks).
	4. Users must be able to advance at project with different time intervals	Incorporate 1 week, 2 week, 1 month, 1 quarter advance options
	5. Screen must have fonts big enough to be viewed on a projector for showing how to play it	Incorporate large fonts and 800 X 600 screen resolution.
Access to Model	1. Provide behavior modes for all model variables	Incorporate analysis capability to view behavior modes of all variables.
	2. Provide causal relationship information for all model variables	Incorporate feature to show causal relationships for all variables
	3. Provide 'where used' relationship for all model variables	Incorporate feature to show where all variables are used
Simulation	1. Initialize staffing levels in model based on tasks and completion date	Incorporate initialization screen that
	2. Provide way to change initial conditions to 'set-up' realistic but challenging project conditions	Incorporate initialization screen that allows input of intial staffing shortfalls
	3. Provide message when project is completed and stop simulation	Incorporate project finish logic to stop simulation and send message to player that simulation is over
	4. Have minimum required information for player to assess project performance on one screen but have additional screens for player to 'dig deeper' for more information if needed.	1. Incorporate main screen that gives visiblity to work accomplishment, anticipated finish date, and budget status. 2. Incorporate Project Status, Work Status, Staffing Status, and Cost Status screens by push-button if requested.
	5. Provide record of inputted decisions	Incorporate decision history graphs

Table 2. Requirements and Features of the BP NPD MFS

of the MFS. The MFS begins with an opening screen to introduce the player to the simulation. The initialization screen allows the player to define a name for the simulation run for comparison to later runs. The initialization screen also allows the option to change some of the parameters of the model, i.e., activate switches in the model for the feedback effects to be in effect, vary staffing shortfalls, set initial project scope and completion date, etc. The initialization screen provides the initial interaction with the model structure and sets the initial values for key stock variables in the model.

After initialization, the MFS goes to the Main Screen, which will be the primary screen used for the simulation. The user can then advance the simulation through Game Control commands on the game screen (input desired staff increase/decrease, overtime hours, or completion date and advance 1 week, 2 weeks, 4 weeks, or 1 quarter). The player can access the Human Resources (staffing status), Project Status, Cost Status, or Work Flow screens from the Main Screen, and from those screens advance the game through the same game control features on those screens. After the simulation advances through the model structure, all screens are updated (the same screen remains visible to the player after advancing), and all figures and WIP graphs are updated based on the model results. At any time, and from any screen, the player can access information on any variable from the Detailed Analysis screen. The player continues playing until the project reaches 98% completion[16] A screen showing a record of decisions for the simulation is also available at any time in the simulation and will be the screen returned by default when the simulation is finished.

[16] Because of the exponential nature of stock-flow equations where flows are dependent on the quantities of the stocks they are flowing from, a completion percentage of 98% must be used otherwise the project would drag on to unrealistic times. This is a fundamental limitation in system dynamics models and must be explained to players.

4.2.4. Description of the MFS Screens

A description of each screen of the MFS is provided in this section with illustrations of each screen. Appendix B contains the equation listing of the VENAPP for this MFS The text for the screens is sized for 800 X 600 resolutions for Windows based Personal Computers (PC). The Title Screen and Opening Screen are shown in Figures 49 and 50, respectively.

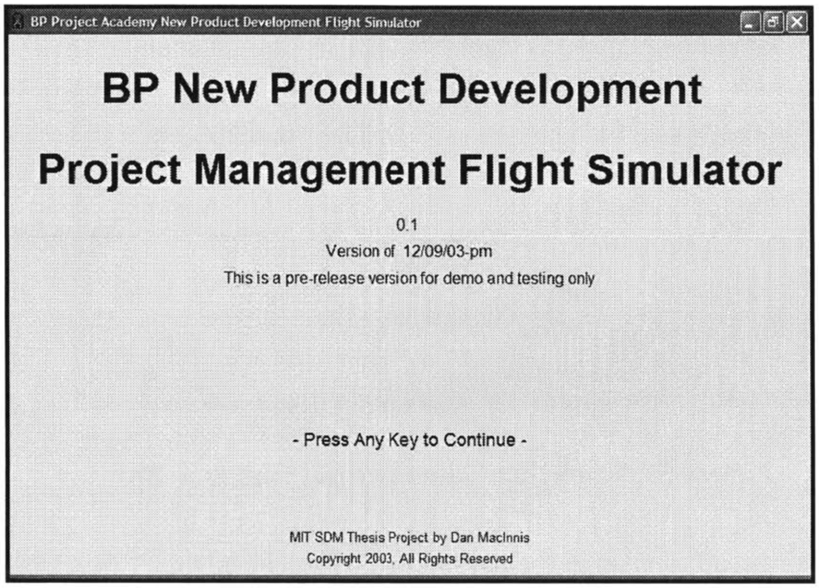

Figure 49. Title Screen for BP NPD MFS

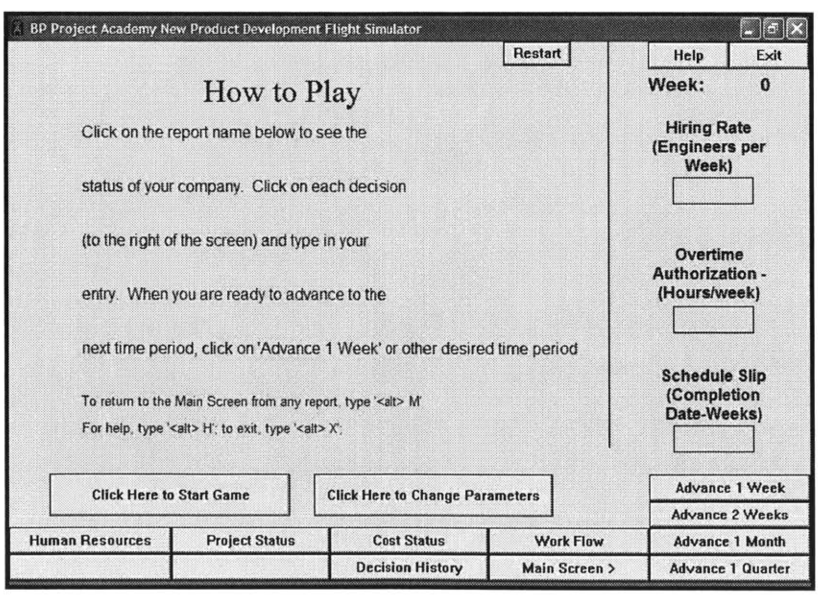

Figure 50. Opening Screen for BP NPD MFS

From the Opening Screen the player can chose to begin the simulation without changing parameters by clicking the "Click here to Start the Game" button, at which point the player will be presented with a "Name for new game output" output dialogue box (shown in Figure 51.)

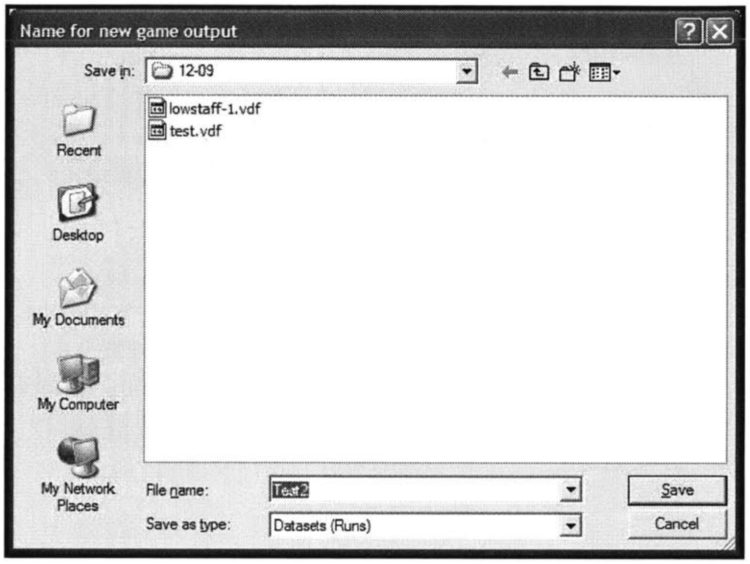

Figure 51. Name for Game Output Dialogue Box

The player can chose instead to change the parameters by selecting the "Click Here to Change Parameters" button which will bring up a screen to change model parameters (Figure 52.).

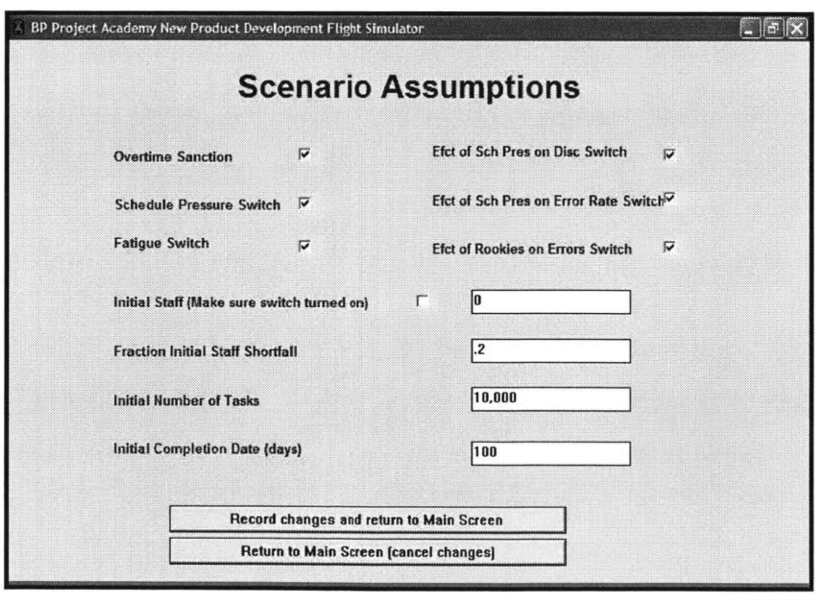

Figure 52. Scenario Assumptions (Change Parameters) Input Screen

Here the player can enable or disable the various feedback effects of the model (they are enabled by default), put in an initial staffing level (that supercedes the estimated staffing levels initialized in model), change the number of tasks or initial completion date (which effects the initial staffing estimates), or impose a percentage shortfall in staffing. The player can then either record these changes or return to the main screen without updating the parameters. The simulation name is named / saved with the same screen as shown in Figure 51 regardless of whether the parameters are changed.

The Main Screen is shown in Figure 53. The decision, game control, analysis, output, and status screen access are segregated for easy accessibility by the player as shown.

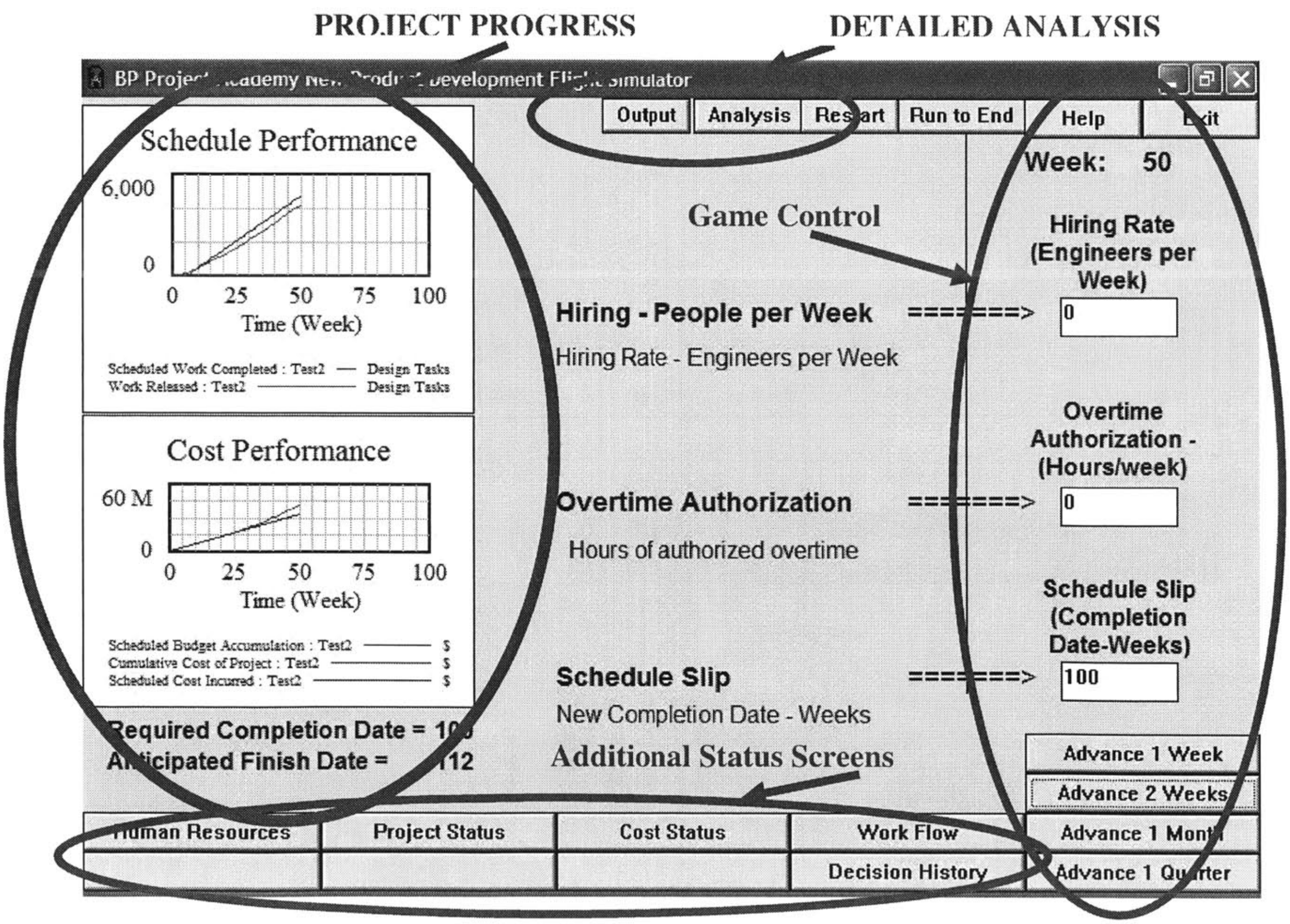

Figure 53. Main Screen of BP NPD MFS

Schedule Performance (tasks accomplished to date and scheduled tasks to date), Cost Performance (budget spent to date and scheduled budget expenditure plan) in the form of WIP graphs, and Anticipated Completion Date are shown in the project status portion of the Main Screen in the left hand side. Output and Analysis buttons are placed at the top of the screen (next to Reset and Run to End buttons) to access detailed analysis of the model variables. Finally, the Game Control section on the right hand side of the screen includes the current time, decision input values (Hiring Rate – engineers per week, Overtime – hours per week, and Schedule Slip – completion date), and game advance buttons.

Staffing Status is accessible in the Human Resources Screen shown in Figure 54.

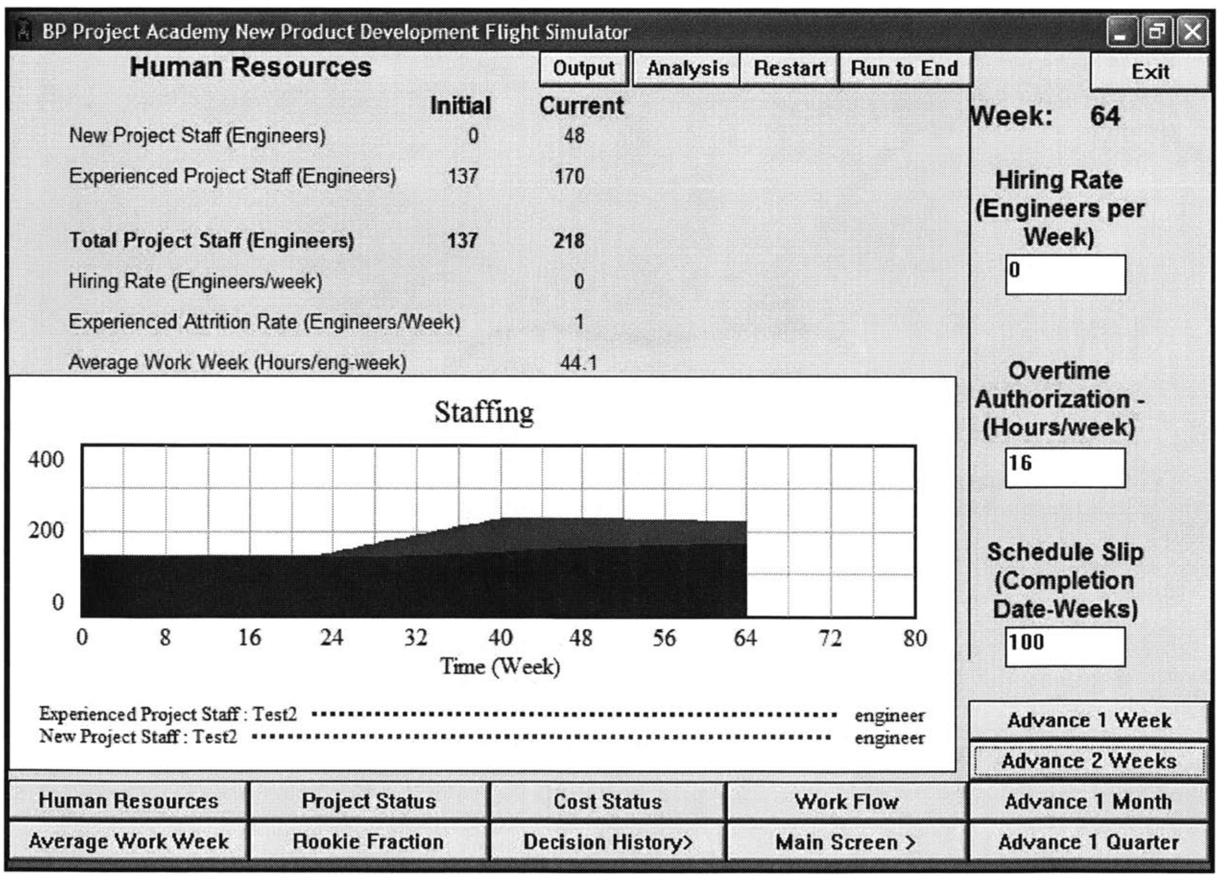

Figure 54. Human Resources Screen to Show Staffing Status

Staffing levels are shown on a WIP graph that shows Experienced Project Staff (in BLUE) and New Project Staff (in RED). Initial and current values are displayed in top left hand corner of the screen. Hiring Rate and Experienced Attrition Rate (engineers/week) are also shown. The value of Average Work Week (Hours/engineer-week) is shown to show parameter used to drive overtime effects. Buttons to transition to the other status screens, analysis screen, and game control displays/buttons are in the same location as the main screen. Two additional buttons (Average Work Week and Rookie Fraction) are shown at the bottom of the Human Resources screens to access additional human resource related information. By pushing these buttons, players can access pop-up windows of screens that can be sized to fit conveniently in the screen, as shown in Figure 55.

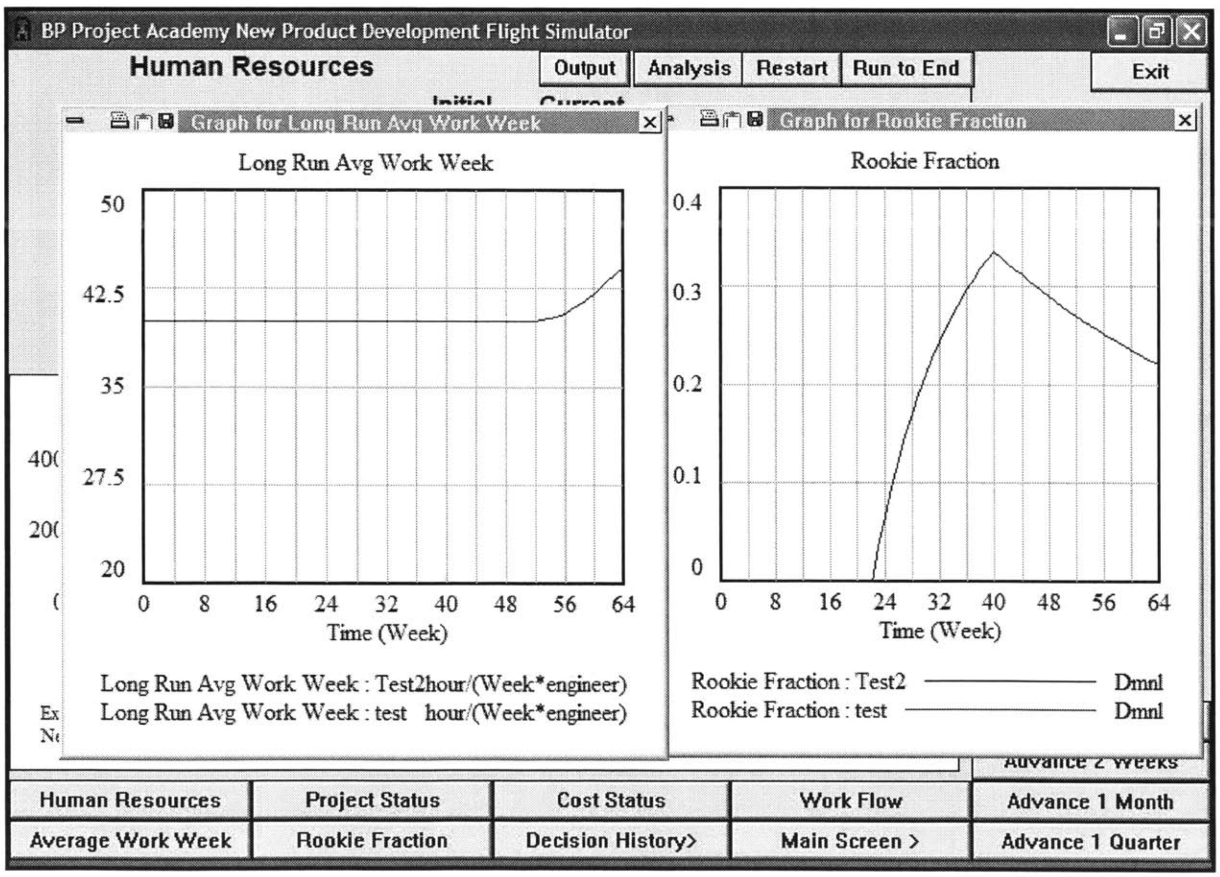

Figure 55. Long Run Avg Work Week and Rookie Fraction Displays

The Project Status screen, shown in Figure 56, provides an overview of project status including tasks scheduled, released, finish date, budget scheduled and expended, percent project scheduled and completed, and CPI and SPI EVMS parameters. WIP graphs of anticipated finish date and scheduled completion date, and scheduled and actual work release are also shown.

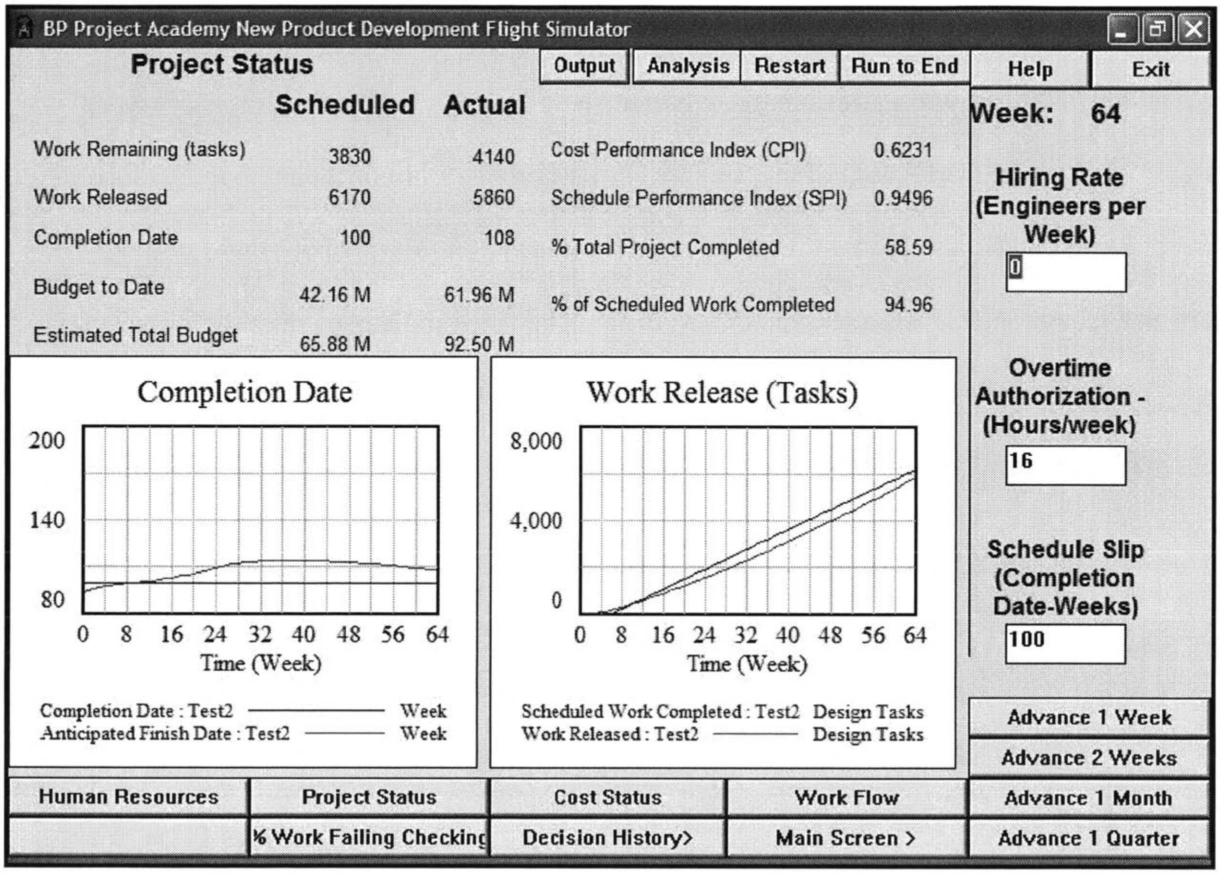

Figure 56. Project Status Screen

An additional button for *% Work Failing Checking* to access a graph of work defects is included at the bottom to open a pop-up window as shown in Figure 57.

The Cost Status screen is shown in Figure 58. Cumulative Cost, Engineering Cost (includes regular staff plus overtime costs), Regular Staff Costs, Overtime Staff Cost, and Overtime Hours is included in addition to Sanctioned Budget and Estimated Total Budget at completion. EVMS values of CPI and SPI are included as is the EVMS bulls-eye chart.

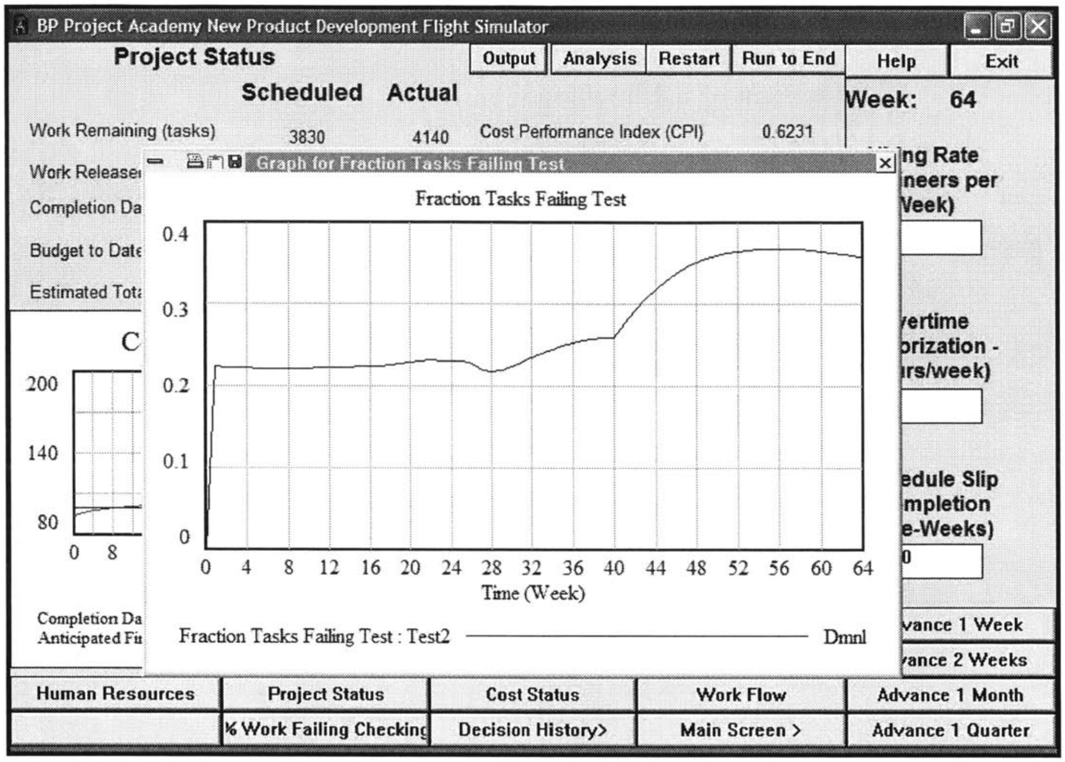

Figure 57. Fraction Tasks Failing Test (Rework) Display

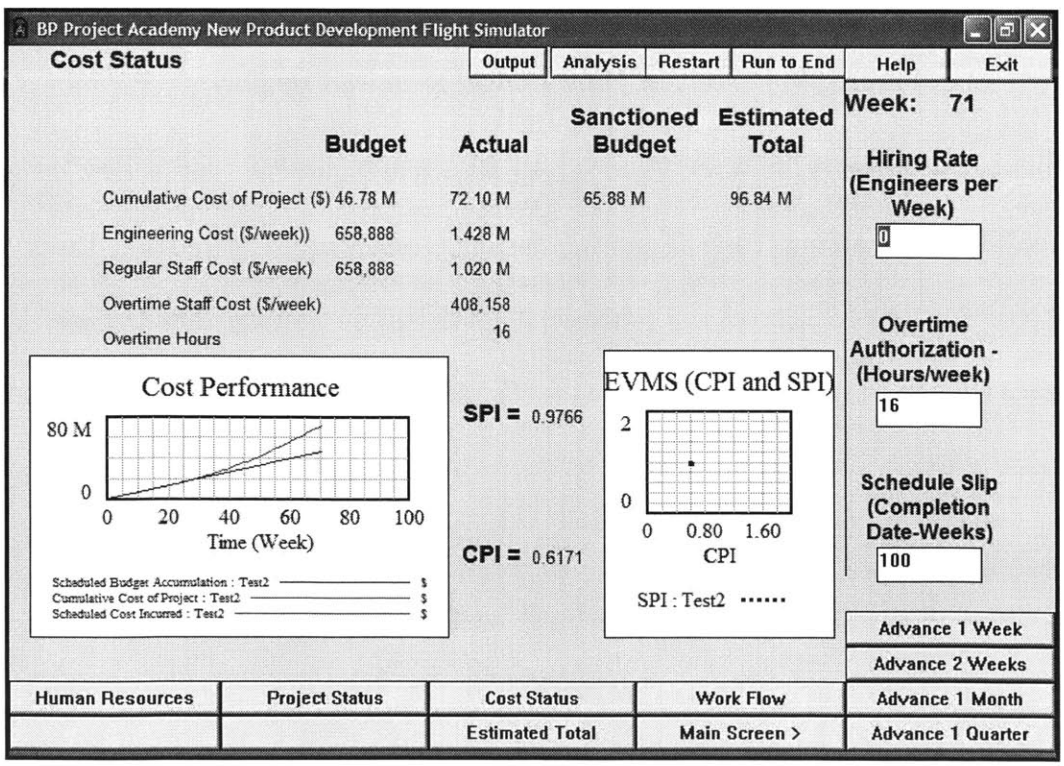

Figure 58. Cost Status Screen

A graph of Estimated Total button provides a time trace for Perceived Total Budget at Completion based on tasks remaining, as shown in Figure 59.

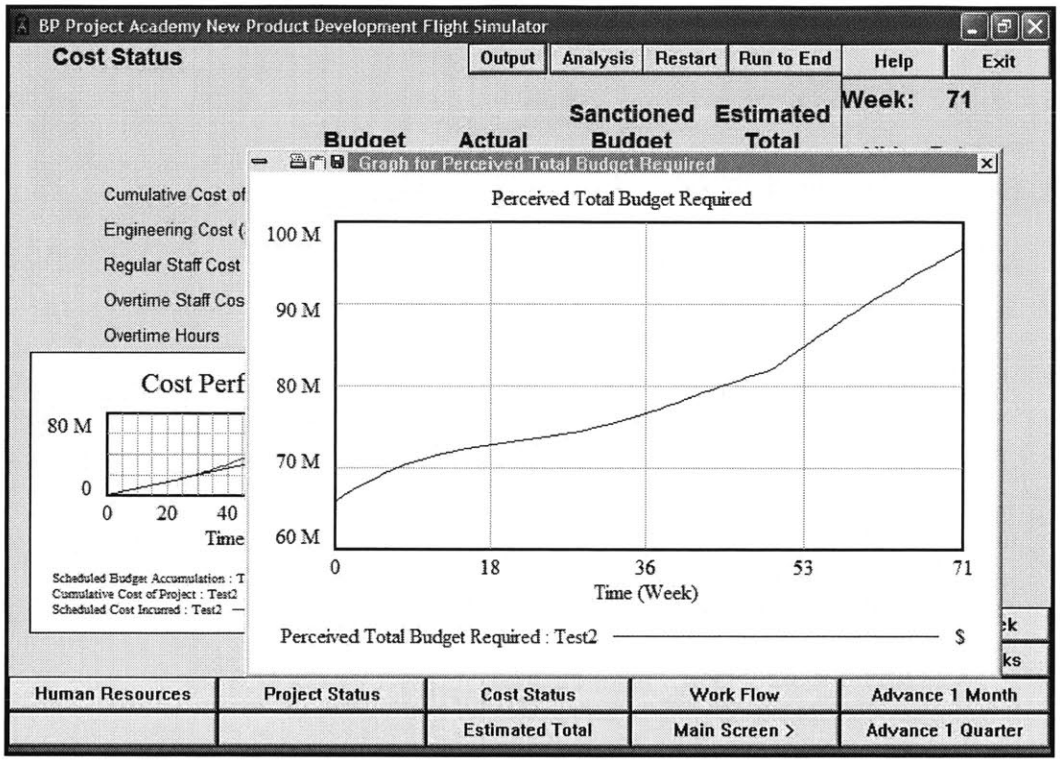

Figure 59. Perceived Total Budget Required Display

The final simulation screen is the Work Flow Screen, shown in Figure 60. This screen includes Uncompleted Tasks Remaining, Tasks Requiring Testing, Tasks Requiring Rework, Tasks Awaiting Approval, Tasks Released, and Fraction of Tasks Failing Testing. The Decision History Screen (Figure 61) provides buttons next to the decision input windows to access pop-up windows to see a history of decisions. Figure 62 shows this screen with all the pop-up windows activated. Finally, Figure 63 shows the Game Completion screen that pops-up when the *Project Finish* indicator in the model indicates that the model is at 98% completion.

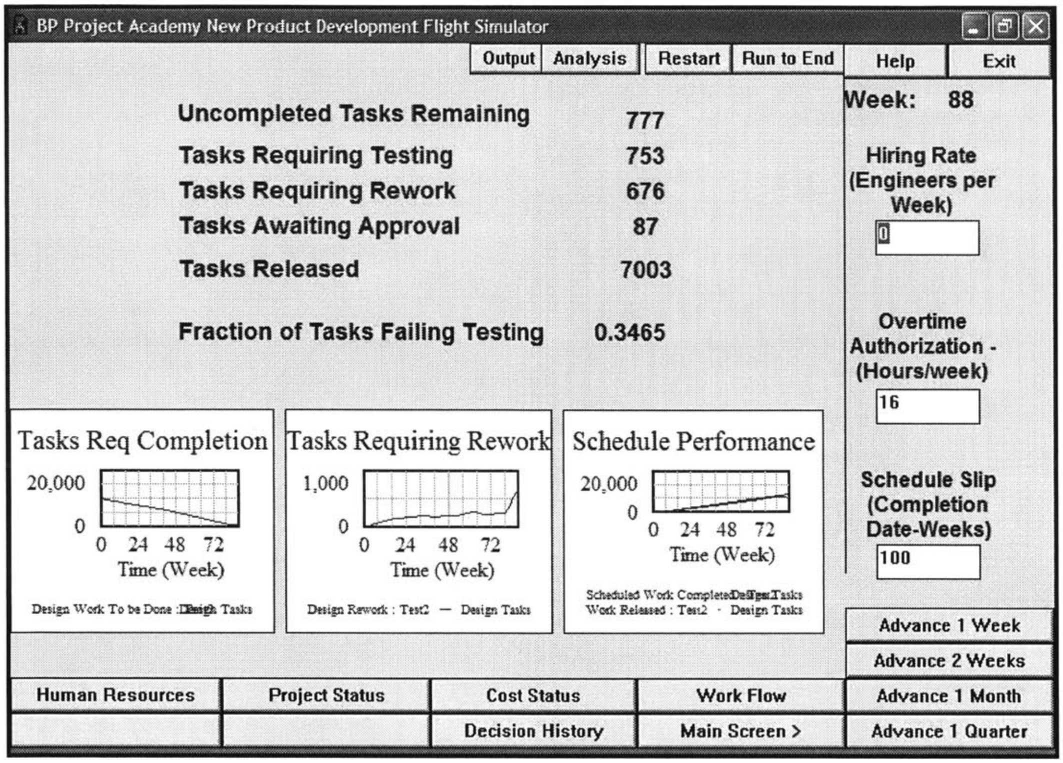

Figure 60. Work Flow Status Screen

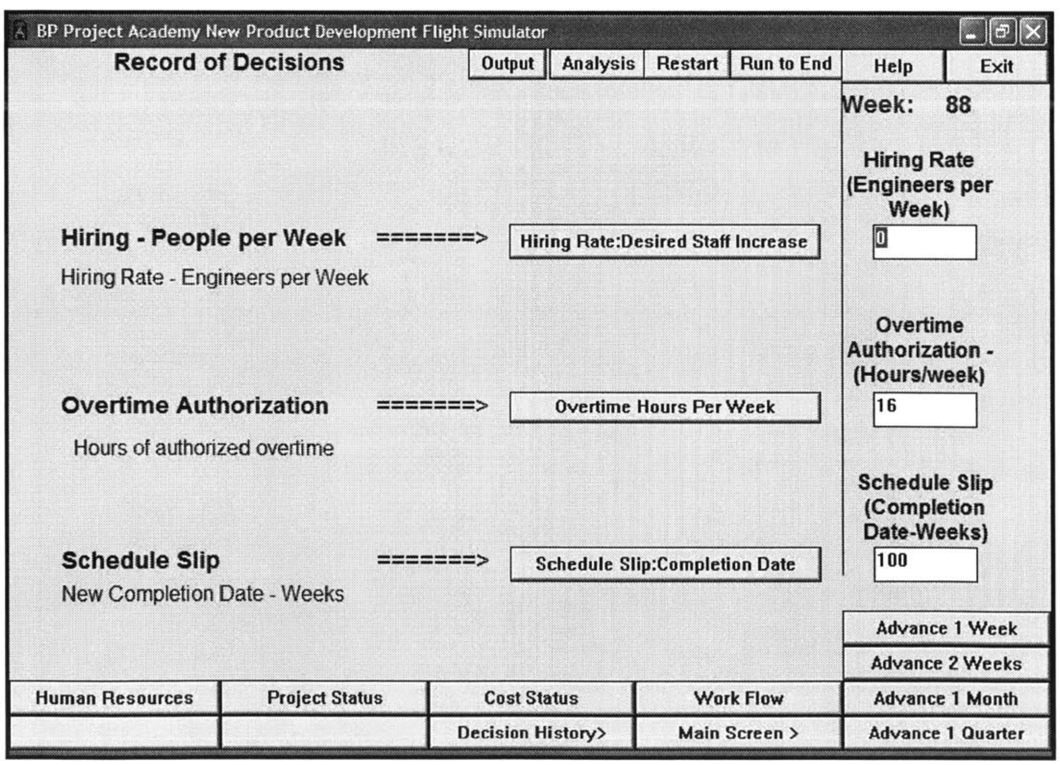

Figure 61. Decision History Screen

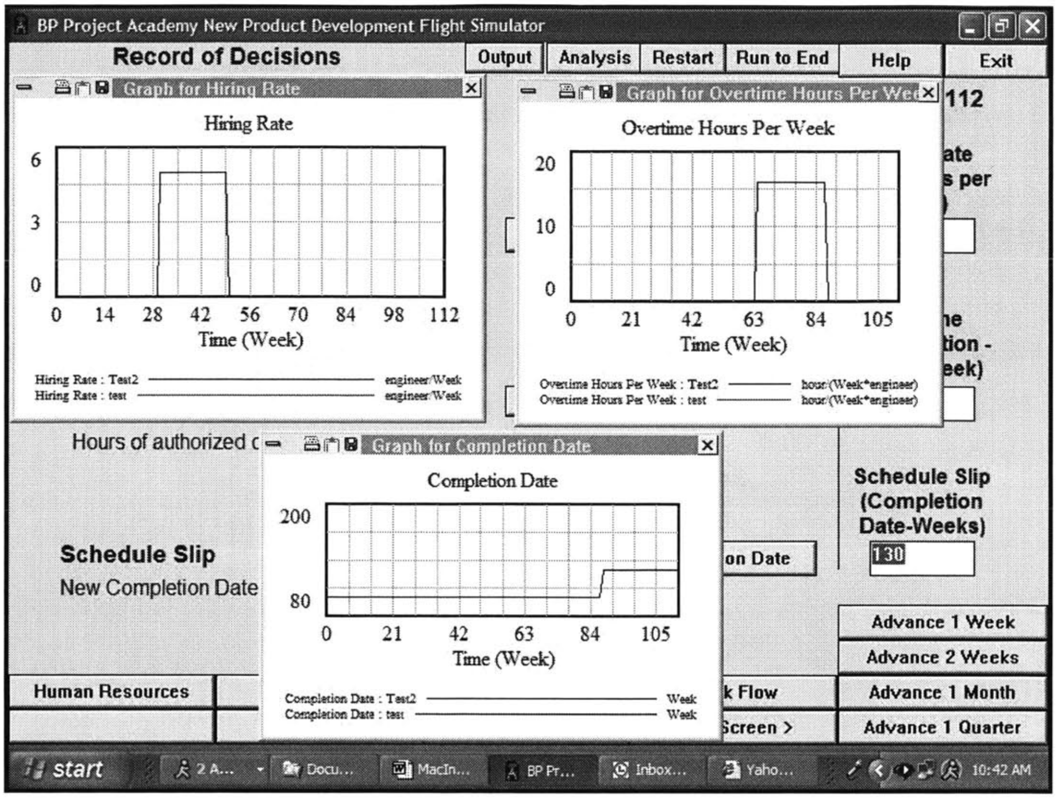

Figure 62. Display of Decisions Made Graphs

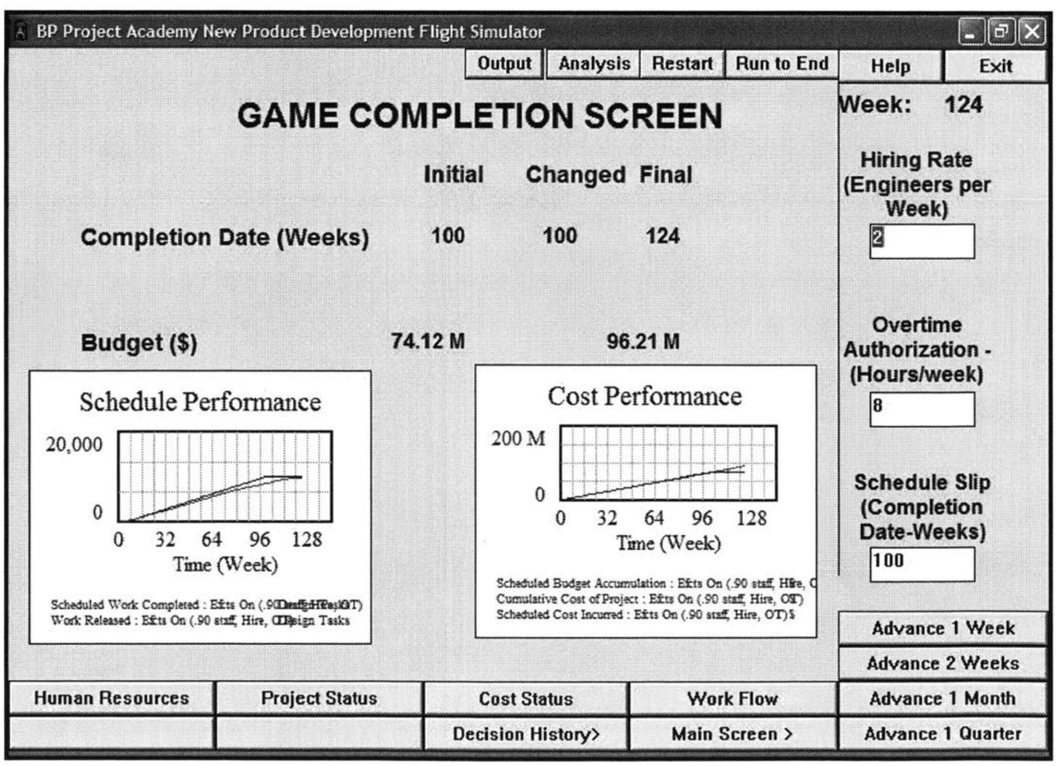

Figure 63. Completion Game Screen

The MFS provides easy access to the model parameters for understanding of the system dynamics model in the analysis module of the MFS. The structure for this portion of the MFS was adapted from a VENAPP template provided with VENSIM (Ventana Systems, Inc. 2003). The analysis module can be accessed through either the *Output* or *Analysis* buttons that are at the top of every simulation screen. By pushing the Output button, the output display screen, shown in Figure 64 is shown. This screen will show the graph of any variable in the BP NPD MFS system dynamics model. A button is included to toggle between graph and table format (*Show table*), and any variable can be selected for analysis through the *Select a new variable* button.

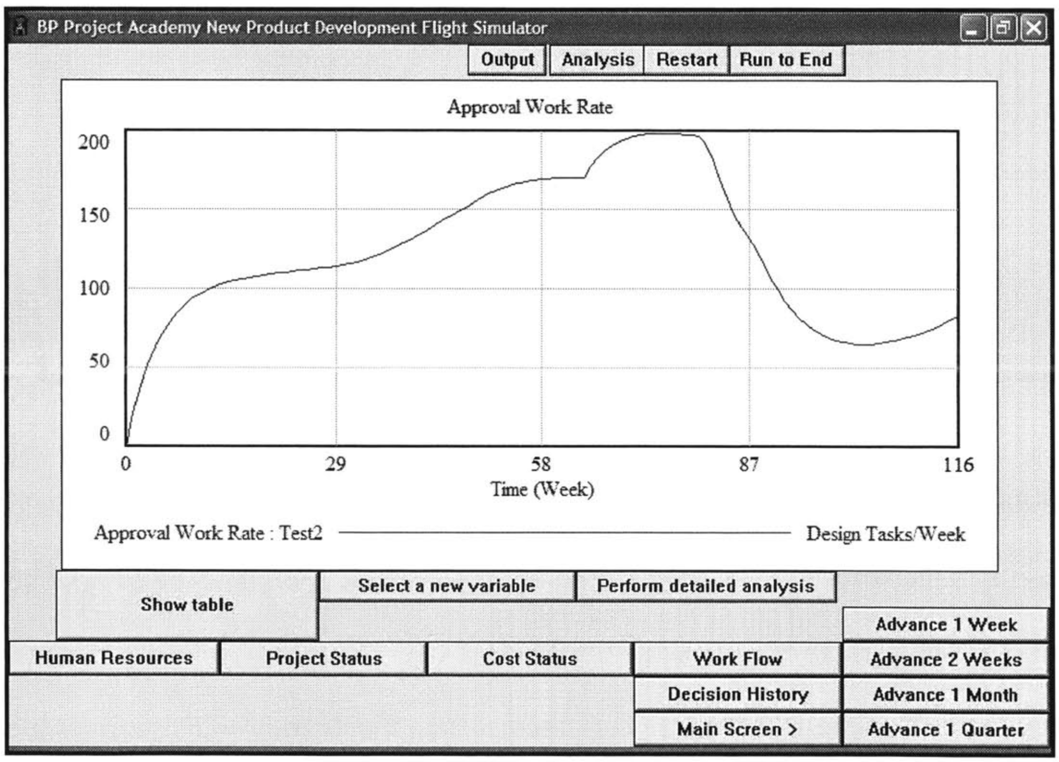

Figure 64. Output Screen for Analysis

Pushing either the *Perform detailed analysis* button in the middle of the screen, or the *Analysis* button at the top of the screen will bring the Detailed Analysis screen, shown in Figure 65. The *Analysis Control* section allows for the selection of past runs for comparison of data in the Output Screens on the other graphs from the other simulation screens. An option for selecting subscripts is also included to facilitate inclusion of multiple phases in later versions, but since subscripts (VENSIM model building technique) are not used in this MFS, this button is not used.

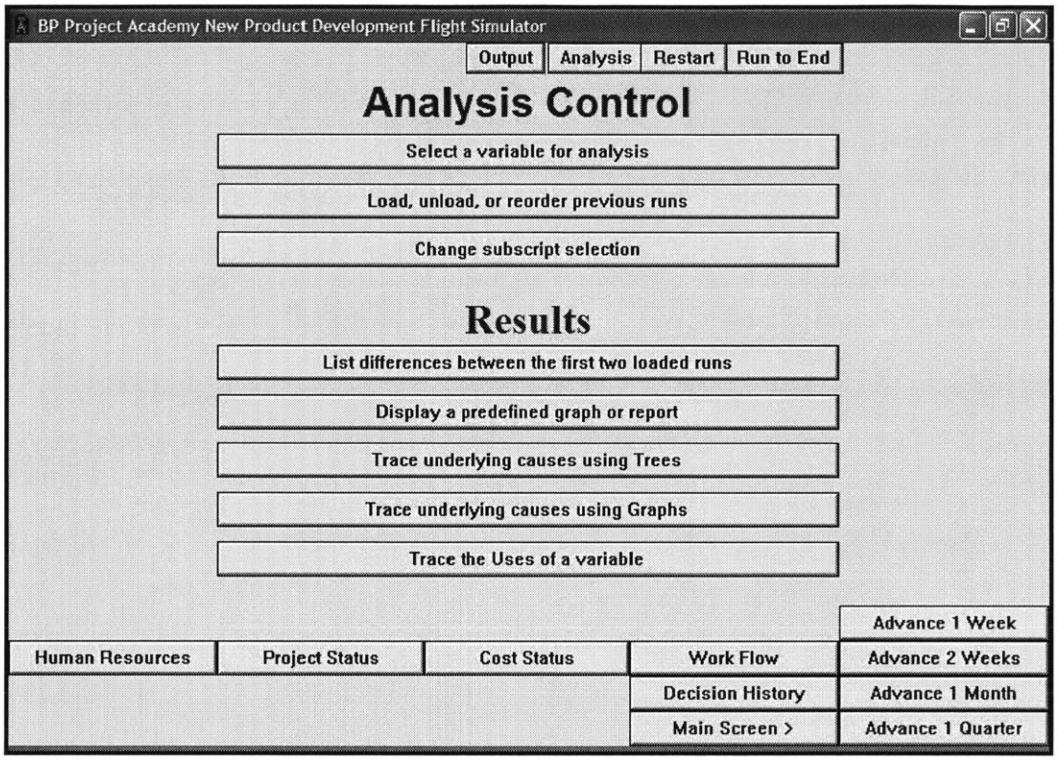

Figure 65. Detailed Analysis Screen

The *Results* section provides a button to list differences between first two runs loaded, a button to display a predefined graph or report (normally accessible from the VENSIM Control Panel), buttons to show traces of underlying causes using trees or graphs (including button in those resulting screens to provide equation definition of the variable in system dynamics model), and a button to show where used relationships. Examples of these very useful features are showed in Figures 66 through 70.

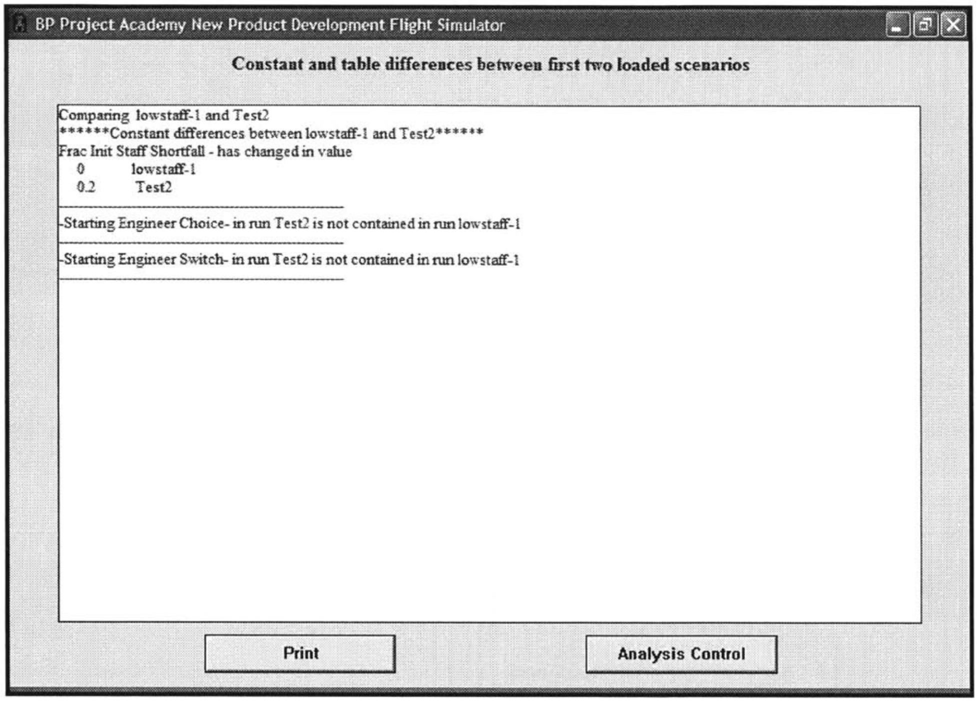

Figure 66. Differences Between Two Runs Display

The various analysis screens can be accessed and/or another variable selected from each other, or the player can revert back to the Detailed Analysis Screen by pushing the *Analysis Control* button. These output and analysis features are very helpful in understanding the model behavior while running the simulation. They are *not* intended to be used by first-time players for management training as they add too much complexity to the MFS for the players to learn the lessons playing the simulation. However, they are very useful in post-simulation learning and in development of the MFS. The next chapter will discuss the results found from multiple participants running the MFS.

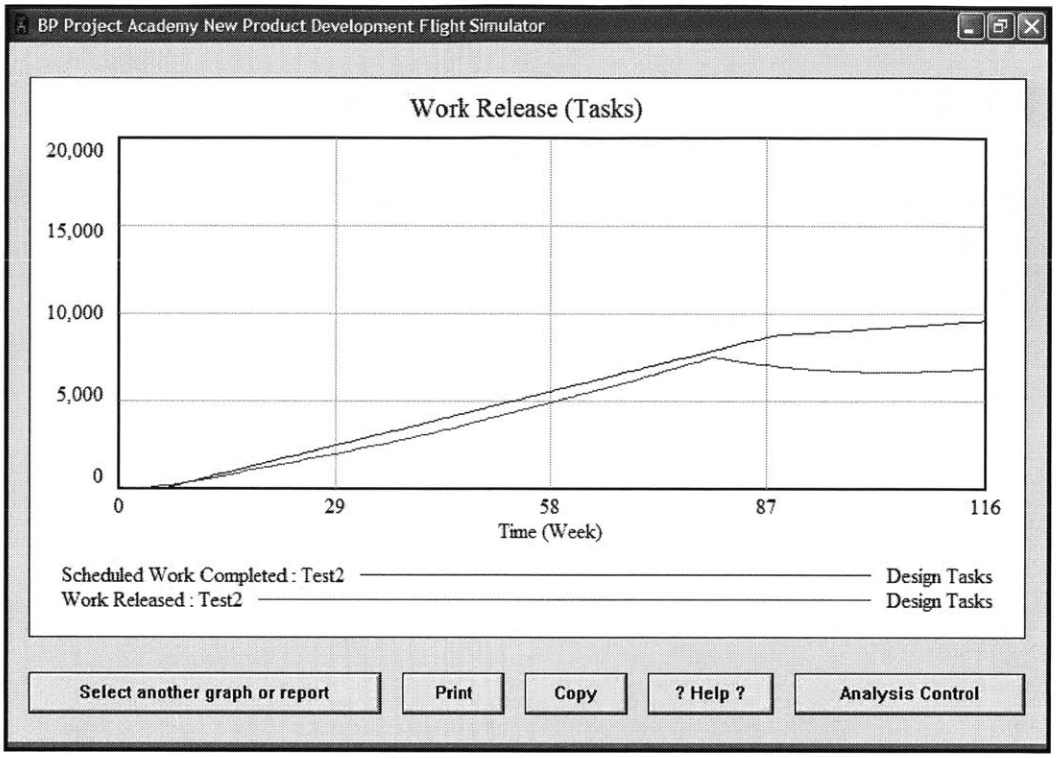

Figure 67. Example of Predefined Graph

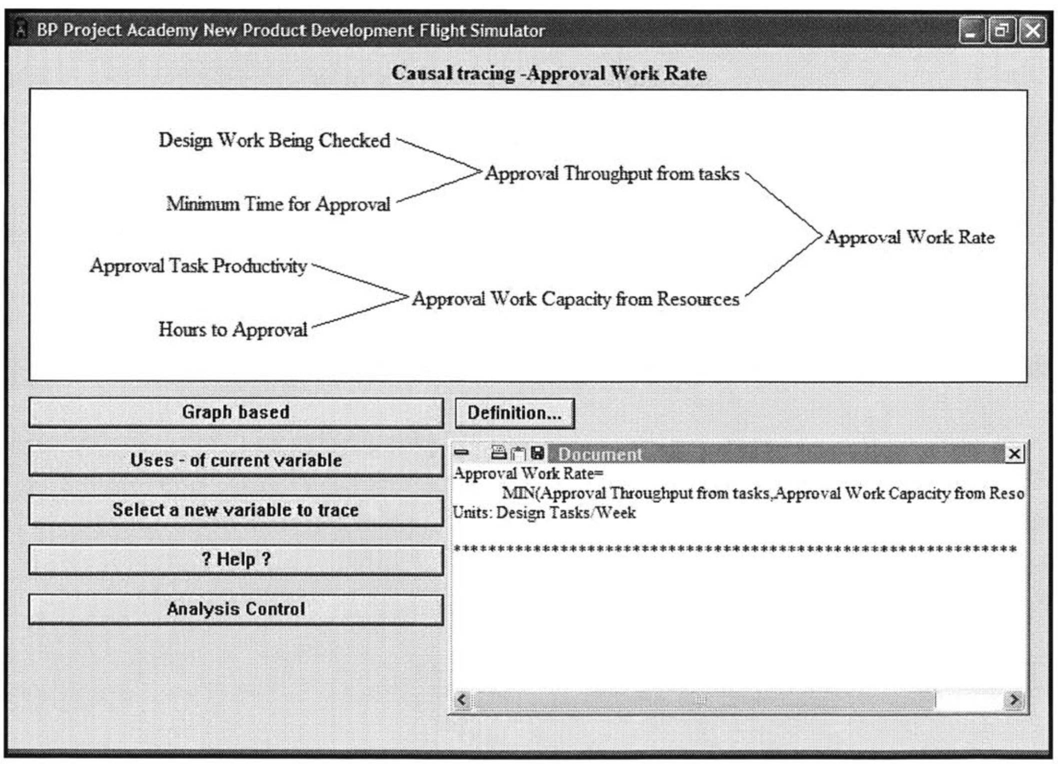

Figure 68. Causal Tracing Using Trees Display

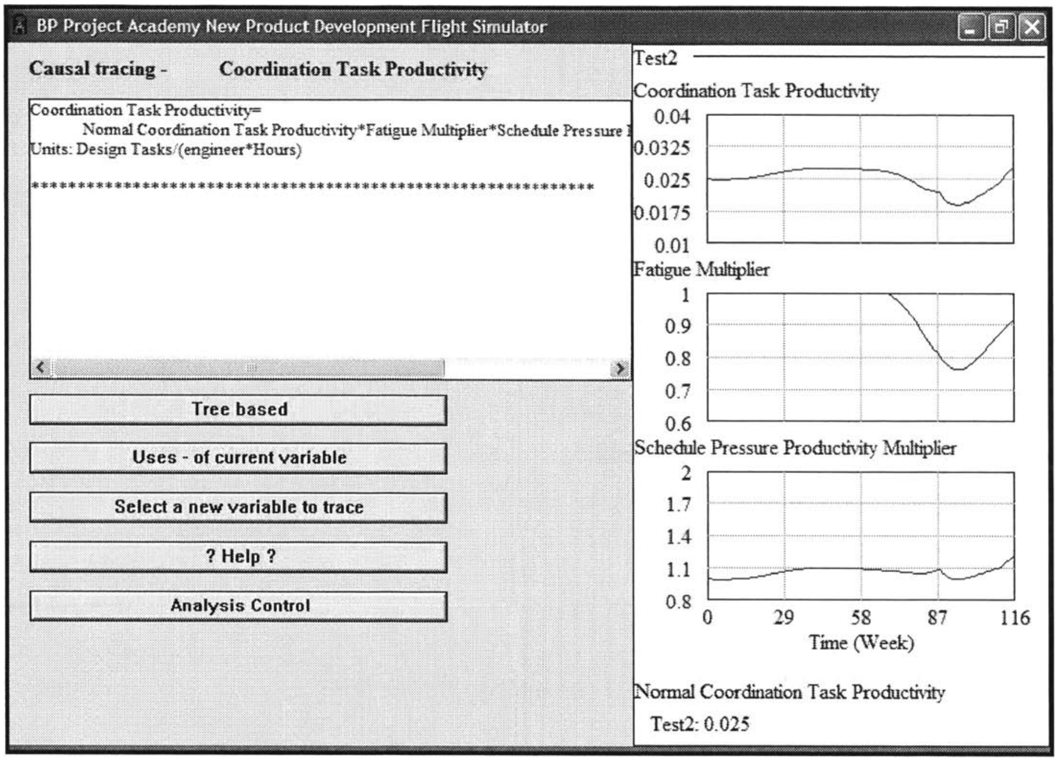

Figure 69. Causal Tracing Using Graphs Display

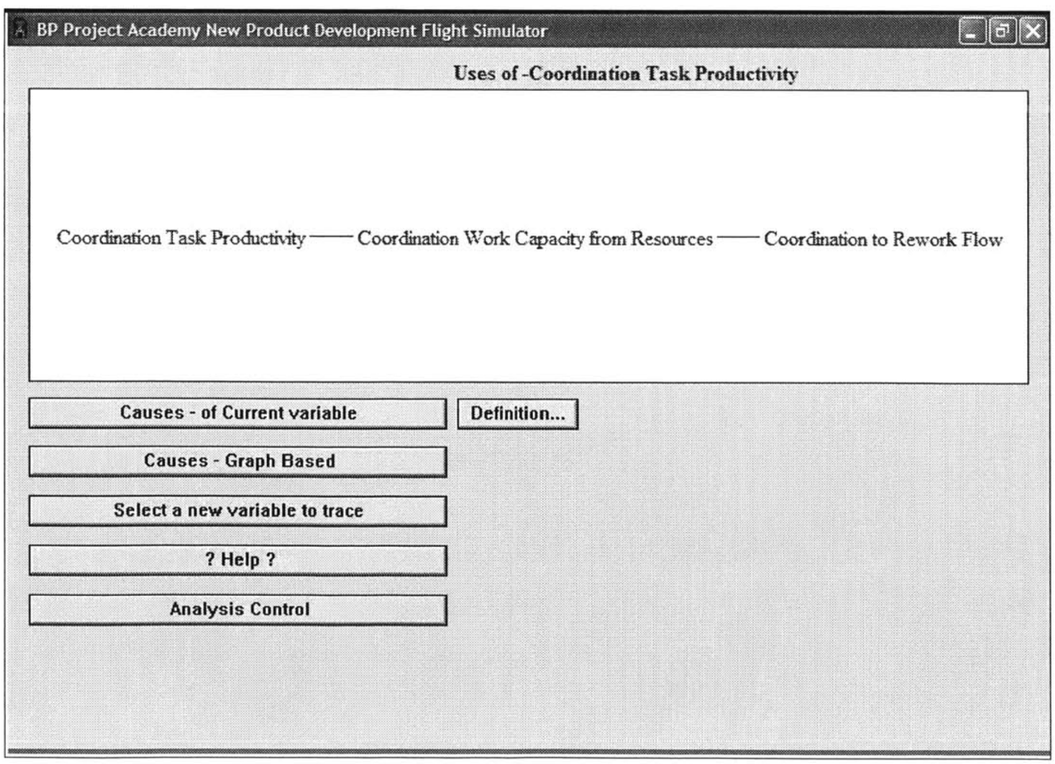

Figure 70. Where Used Relationships Display

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 5

Results

This chapter details the results from running the MFS including description of the project dynamics illustrated by the MFS and results from trial runs by faculty and PhD. candidates from the Management Sciences Department of the MIT Sloan School of Management.

5.1 NPD Project Dynamics

The dynamics of NPD projects can be simulated using the MFS. As was seen in Chapters 3 and 4, the stock-flow structure of the rework cycle and feedback effects from overtime, schedule pressure, and hiring of inexperienced personnel result in the adverse project dynamics that lead to problems of lateness and over-budget. An illustration of these effects can be seen in the traces of project performance with the MFS run with and without these negative effects active in the model.

Figure 71 shows the simulation run using the MFS with no effects turned on but with an imposed 15% staffing shortfall (because model sets initial staffing estimates, it is necessary to impose a 'pressure' into the simulation to compare the effects). No other decision variables are changed while running the simulation. The project ends in 110 weeks (10 weeks late) with a budget over-run of about $7M. Figure 72 shows the same scenario run with all the feedback effects turned on (effects of schedule pressure on productivity, quality, and attrition). The same simulation ends in 144 weeks with a budget overrun of about $15M. A similar experiment is conducted with and without feedback effects for a 15% staffing shortfall. Figure 73 shows the hiring and overtime decisions for runs without feedback effects (Figure 74) and with (Figure 75).

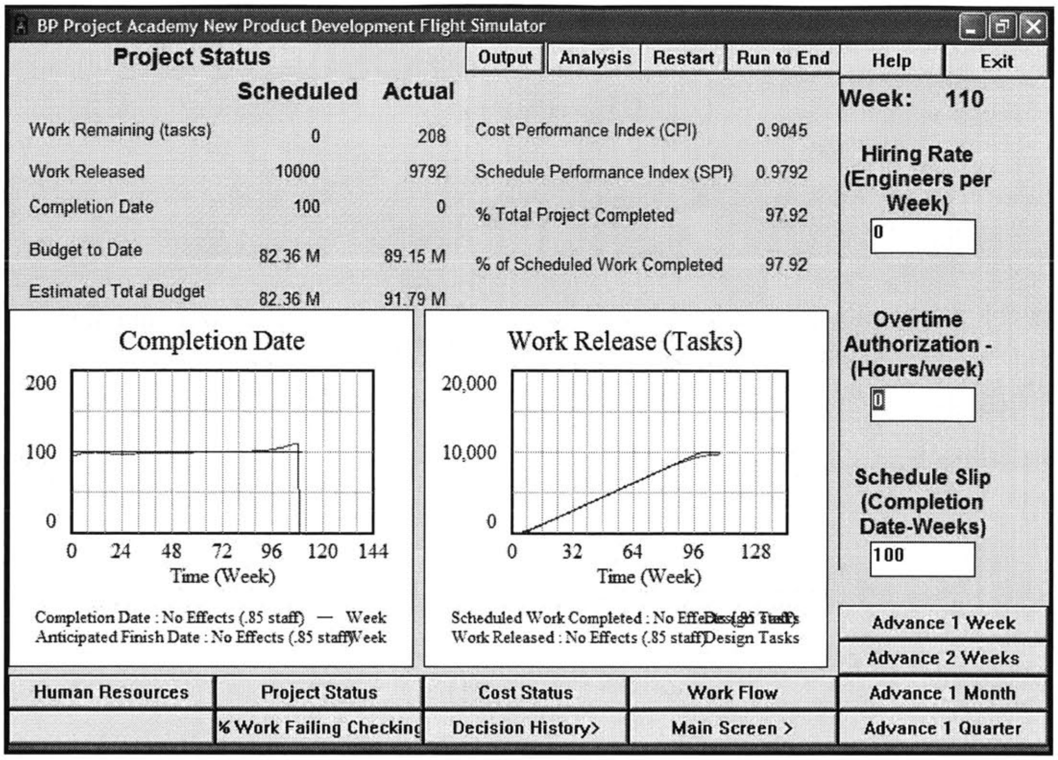

Figure 71. Simulation with Feedback Effects Deactivated, (15% Staffing Shortfall)

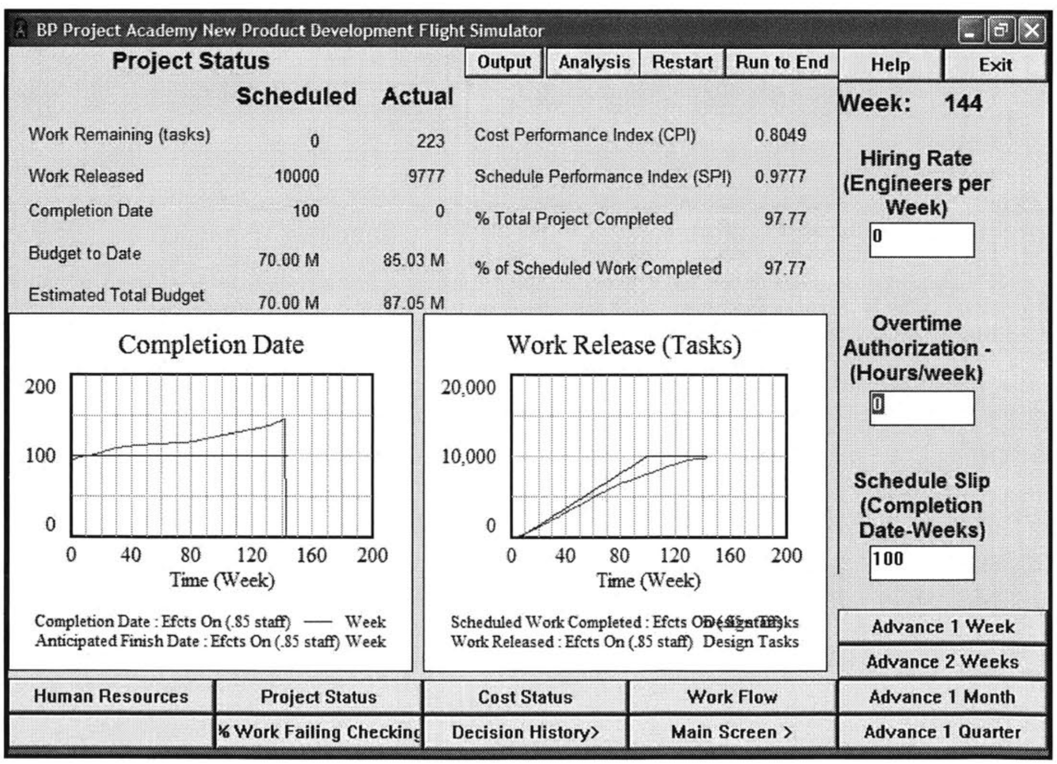

Figure 72. Simulation with Feedback Effects Turned On (15% Staffing Shortfall)

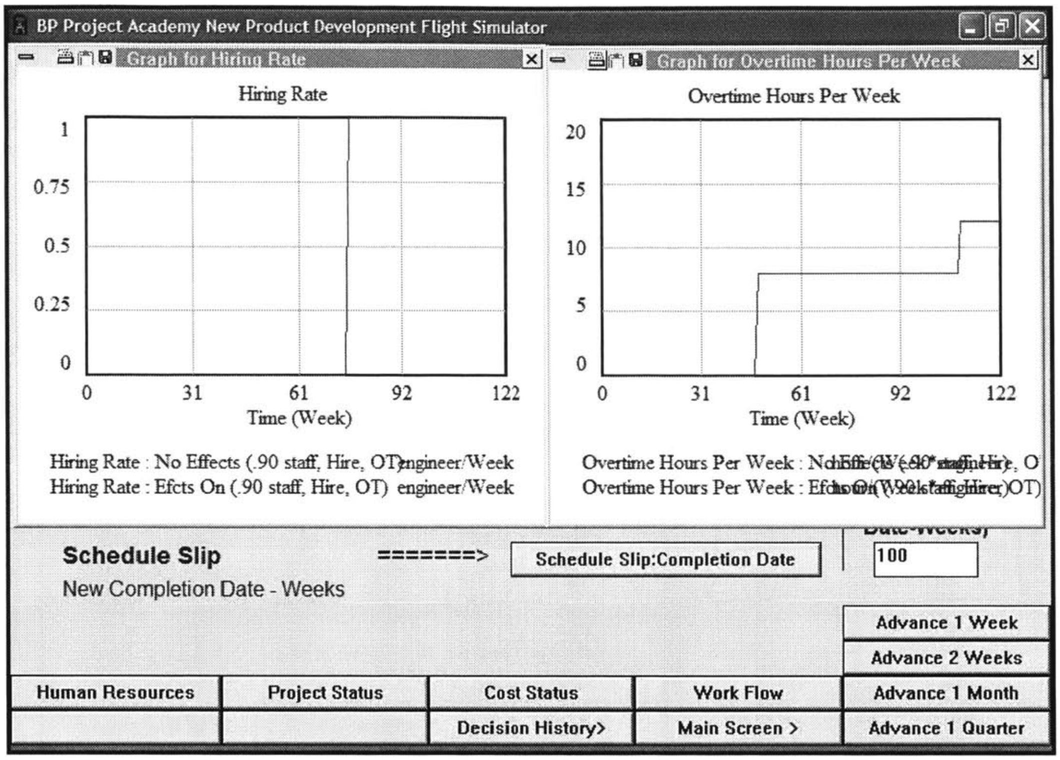

Figure 73. Simulation Hiring and Overtime Decisions for 15% Staffing Shortfall

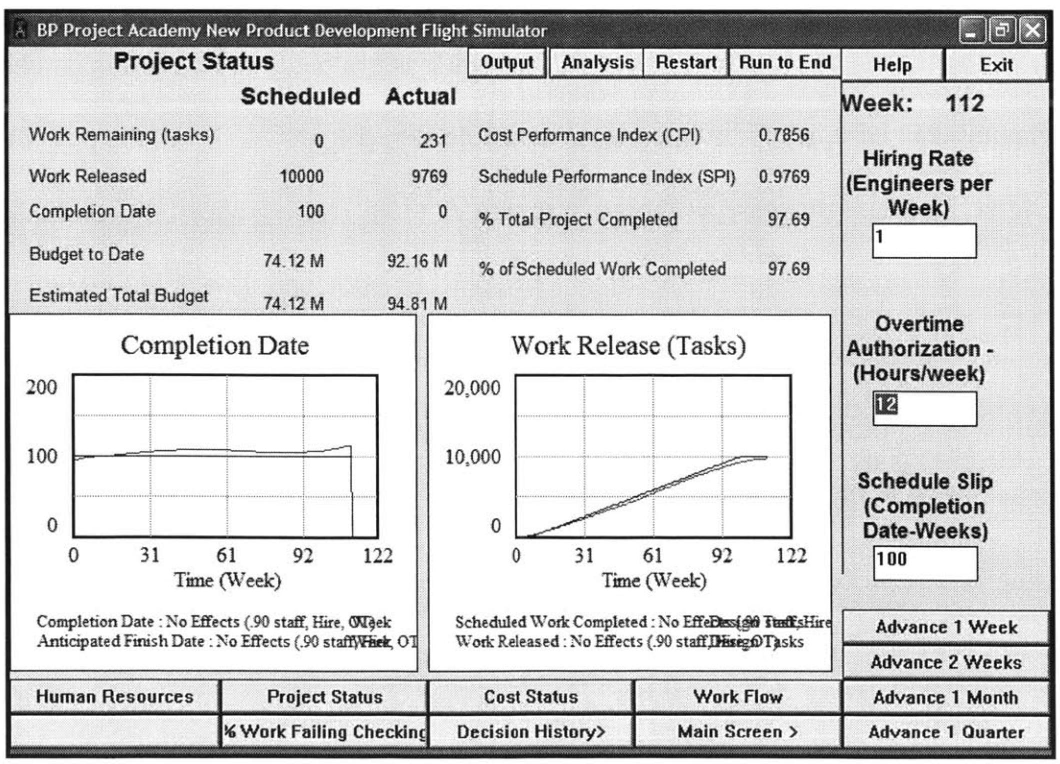

Figure 74. No Feedback Effects for 15% Staffing Shortfall with Hiring and Overtime

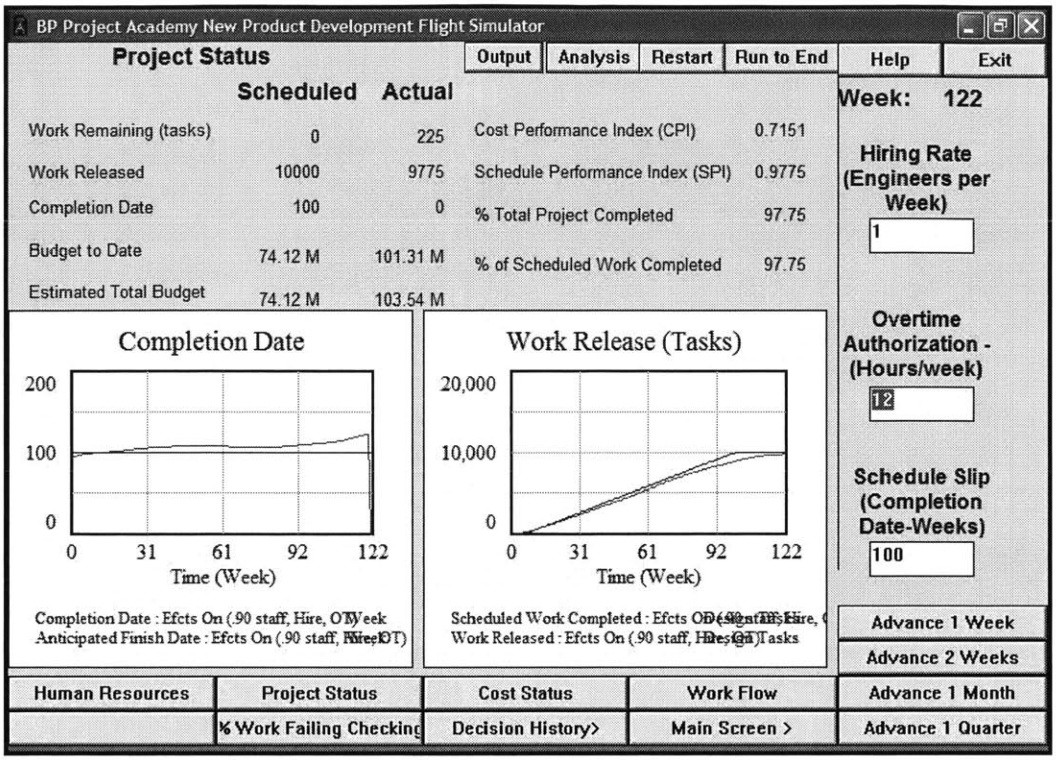

Figure 75. Feedback Effects for 15% Staffing Shortfall with Hiring and Overtime

Finally, the management action of slipping schedule is evaluated to determine if schedule pressure dynamics can be relieved by intentional schedule slip. Figure 76 shows the project completion screen for the same case as shown in Figure 72 (15% staffing shortfall, feedback effects turned on, intentional schedule slip to 130 weeks when anticipated completion date is 108 weeks). As can be seen, the project completes *sooner* (126 weeks vs. 144 weeks) than if the same schedule was kept in place. The MFS accurately portrays the project dynamics that drive undesirable project performance as discussed in Chapter 3.

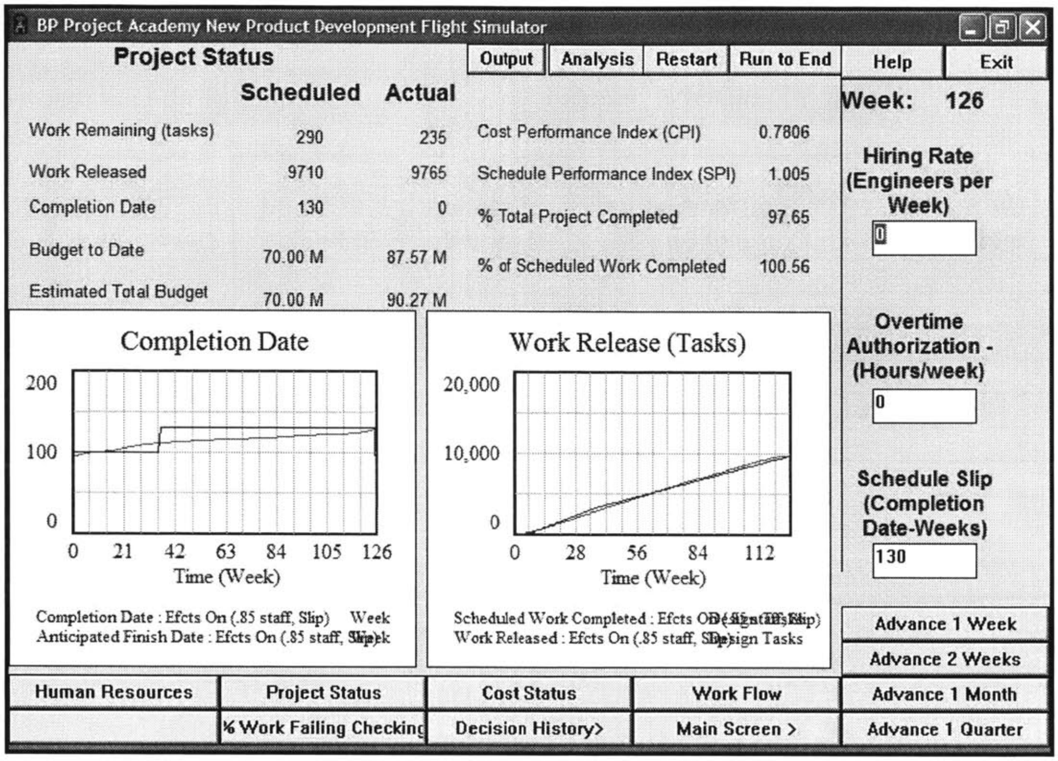

Figure 76. Feedback Effects for 15% Staffing Shortfall, Schedule Slip

5.2 Results from "Flight Testing" the MFS

After the first prototype of the MFS was completed, it was made available to Faculty and PhD candidates from the Management Science Department of the MIT Sloan School of Management. Players were instructed how to run the simulation and access the various simulation screens and game controls. Initial 'games' were played without any staffing shortfalls, and all feedback effects were activated. Final completion dates and final budget completion dates were recorded as shown in Table 3.[17]

[17] The results for this 'test flight' session were for a simulation that incorporated 5,000 tasks in 120 weeks. The final version was modified to incorporate 10,000 tasks in 100 weeks, due to feedback from this flight test session.

A graphical representation of these results with the "on schedule, on budget" value indicated is shown in Figure 77. What is interesting to note is the number of players that were

Final Completion Date (weeks)	Total Budget at Completion ($M)	Total Weeks Late to Schedule	% Late to Schedule	% Overbudget
120	49.7	0	0.0%	22.1%
146	72.4	26	21.7%	77.9%
64	74	-56	-46.7%	81.8%
127	59	7	5.8%	45.0%
130	49.5	10	8.3%	21.6%
96	57	-24	-20.0%	40.0%
108	55	-12	-10.0%	35.1%
145	58	25	20.8%	42.5%
116	53	-4	-3.3%	30.2%
130	48.9	10	8.3%	20.1%
113	51.9	-7	-5.8%	27.5%
71	66	-49	-40.8%	62.2%

Table 3. Results from Initial "Test Flights" of BP NPD MFS

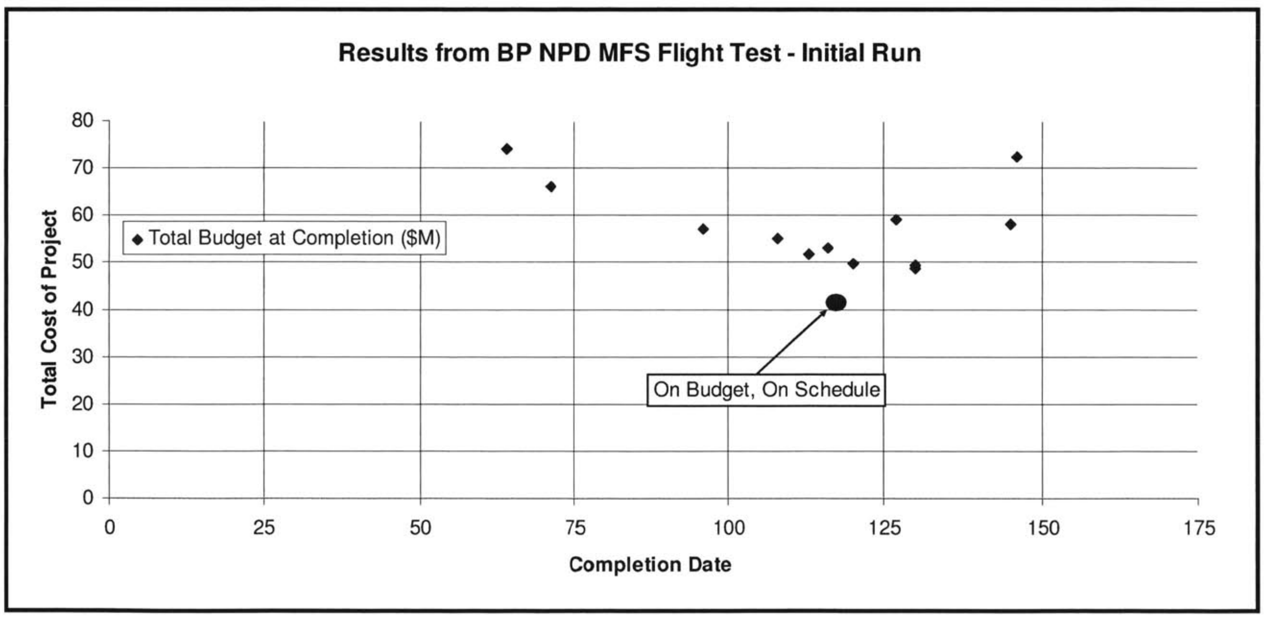

Figure 77. Graphical Results from Initial MFS "Flight Test"

able to achieve project completion ahead of schedule. The author and other non Management Science players who played the game averaged around 140 weeks. The Management Science faculty and students, however, were aware of the project dynamics and opted for "hire early" strategies that although resulted in higher total project costs, were able to get the project completed well before the 'norm'. In fact, although the same total cost was spent, (e.g., around $60M), those that played the simulation with the "right strategy" were able to complete the project about 40 weeks sooner. These results will be discussed in more detail in Chapter 6.

The intent of the flight test was not only to assess the correct strategy to optimize project performance results, but to identify areas for improvement. The flight test was helpful in identifying improvements to the MFS. Table 4 contains the list of improvement ideas not only from the flight test but throughout the development process. Some particularly useful suggestions were to make even numbered task and completion dates to ease the mental calculation required of the players to maximize the learning impact of playing the MFS (Sterman, 2003). Additionally, internal process concurrence relationships were found not to be working as expected and led to refinement of the model.

In Conclusion, the MFS, by incorporating a system dynamics model using the rework cycle, internal process concurrence, and quasi-external process concurrence relationships of the Ford-Sterman model, reflects the undesirable project dynamics that lead to new product development project failures, namely late to schedule and over-budget. The next Chapter will discuss these results in more detail to show how the MFS can be effective in teaching why project dynamics are so crucial to understanding why projects fail and how to change management actions to ensure project success.

Venap Version	Model Version	Features Added	Modeler	Date	Comments
BP-MFS-11-19-03	BP-MFS-11-19-03-am.vmf	1. The opening input screen now works 2. Fixed some errors in the model 3. i added a new way to calcuate cost and schedule--the old version did not change with deadline slips--see what you think. 4. i added schedule pressure as a variable that can be manipulated--its pretty simple write now, we probably can think of something better.	Repenning	11-19-03 AM	
BP-MFS-11-19-03-pm	BP-MFS-11-19-03-pm.vmf	1. Ensure Run to End button on every status screen 2. Cost Screen Changes: a. Added Estimated Total to Cost (Perceived Total Budget Required in Model) b. Added Estimated Total button for graph 3. Project Screen Changes: a. Took out Weeks Remaining b. Added Scheduled and Actual columns for work remaining and work released c. Made completion date WIP graph smaller, added work release WIP graph to compare sched vs. actual d. Added Fractional Defects in Test plot button, added fractional defects in test to status numbers 4. HS Screen changes: a. Added Long Run Average Work Week b. Take out FTE Project Staff, Experience gain rate, rookie discount c. Fixed Frac Init Staff Shortfall MODVAR button - changed it back to sched pressure 5. Worked on ending program after project finish = 1, preliminary IFTHENELSE, BRANCH code added but game bombed 6. Fixed various screen issues (made sure had 1 wk, 2wk, 1mo, 1QTR on each input screen, etc.)	MacInnis	11-19-03 PM	
BP-MFS-11-20-03-am	BP-MFS-11-20-03-am	1. Fixed avg work week in HR screen--just had wrong showvar	Repenning	11-20-03 AM	
BP-MFS-11-20-03-am	BP-MFS-11-20-03-am	2. Added rework fraction to workflow screen	Repenning	11-20-03 AM	
BP-MFS-11-20-03-am	BP-MFS-11-20-03-am	3. Updated formulation for error discovery--now it goes down and fatigue, rookies and pressure go up	Repenning	11-20-03 AM	
BP-MFS-11-20-03-am	BP-MFS-11-20-03-am	4. Also change error formulation slightly, now it's the presure ratio--how much schedule pressure you feel-- that influences the error rate, otherwise as fatigue goes up, error rates fall, which doesn't make sense.	Repenning	11-20-03 AM	
BP-MFS-11-20-03-am	BP-MFS-11-20-03-am	5. Add simple formulation whereby errors making it to release are eventually returned and require coordination, its simple but it works.	Repenning	11-20-03 AM	
BP-MFS-11-20-03-pm	BP-MFS-11-20-03-pm	1. Removed Back up button from control screens - wasn't working	MacInnis	11-20-03 PM	
BP-MFS-11-20-03-pm	BP-MFS-11-20-03-pm	2. Added consistent go to buttons to get to other screens on every control screen	MacInnis	11-20-03 PM	
BP-MFS-11-20-03-pm	BP-MFS-11-20-03-pm	3. Changed Main screen to have bigger WIP graphs, removed DOTS	MacInnis	11-20-03 PM	
BP-MFS-11-20-03-pm	BP-MFS-11-20-03-pm	4. Added Analysis Button and Analysis functions from Template	MacInnis	11-20-03 PM	May be too much for BP, but will help us
BP-MFS-11-20-03-pm	BP-MFS-11-20-03-pm	5. Added Output Button and Output functions from Template	MacInnis	11-20-03 PM	May be too much for BP, but will help us
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	1. Added Project Finish Level stock and incorporated Bob Eberlein's logic for project finish status screen	MacInnis	11-28-03 PM	
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	2. Changed Project Finish Level to 99.9% to get a better finish date	MacInnis	11-28-03 PM	
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	3. Added Anticipated Finish Date to Main and Project Screens	MacInnis	11-28-03 PM	
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	4. Added Sanctioned Budget to Cost Status	MacInnis	11-28-03 PM	
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	5. Added Earned Value Metrics (CPI and SPI added to Project and Cost Screens). Bullseye chart added to Cost Screen	MacInnis	11-28-03 PM	
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	6. Corrected number formats for integer values (tasks, employees, etc.)	MacInnis	11-29-03 PM	
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	7. Added Initial staff numbers to HS screen per Nick suggestion	MacInnis	12-1-03 AM	
BP-MFS-11-26-03-pm	BP-MFS-11-26-03-pm	1. Changed project finish level back to .98	MacInnis	12-2-03-PM	
bp-mfs-12-02-03-pm-venap	bp-mfs-12-02-03-pm-model	2. Added Project Finish completion screen (Note - have to use time for final time since anticipated completion date goes to zero after project finished - need to look into that	MacInnis	12-2-03-PM	
bp-mfs-12-02-03-pm-venap	bp-mfs-12-02-03-pm-model	3. Only two graphs on main screen - counters for completion date	MacInnis	12-2-03-PM	
bp-mfs-12-02-03-pm-venap	bp-mfs-12-02-03-pm-model	4. Removed upper y-axis limit on Rework graph	MacInnis	12-2-03-PM	
bp-mfs-12-09-03-venap	bp-mfs-12-09-03-model	1. Changed format to be consistent with 600X800 screen setting	MacInnis	12-9-03-PM	

Table 4. BP NPD MFS Improvement Log

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 6

Discussion

This chapter discusses in detail the lessons learned as a result of developing and running the MFS that is hoped to be passed on to managers playing the MFS. This chapter also ties lessons learned in first-hand experience from managing new product development projects in the aerospace industry.

6.1 Lessons to be Learned by Playing the MFS

As discussed in Section 3.2, the role of MFS is not to teach the 'answers' but to change the mental models that currently drive management decisions and facilitate the learning of system dynamic principles as applied to NPD project management. This section describes the lessons to be learned.

6.1.1. Ensure a Realistic and Attainable Schedule

As mentioned before, this is the most important lesson to be learned in NPD Project Management. I have been involved in project management projects where because of the aggressive bidding and winning of new product contracts, the capacity of the product development organizations necessitated that projects be staffed to a *maximum* of only 90% of what was required of the project. Because of the delays associated with staffing these projects, they rarely get staffed even to the 90% level until much later in the project where because of the fire-fighting dynamics people are pulled off other projects to support the failing project. The MFS is very effective in showing the difference in project dynamics that results when even there is a 10% shortfall in staff, as shown in figures 78 and 79, which typify the "90% Syndrome"

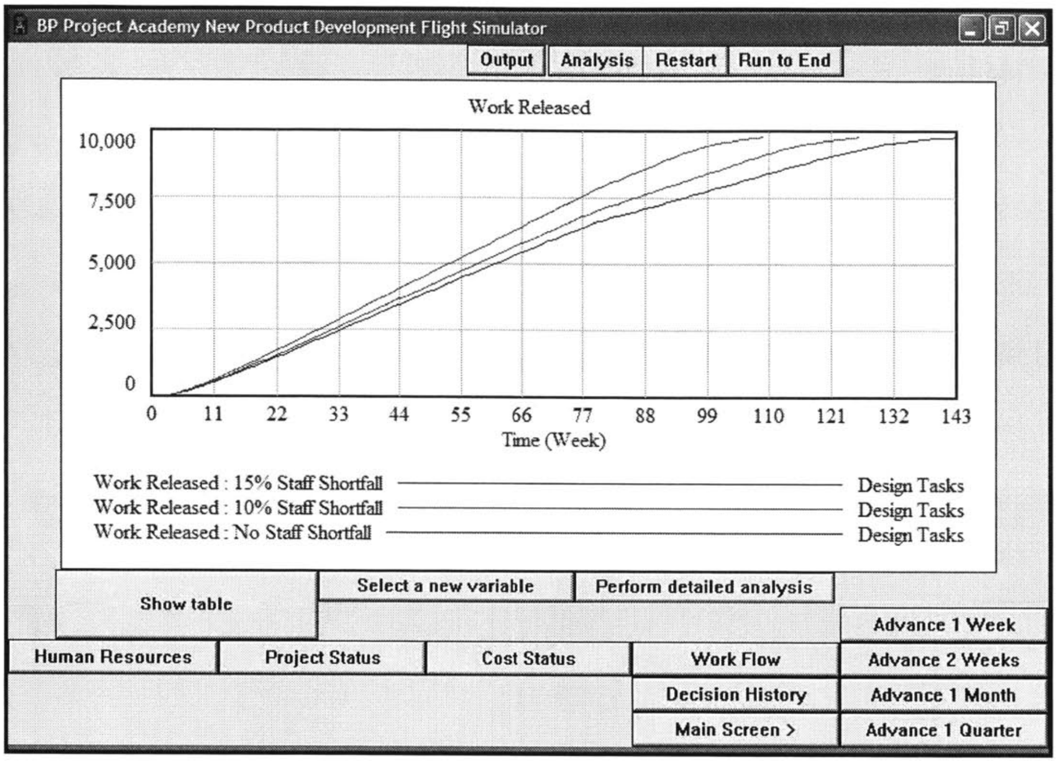

Figure 78. Late To Schedule Effects of Unrealistic Schedules

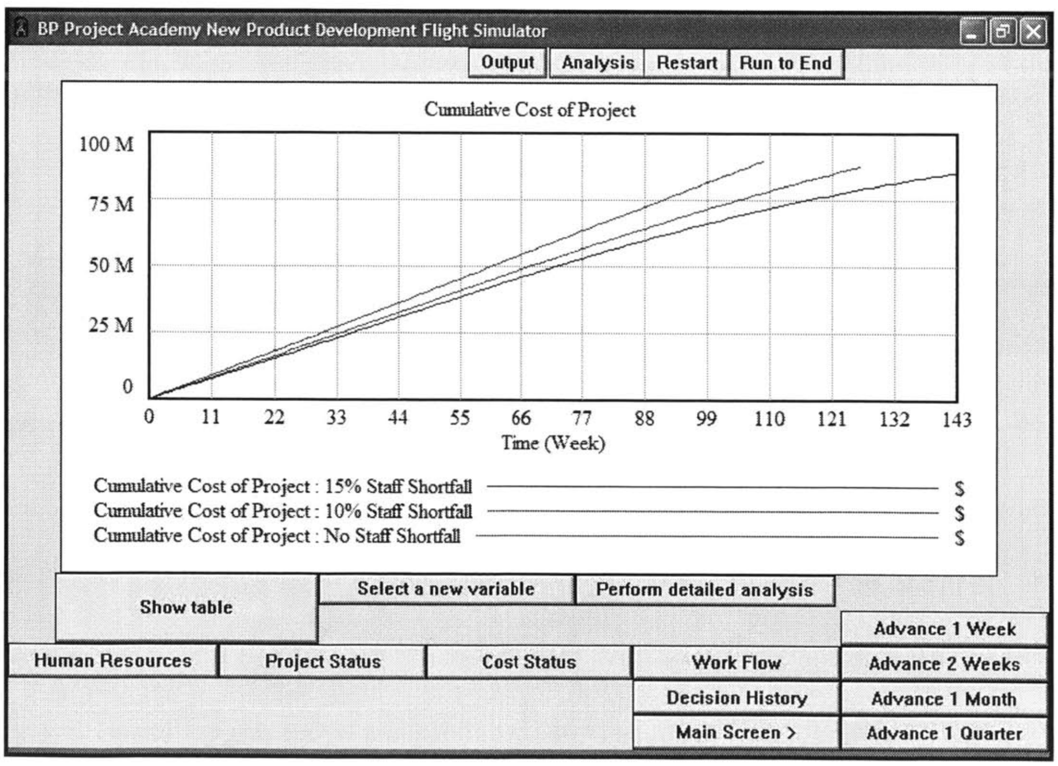

Figure 79. Budget Effects of Unrealistic Schedules

6.1.2. Intuitive Management Actions can Exacerbate Undesirable Project Dynamics

Another lesson to be learned is that intuitive management actions to address late projects can exacerbate the project dynamics to make the situation worse. Figures 80 and 81 show the effects of hiring and overtime which not only significantly drives cost up, but because of the feedback effects that cause higher error rates, slower discovery of rework, and employee turnover, the project takes much longer to complete, further extending the "90% syndrome".

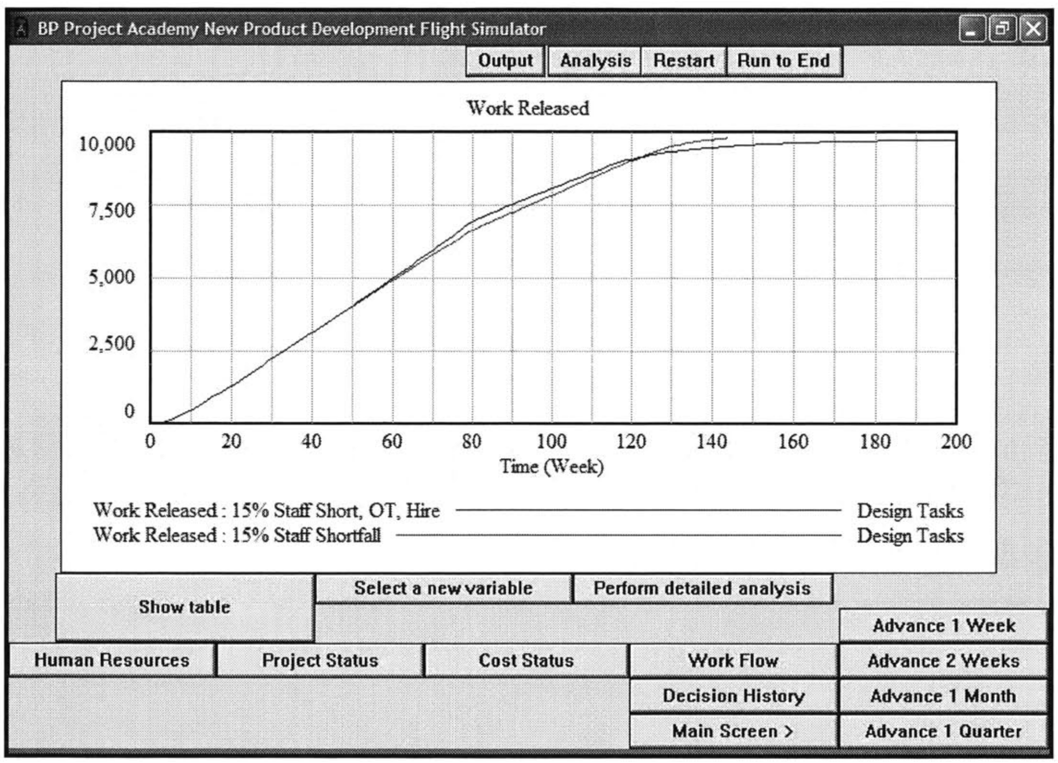

Figure 80. Exacerbating Effects on Schedule of Hiring and Overtime on Late Projects

Figure 82 shows the effects on rework that hiring and overtime have on rework, which was shown in section 3.1 to be the major cause of project failures. My own experience from project management supports this concept. I was involved in one project that had such a major set-back that a very senior manager ordered the transfer of 10 individuals to our project immediately. The response from the customer was favorable, but it only exacerbated the situation much worse.

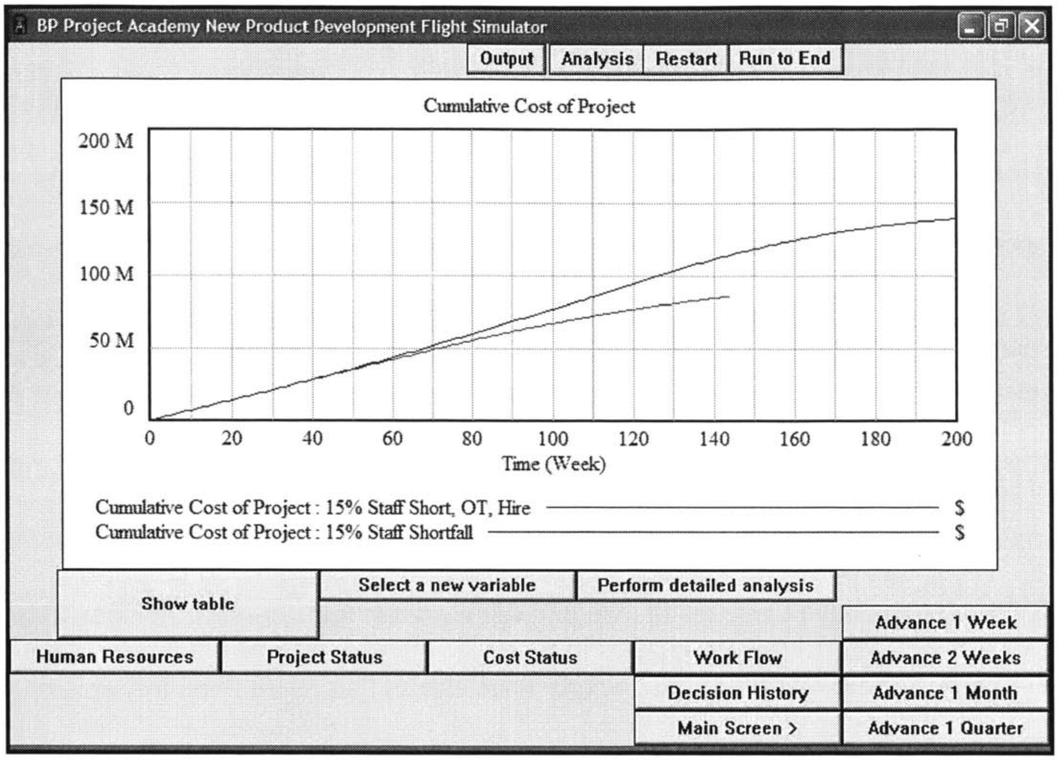

Figure 81. Exacerbating Effects on Budget of Hiring and Overtime on Late Projects

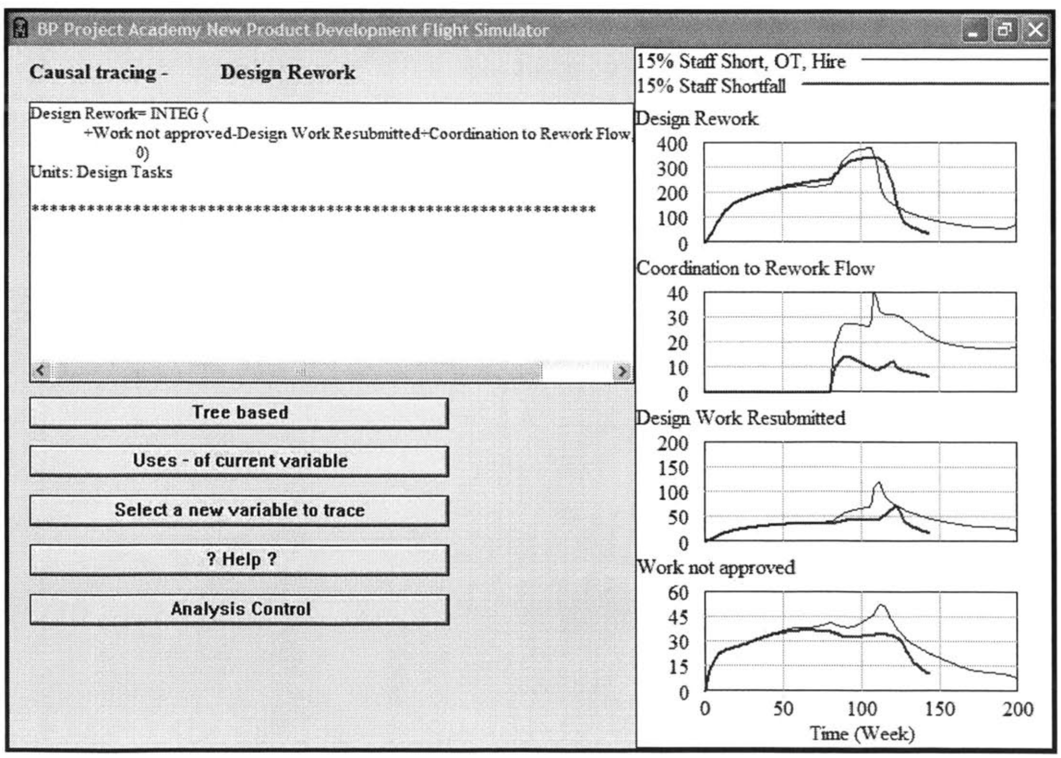

Figure 82. Effects of Management Actions of Hiring and Overtime on Rework

6.1.3. Other Lessons for Understanding Project Dynamics

Another lesson to be learned is that **it is better to slip interim milestones to make a project perform better in the end**. The dynamics of slipping a schedule early to achieve an ultimately sooner and less costly project outcome was shown by comparing the results indicated in Figures 72 and 76. Additionally, **"worse before better" management actions yield better overall results** (the management action to intentionally slip the schedule is an example of this concept). Consider the results from the flight test session of the MFS where players were able to have the project finish significantly sooner at the same cost as those that finished much later. The 'strategy' used to achieve those results were to **aggressively hire personnel early** to allow them to gain experience early to offset the inexperience effects that result from hiring late in a project. Staff was then reduced as the project neared completion. Figure 83 shows the cumulative cost of such a strategy that while showing a much higher initial cost, results in a

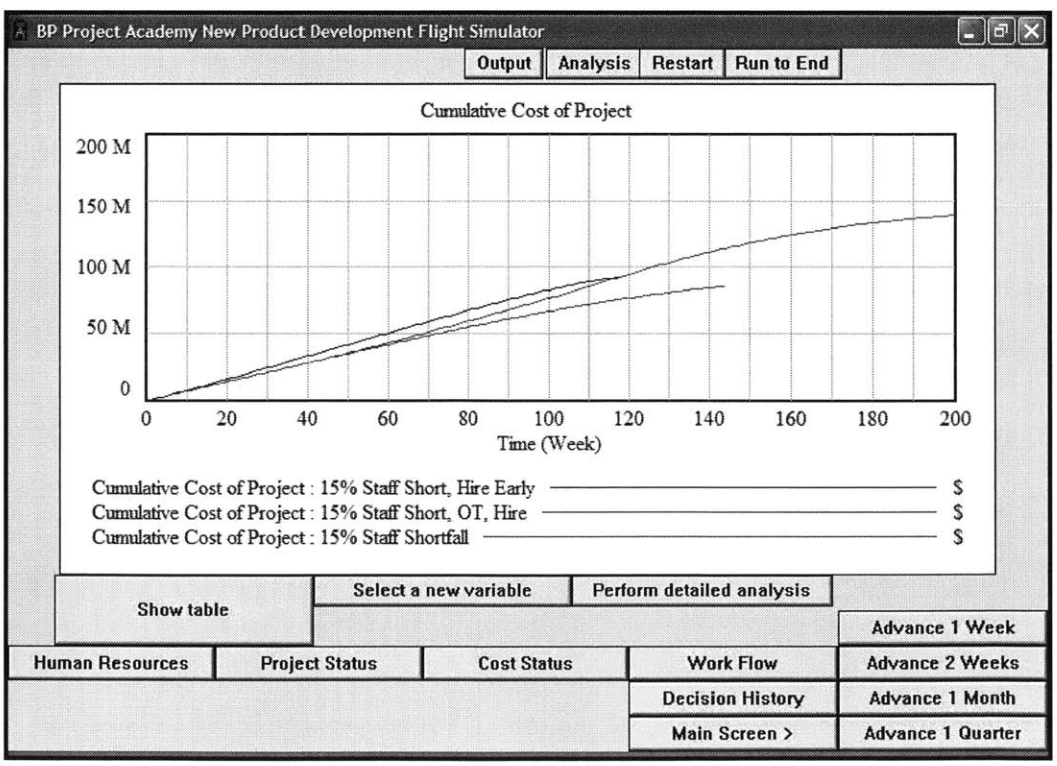

Figure 83. Results of a Hire Early Strategy

project that finishes over a year sooner than the project using traditional late hiring and overtime policy at about 40% less cost, and about 30 weeks sooner than a doing nothing different policy. It is interesting to note that the experienced project managers (myself and Nick McKenna) who played this MFS did not consider hiring so aggressively due to the budget constraints – we were so entrenched with the "better before worse" mental models that preclude such strategies from our NPD project management experience. Other lessons reinforced through playing the simulator are to avoid the use of sustained overtime and hire early.

6.2 Insights from the Role of the MFS in the Learning Process

The MFS provides more value than just teaching the principles of project dynamics, or establishing which strategies will lead to success. It can instigate open discussion amongst the participants on what their mental models are concerning NPD Project Management and that discussion along with teaching of the system dynamics can lead to further insights that were unknown to them. While this thesis does not address the teaching effectiveness of the MFS, the feedback from BP at the initial introduction of the MFS to the BP Project Academy has been very positive. An example of the types of discussion described above was relayed to me by Nelson Repenning who presented a modified version of the MFS to BP. When explaining the model assumptions on productivity for new hires (modeled as the Rookie Discount), one participant said "That's not realistic", whereupon another participant yelled out "It sure is realistic, they should probably be discounted even more". More productive discussions arose from that initial discussion and more misconceptions about each others mental models were exposed and let out for debate (Repenning, 2004). It is this type of interaction that the MFS enables that makes it such an effective organizational learning tool. The next chapter will define recommendations for future work based on my findings and experience in NPD project management.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 7

Future Work

This chapter provides suggestions for future development of NPD MFSs based on the literature and my experience in NPD Project Management.

7.1 Future Development of NPD System Dynamics Models

This section provides suggestions for future development of system dynamics models for NPD Project Management.

7.1.1. Self-imposed Schedule Pressure

The effects of schedule pressure on project dynamics are clear: they result in an increase in work output due to increased effort, but also lead to higher defect rates through cutting corners (out of sequence work), poorer quality (due to speed of work), and less rework discovery. Schedule pressure also drives the use of overtime and fire-fighting that has been shown to further exacerbate the undesirable project dynamics. One factor about schedule pressure, is that it is frequently *self-imposed* by managers who respond to social system forces to perform well. From my experience, any time a project is shown to be behind schedule, the first thing asked by a superior is "When will you be back on schedule." This situation and question have several implications to why projects get trapped in vicious cycles:

- The question and expected answer show linear, non-systems thinking mentality
- The situation is causing social tension that will only be relieved when the project gets back on schedule, regardless of when the finished project is completed.

- The recipient is expected to give a date, usually on the spot, without fully understanding the dependencies that such a date requires. Additionally, due to the concurrency relationships, the recipient is usually not in control of the factors that enable a date to be given, let alone ensure that the commitment can be made to that date.
- The recipient is expected to give a date that is sooner rather than later because the project is already *behind* schedule, so the quicker the project gets back on schedule, the quicker the social tension will be relieved. The sooner the date, the more tension is relieved. The quicker the date is provided, the quicker the tension is relieved. Therefore, the recipient is pressured to give a date *quicker* that is *sooner* than is reasonable.
- Because social tension is relieved, the recipient is rewarded by the senior manager for providing a date quickly that is soon ("we'll be back on track next week").

Thus, the manager has just perpetuated a vicious cycle of self-imposed schedule pressure. He/she must now figure out how they are going to be able to make the commitment they just made. The situation gets worse, because since the date is unrealistic, the likelihood of them being able to meet that commitment is slim. When that commitment is missed, the tension that was relieved before will be even stronger because not only is the project still behind and later into the schedule, but now the *credibility* of the recipient is decreased, providing even more social tension that results in a stronger demand for when the project gets back on track.

Another result of this social tension is an incentive for managers not to reveal how badly the project is doing, a phenomenon that was recently studied in depth by Ford and Sterman that they refer to as "The Liars Club" (Ford and Sterman, 2003). This phenomenon, where managers work together to hide problems from upper management to avoid the above dynamics (as well as the "senior help" dynamics to be suggested in the next section) is shown to exacerbate the rework cycle through delayed rework discovery among other complex dynamics. The same phenomenon was observed in my previous project management organizations by the term "snorkeling", where different departments would "snorkel" at the surface (keep their problems hidden) while other departments in the organization would be getting bit by the sharks down below (having their problems exposed).

7.1.2. Modeling Schedule Pressure Management Actions

Schedule pressure as modeled in system dynamics models refers to the anticipated lateness of a project at completion. I believe the management actions taken in response to schedule pressure impose more direct effects on productivity than previously modeled. In my experience, schedule pressure is manifested through a series of frequent and high level management reviews. Also referred to as "senior help", this is a phenomenon that through manager reviews detracts from the productive ability to do the work. I have been in NPD projects where there were 2 or 3 meetings a day with senior management to brief them on the project status. More time is spent preparing for the review then actually solving the problem and getting the work done. The MFS developed for this thesis was intended to capture these dynamics through management review seniority and frequency decision levers. Like fire-fighting, this is a pervasive management practice in NPD organizations and a system dynamics model capturing the consequences of these actions may help change the mental models that seem to drive their implementation.

7.2 Future Development of NPD Management Flight Simulators

This section provides suggestions for future development of MFSs for NPD Project Management. The first suggestion, which has already been accomplished for the BP Project Academy MFS since the completion of the model developed for this thesis, is to include a front end module to show the importance of front end development activities (Repenning, 2004b). This would satisfy the BP requirements shown in section 4.2.2. and help make managers understand the importance of the front-end activities on overall project performance.

Another recommendation is to incorporate more decision levers that represent management actions a manager can take to manage the project. Percentage of time spent per day by the manager to address integration and dispute resolution was suggested by Nelson Repenning for the initial MFS. Additional management action decision levers should account for the dynamics outlined in Section 7.1, namely, the dynamics of self-imposed schedule pressure and senior management reviews on day to day project management. Figure 84 shows an example simulation screen with these inputs. Additional suggestions include:

- Pop-up messages simulating senior management requests for when the project will be back on schedule. The player should then provide a date when the project will get back on schedule. The model should incorporate dynamics that reinforce the relief in tension (perhaps use a MFS variable that shows the senior manager's approval rating of the manager). The MFS should reflect the correct "better before worse" and/or "worse before better" dynamics, i.e., if the manager puts a date too soon, then approval rating goes up initially but gets worse as the project gets worse and visa versa.

- Incorporate visual color indicators of project status, i.e., RED, GREEN, YELLOW (suggestion by John Sterman in MFS flight tests).
- Incorporate a bid process in MFS to show the dynamics of aggressive project bids (Suggested by Harvard PhD candidate during MFS flight tests. A bid process was incorporated in later version of BP NPD MFS for BP Project Academy by Nelson Repenning).
- Allow players to plan staffing similar to that of projects as shown in Figure 5.
- Incorporate Multi-Project MFS to simulate the "Tipping" effects of fire-fighting on the NPD organization's ability to sustain its long-term capability.

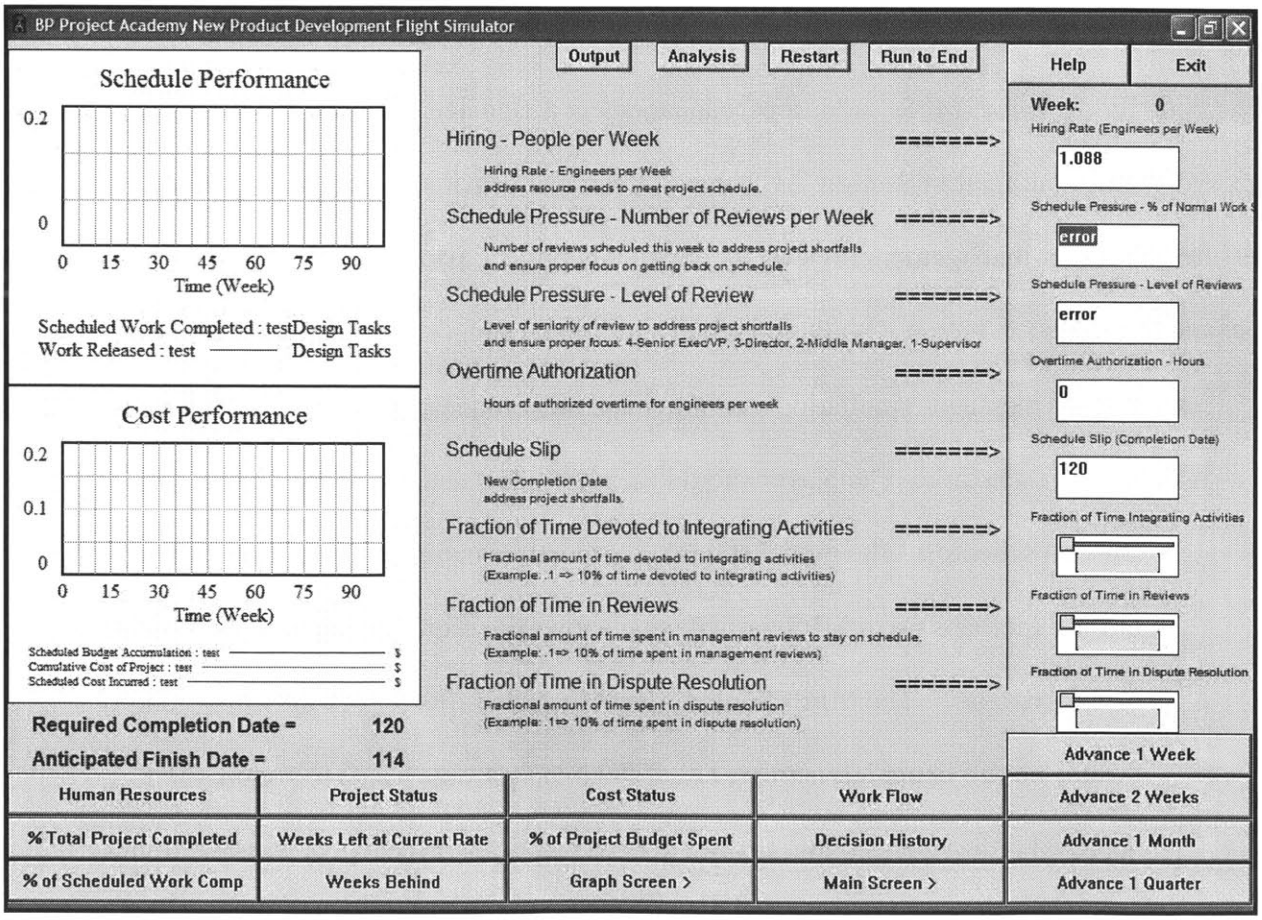

Figure 84. Suggested MFS Input Screen for Additional Management Action Decision Levers

Future work for NPD MFS development includes modeling of additional schedule pressure effects (self-imposed schedule pressure, effects of senior management reviews on productivity), and incorporating more decision levers (frequency and management level of reviews, catch up date, fraction of time spent on different management activities, etc.), and inclusion of multiple phases to stress importance of completing front-end development activities correctly. The next chapter covers conclusions and recommendations.

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 8

Conclusions and Recommendations

This chapter summarizes the conclusions made in this thesis and provides recommendations based on those conclusions.

8.1 The Problem in New Product Development Project Management

New product development (NPD) organizations are plagued with ever increasing occurrences of projects being late to schedule, over-budget, and with inferior performance and quality. Also called the "90% syndrome", this phenomenon is becoming more and more the norm. Partly driven by over-aggressive bidding by the need to remain competitive in mature industries, the problem is exacerbated by continued complexity in products, and in the product development processes and organizations that develop them. A lack of viewing the NPD process as a complex dynamic system has resulted in management actions that exacerbate the situation, despite well intentions. Because the field of System Dynamics is well suited to the understanding of complex systems, it is appropriate for addressing the problems of NPD project management. All system dynamics models of NPD project management use the "rework cycle" to explain why these problems persist.

8.2 The Rework Cycle and Process Concurrence Models

The reason why NPD projects continue to be late and over-budget is because of rework, the repeated accomplishment of tasks until they are satisfactorily completed to meet the needs of upstream phases in the NPD process. The "rework cycle" is a system dynamics model that accounts for rework. The model also incorporates feedback effects of schedule pressure, out-of-sequence work, experience level, and fatigue through overtime that impact productivity, quality, and discovery of rework to further degrade project performance if not prevented. "Knock-On" effects from defects in upstream phases in the NPD process affecting downstream phases makes the situation even worse and explains why projects are late (Reichfelt and Lyneis, 1999).

System dynamics models incorporating the rework cycle have also been developed that recognize that the NPD processes themselves can constrain work accomplishment. These models, called Ford-Sterman models, incorporate internal and external process concurrence relationships that acknowledge that some tasks cannot be accomplished without prior accomplishment of previous tasks (Ford and Sterman, 1998). While these models are sufficient to explain the behavior of NPD project dynamics, they are not sufficient to change the mental models of managers to affect a change in their management actions. Management Flight Simulators (MFS) are necessary tools to facilitate learning and affect change in management actions.

8.3 The Role of Management Flight Simulators

Humans learn best by doing or playing, observing the results of their actions, and then adjusting their behavior based on their observations. The feedback from their actions must be timely and relevant for it to be effective in the learning process (Senge, 1990). The problem with project management in the NPD domain is that managers are unable to observe the results of their actions in a timely or unambiguous way. The separation in time and space due to the complexities and time delays in the NPD process precludes this. MFSs provide a way to compress time and space for managers to observe the feedback of their actions in a timely and relevant way. They also are able to test different hypotheses and observe the effects of those hypotheses, another process that improves learning over traditional methods (Graham, etal, 1994).

Another important way MFSs improve learning is that they provide a way for participants to explore and learn fallacies in the mental models of others in the NPD organization that they would otherwise be unaware of. MFSs enable the free and open dialogue that uncovers these misconceptions in a way no other training vehicle can. They uncover elements of the system and its behavior that would be forever hidden otherwise. Though not intended to teach the "answers", they do expose the player to alternate ways of thinking about the problem and provided insight to the dynamic complexity of the system through a learning process incorporating investigation, conceptualization and framing, and transfer. Through this process, the lessons of project management from a system dynamics perspective can be learned, and applied to other project problems.

8.4 The Project Management Lessons Learned from NPD MFS

The following projects management lessons are learned from application of the BP NPD MFS:

- **Managers must have a realistic and attainable budget and schedule for the projects they manage.**
- **Management actions that are intuitively taken to manage new product development projects may show beneficial results in the short term but will yield undesirable long term results.**
- **Proper management actions for new product development projects often yield a 'worse-before-better' result. Managers must learn that the *right* policy, procedure, or decision may in the short-term look worse than if they had taken the intuitive 'better-before-worse' course of action.**
- **It is better to "get it right the first time" and be late than to be on time and have it wrong.**

8.5 Recommendations

In addition to following the project management lessons learned in section 8.4, it is recommended that management teams of NPD organizations adopt system dynamics based MFS teaching methods to improve the performance of their projects. The following further work in the development of NPD project system dynamic models and MFS is also recommended:

- **Study and model the effects of self-imposed schedule pressure and incorporate in future MFSs.**
- **Study and model the effects of "senior help" management reviews and incorporate in future MFSs.**
- **Incorporate multiple-phases and multiple projects in future MFSs**
- **Incorporate decision levers for decisions project managers make on a daily/weekly basis (fraction of time spent on integrating, dispute resolution, management reviews, etc.)**
- **Incorporate visual indicators of project performance (color status) and pop-up message windows to more realistically reflect project dynamics.**

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 9

References

Ford, David N. and Sterman, John D. 1998. "Dynamic Modeling of Product Development Processes", *System Dynamics Review,* 31-68.

Ford, David N. and Sterman, John D. 2003. "The Liar's Club: Concealing Rework in Concurrent Development", *Concurrent Engineering: Research and Applications*, Volume 11, Number 3, pages 211-219.

Forrester, Jay W. 1961. "Industrial Dynamics", Pegasaus Communications, Waltham, ISBN 1-883823-36-6

Johnson, Scott, 2003. E-mail correspondence and Power Point Presentations on BP Project Academy.

Graham, Alan K., Morecroft, John D. W., Senge, Peter M. and Sterman, John D. 1994. "Model-Supported Case Studies for Management Education", in (Morecroft and Sterman, 1994, pages 219-241).

Lyneis, James, 2002. Course Lecture Notes and discussion in ESD.36J, System Project Management course, MIT, Cambridge, MA.

Lyneis, James, 2003. In-class discussion from lectures at MIT, Cambridge, MA.

Lyneis, James M., Cooper, Kenneth G. and Els, Sharon A., 2001. "Strategic management of complex projects: a case study using system dynamics", *System Dynamics Review*, Vol. 17. No. 3 (Fall 2001): 237-260.

McQuarrie, Allan J. Jr., 2003. "Fire Fighting in Aerospace Product Development: A Study of Project Capacity and Resource Planning in an Aerospace Enterprise", S.M. Thesis for MIT SDM Program, Cambridge, MA.

Morecroft, John D. W. and Sterman, John D., 1994. (editors) "Modeling for Learning Organizations", Productivity Press, Portland, OR, ISBN 1-56327-060-9.

Morrison, Bradley, 2003. In-class discussion from lectures at MIT, Cambridge, MA.

Reichfelt, Kimberley and Lyneis, James. 1999. "The Dynamics of Project Performance: Benchmarking the Drivers of Cost and Schedule Overrun", *European Management Journal* Vol. 17 No. 2, April 1999.

Repenning, Nelson, 2002. "Past the Tipping Point: Understanding Firefighting in New Product Development", Presentation given at 2002 MIT SDM Alumni Conference, MIT Cambridge, MA.

Repenning, Nelson, 2003. Conversations with Nelson Repenning, Cambridge, MA

Repenning, Nelson, 2004. Discussion with Nelson Repenning debriefing initial introduction of MFS to BP Project Academy.

Repenning, Nelson, 2004b. "Project Manager Flight Simulator", Presentation to BP Project Academy, January 2004.

Repenning, Nelson P., Goncalves, Paulo and Black, Laura J. 2001. "Past the Tipping Point: The Persistence of Firefighting in Product Development", *California Management Review*, 43, 4: 44-63.

Repenning, Nelson, Sterman, John and Leach, David, 1995. Total Quality Flight Simulator, MIT, Cambridge, MA.

Senge, Peter M. 1990. "The Fifth Discipline: the Art and Practice of the Learning Organization", Currency Doubleday, New York, ISBN 0-385-26095-4.

Sterman, John D. 1988. "Strategy Dynamics: The Rise and Fall of People Express," Lecture Notes, MIT System Dynamics Working Paper D-3959/3967.

Sterman, John D. 2000. "Business Dynamics: Systems Thinking and Modeling for a Complex World", McGraw-Hill, Boston, ISBN 0-07-231135-5.

Sterman, John D. 2003. Discussions with John Sterman in MIT Sloan School of Management course 15.874 Business Dynamics, Cambridge, MA.

Ulrich, Karl T. and Eppinger, Steven D. 2000. "Product Design and Development", McGraw-Hill, Boston, ISBN 0-07-229647-X.

Utterback, James M. 1994. "Mastering the Dynamics of Innovation", Harvard Business School Press, Boston, ISBN 0-87584-740-4.

THIS PAGE INTENTIONALLY LEFT BLANK

Appenix A

Equation Listing of BP NPD MFS Model

ACWP=
Cumulative Cost of Project
Units: $

Adjustment Delay=
12
Units: Week

Anticipated Design Task Productivity=
Normal Design Task Productivity
Units: Design Tasks/(engineer*Hours)

Anticipated Finish Date=
(1-Project Finish)*(Elapsed Time in Project+Days to Comp at Current Rate)
Units: Week

Anticipated Lateness as Frac of Init Project Duration=
Anticipated Lateness at Current Pace/Initial Completion Date
Units: Dmnl

Anticipated Lateness at Current Pace=
(1-Project Finish)*(Anticipated Finish Date-Completion Date)
Units: Week

Approval Task Productivity=
Normal Approval Task Productivity*Fatigue Multiplier*Schedule Pressure Productivity Multiplier
Units: Design Tasks/(engineer*Hours)

Approval Throughput from tasks=
Design Work Being Checked/Minimum Time for Approval
Units: Design Tasks/Week

Approval Work Capacity from Resources=
Hours to Approval*Approval Task Productivity
Units: Design Tasks/Week

Approval Work Rate=
MIN(Approval Throughput from tasks,Approval Work Capacity from Resources)
Units: Design Tasks/Week

Approved Work Rate=
Tasks Approved with Defects+Tasks without Defects Approved
Units: Design Tasks/Week

Avg Tasks per Week=
Smooth3I(Effective Tasks Per Week,Time to average work week,Normal Tasks per Week
)
Units: **undefined**

Avg Time to Release Tasks=
1
Units: Week

BCWP=
Budgetted Cost per task*Work Released
Units: $

BCWS=
Budgetted Cost per task*Scheduled Work Completed
Units: $

Budget Remaining= INTEG (
-Plan Spend Rate,
Estimated Total Project Cost)
Units: **undefined**

Budgetted Cost per task=
Sanction budget/Initial Tasks
Units: $/Design Tasks

Calculated Completion date= INTEG (
Changes in Completion Date,
Initial Completion Date)
Units: Week

Cap Util Approval=
zidz(Approval Work Rate,Approval Work Capacity from Resources)
Units: Dmnl

Cap Util New Work=
zidz(Design Work Completion Rate,New Work Capacity from Resources)
Units: Dmnl

Cap Util Rework=
zidz(Design Work Resubmitted,Rework Work Capacity from Resources)
Units: Dmnl

Change in Perc Required Budget=
(Indicated Required Remaining Budget-Perceived Required Remaining Budget)/
Time to Adjust Budget Perception
Units: $/Week

Change in Perc Value Earned=
(Indicated Value Earned-Perceived Value Earned)/Time to Adjust Value Perception
Units: Design Tasks/$/Week

Changes in Completion Date= GAME (
(Completion Date Gap/Adjustment Delay)*Willingness to Slip Deadline*(1-Project Finish
))
Units: Week/Week

Chng in Desired Staff=
(1-Project Finish)*(Indicated Desired Project Staff-Desired Project Staff)
/Time to Update Desired Staff
Units: engineer/Week

Chng in Perc Comp Rate=
(Indicated Compeltion Rate-Perceived Comp Rate)/Time To Adj Comp Rate
Units: Design Tasks/Week/Week

Comp Rate Evaluation Period=
Max(Elapsed Time in Project-Schedule Delay,1)
Units: Week

Comp Rate Used To Calculate Days Remaining=
(1-Frac Influence of Per Comp Rate)*Initial Desired Work Rate+Frac Influence of Per Comp Rate
*Perceived Comp Rate
Units: Design Tasks/Week

Completion Date=
GAME(Calculated Completion date)
Units: Week

Completion Date Gap=
Anticipated Finish Date-Calculated Completion date
Units: Week

Coordination Capacity from Tasks=
Tasks to be Coordinated/Minimum Time for Coordination
Units: Design Tasks/Week

Coordination Hours to be Done=
Tasks to be Coordinated/Normal Coordination Task Productivity
Units: engineer*hour

Coordination Task Productivity=
Normal Coordination Task Productivity*Fatigue Multiplier*Schedule Pressure Productivity Multiplier
Units: Design Tasks/(engineer*Hours)

Coordination to Rework Flow=
MIN(Coordination Work Capacity from Resources,Coordination Capacity from Tasks
)
Units: Design Tasks/Week

Coordination Work Capacity from Resources=
Coordination Task Productivity*Hours to Coordination Work
Units: Design Tasks/Week

Cost Variance=
BCWP-ACWP
Units: $

CPI=
XIDZ(BCWP,ACWP,1)
Units: Dmnl

Cumulative Cost of Project= INTEG (
Inc in Cum Cost of Project,
0)
Units: $

Cumulative Expected Value Earned=
Expected Value Earned*Scheduled Budget Accumulation
Units: Design Tasks/Week

Cumulative Perceived Value Earned=
Perceived Value Earned*Cumulative Cost of Project
Units: Design Tasks/Week

"Cumulative spending, per scheduled budget rate of spending"=
(zidz(Cumulative Cost of Project,Scheduled Budget Accumulation))*100
Units: Dmnl

Current work week=
Normal Work Week+Overtime Hours Per Week
Units: hour/(engineer*Week)

Days to Comp at Current Rate=
zidz(Work Remaining,Comp Rate Used To Calculate Days Remaining)
Units: Week

Defect Fraction of Tasks in Coordination=
1
Units: Dmnl

Defects Eliminated by Doing Rework=
Design Work Resubmitted*Frac Defects in Tasks in Rework
Units: Defects/Week

Defects in Tasks Awaiting Release= INTEG (
+Defects in Tasks Passing Approval-Defects in Tasks Being Released,
0)
Units: Defects

Defects in Tasks Being Released=
Work Release Rate*Frac Defective in Tasks Awaiting Release
Units: Defects/Week

Defects in Tasks in Check= INTEG (
Defects Introduced by New Work+Defects Introduced by Rework-Defects in Tasks Passing Approval
-Defects in Tasks Moved to Coordination-Defects in Tasks Moved from Approve to Rework
,
0)
Units: Defects

Defects in Tasks in Coordination= INTEG (
+Defects in Tasks Moved from Release to Coordinate+Defects in Tasks Moved to Coordination
-Defects in Tasks Moved from Coordinate to Rework
,
0)
Units: Defects

Defects in Tasks in Rework= INTEG (
+Defects in Tasks Moved from Approve to Rework+Defects in Tasks Moved from Coordinate to Rework
-Defects Eliminated by Doing Rework,
0)
Units: Defects

Defects in Tasks Moved from Approve to Rework=
Tasks with Defects Moved to Rework/Design Tasks to Defects Ratio
Units: Defects/Week

Defects in Tasks Moved from Coordinate to Rework=
Coordination to Rework Flow*Defect Fraction of Tasks in Coordination
Units: Defects/Week

Defects in Tasks Moved from Release to Coordinate=
DownStream Start Time*Defects in Work Released/DownStream Discovery Time
Units: Defects/Week

Defects in Tasks Moved to Coordination=
0
Units: Defects/Week

Defects in Tasks Passing Approval=
Tasks Approved with Defects/Design Tasks to Defects Ratio
Units: Defects/Week

Defects in Work Released= INTEG (
Defects in Tasks Being Released-Defects in Tasks Moved from Release to Coordinate
,
0)
Units: Defects

Defects Introduced by New Work=
Design Work Completion Rate*Initial Defect Fraction
Units: Defects/Week

Defects Introduced by Rework=
Design Work Resubmitted*ReWork defect fraction
Units: Defects/Week

Defects Per Task in Check=
XIDZ(Defects in Tasks in Check,Design Work Being Checked,Initial Defect Fraction
)
Units: Defects/ Design Tasks

Design Hours to be Done=
Work Available to complete/Normal Design Task Productivity
Units: engineer*hour

Design Rework= INTEG (
+Work not approved-Design Work Resubmitted+Coordination to Rework Flow,
0)
Units: Design Tasks

Design Speed Discovery Multiplier=
Efct of Design Speed on Discovery Table(Pressure Ratio)
Units: **undefined**

Design Speed Error Multiplier=
Efct of Sch Pressure on Error Rate Switch*Efct of Design Speed on Error Rates Table
(Pressure Ratio)+(

1-Efct of Sch Pressure on Error Rate Switch)
Units: Dmnl

Design Task Productivity=
Normal Design Task Productivity*Fatigue Multiplier*Schedule Pressure Productivity Multiplier
Units: Design Tasks/Hours/engineer

Design Tasks to Defects Ratio=
1
Units: Design Tasks/Defects

Design Work Being Checked= INTEG (
Design Work Completion Rate+Design Work Resubmitted-Approved Work Rate-Work not approved
-Discovery of Coordination Work,
0)
Units: Design Tasks

Design Work Completed and Approved= INTEG (
Approved Work Rate-Work Release Rate,
0)
Units: Design Tasks

Design Work Completion Rate=
MIN(New Work Capacity from Resources,New Work Capacity from Concurrence)
Units: Design Tasks/Week

Design Work Resubmitted=
MIN(Rework Work Capacity from Resources, Rework Capacity from tasks)
Units: Design Tasks/Week

Design Work To be Done= INTEG (
New Design Work Initiation Rate-Design Work Completion Rate,
Initial Tasks)
Units: Design Tasks

Desire to Hire New Staff=
0
Units: Dmnl

Desire to Hire New Staff c=
1
Units: Dmnl

Desired Gross Increase in Staff=
GAME(Desired Net Increase in Staff/Time to Hire New Staff+Total Attrition)
Units: engineer/Week

Desired Net Increase in Staff=
(Desired Project Staff-Total FTE Project Staff)*Desire to Hire New Staff
Units: engineer

Desired Project Staff= INTEG (
Chng in Desired Staff,
FTE Starting Engineers)
Units: engineer

Desired Staff Multiplier=
Efct of Sch Pressure on Hiring Table(Sustained Schedule Pressure)
Units: Dmnl

Desired Work Week=
40
Units: hour/Week

Discovery of Coordination Work=
0
Units: Design Tasks/Week

DownStream Discovery Time=
12
Units: Week

DownStream Start Time=
step(1,30)
Units: Week

Efct of Approval Speed on Frac Defects Identifed Table(
[(0,0)-(3,1)],(0,1),(1,1),(1.25,0.856322),(1.5,0.5),(1.75,0.327586),(2,0.281609
),(2.5,0.26),(3,0.25))
Units: Dmnl

Efct of Design Speed on Discovery Table(
[(0,0)-(3,1)],(0,1),(1,1),(1.23345,0.948276),(1.5,0.75),(1.72474,0.568966)
,(2,0.5),(2.49826,0.5),(3,0.5))
Units: Dmnl

Efct of Design Speed on Error Rates Table(
[(0,0)-(2,4)],(0,0.5),(0.407665,0.551724),(0.627178,0.689655),(1,1),(1.25,
1.6),(1.5,2.5),(1.75,2.87356),(2,3),(2.24739,3),(2.4878,3),(3,3))
Units: Dmnl

Efct of Fatigue on Discovery Table(
[(0,0)-(3,1)],(1,1),(1.49477,0.913793),(2.22648,0.54023),(3,0.5))
Units: Dmnl

Efct of Fatigue on Error Switch=
1
Units: Dmnl

Efct of Fatigue on Error Table(
[(0,0)-(2,3)],(0,1),(1,1),(1.26829,1.08621),(1.49826,1.31034),(1.70732,1.81034
),(1.83275,1.94828),(2,2))
Units: Dmnl

Efct of Rookies on Discovery(
[(0,0)-(1,1.5)],(0,1),(0.108014,0.939655),(0.393728,0.310345),(0.5,0.25),(
1,0.25))
Units: Dmnl

Efct of Rookies on Errors Switch=
1
Units: Dmnl

Efct of Rookies on Errors Table(
[(0,0)-(1,4)],(0,1),(0.132404,1.10345),(0.383275,2.8046),(0.5,3),(1,3))
Units: Dmnl

Efct of Sch Pressure on Attrition Rate Table(
[(1,0)-(3,0.4)],(1,0),(1.5,0.0367816),(2.5,0.216092),(3,0.25))
Units: 1/Week

Efct of Sch Pressure on Discovery Switch=
1
Units: Dmnl

Efct of Sch Pressure on Error Rate Switch=
1
Units: Dmnl

Efct of Sch Pressure on Hiring Table(
[(0,0)-(4,2)],(0,0.5),(0.515679,0.62069),(1,1),(1.25,1.08046),(1.6446,1.13793
),(1.97909,1.18391),(3,1.25))
Units: Dmnl

Effect of Approval Speed on Frac Defects Identified=
Efct of Approval Speed on Frac Defects Identifed Table(Ratio Approval Task Prod to Normal
)
Units: Dmnl

Effective Tasks Per Week=
Current work week*Design Task Productivity
Units: **undefined**

Elapsed Time in Project=
Time
Units: Week

Engineer Cost Per Hour=
120
Units: $/hour

Engineering Cost=
OT Staff Cost+Regular Staff Cost
Units: $/Week

Estimated Approval Hours=
Estimated Initial Work/Normal Approval Task Productivity
Units: hour*engineer

Estimated Cost Per Task=
Estimated Total Project Cost/Initial Tasks
Units: **undefined**

Estimated Defect Fraction=
Normal New Work Defect Fraction*Normal Frac Defects Identified
Units: Defects/Design Tasks

Estimated Design Hours=

Estimated Initial Work/Normal Design Task Productivity
Units: hour*engineer

Estimated Engineering Hours=
Estimated Approval Hours+Estimated Design Hours
Units: hour*engineer

Estimated FTE Engineering Hours per Task Completed=
Estimated Engineering Hours/Initial Tasks
Units: **undefined**

Estimated Initial Work=
Initial Tasks/(1-Estimated Defect Fraction*Design Tasks to Defects Ratio)
Units: Design Tasks

Estimated Total Project Cost= INITIAL(
Scheduled Engineering Cost*Initial Completion Date)
Units: **undefined**

Exp Staff Transfered=
MIN(Max(-Desired Gross Increase in Staff,0),Experienced Project Staff/Time to Transfer Exp Staff
)
Units: engineer/Week

Expected Value Earned=
Initial Tasks/Sanction budget
Units: Design Tasks/$

Experience Gain Rate=
New Project Staff/Time to Gain Experience
Units: engineer/Week

Experienced Attrition=
(Experienced Project Staff*Frac Exp Eng Leaving Project)*Willingness to transfer
Units: engineer/Week

Experienced Project Staff= INTEG (
Experience Gain Rate-Experienced Attrition-Exp Staff Transfered,
Initial Exp Staff)
Units: engineer

Fatigue Discovery Multiplier=
Efct of Fatigue on Discovery Table(Tasks per Week Ratio)
Units: **undefined**

Fatigue Error Multiplier=
Efct of Fatigue on Error Table(Tasks per Week Ratio)*Efct of Fatigue on Error Switch
+(1-Efct of Fatigue on Error Switch)
Units: Dmnl

Fatigue Multiplier=
IF THEN ELSE(Fatigue Switch=1,Table for Fatigue Multiplier(Tasks per Week Ratio
), 1)
Units: Dmnl

Fatigue Switch=

1
Units: Dmnl

Fatigue Switch c=
1
Units: Dmnl

FINAL TIME = 200
Units: Week

Frac Available Based on Int Concurrence=
Internal Concurrence(Fraction Perceived Complete)
Units: **undefined**

Frac Defective in Tasks Awaiting Release=
zidz(Defects in Tasks Awaiting Release,Design Work Completed and Approved)
Units: Defects/Design Tasks

Frac Defective in Work Released=
zidz(Defects in Work Released,Work Released)
Units: Defects/Design Tasks

Frac Defects Identified=
Design Speed Discovery Multiplier*Fatigue Discovery Multiplier*Rookie Discovery Multiplier
*Normal Frac Defects Identified
+(1-Efct of Sch Pressure on Discovery Switch)*Normal Frac Defects Identified
Units: Dmnl

Frac Defects in Tasks in Rework=
zidz(Defects in Tasks in Rework,Design Rework)
Units: Defects/Design Tasks

Frac defects in test=
zidz(Tasks with Defects Moved to Rework,Approval Work Rate)
Units: Dmnl

Frac Exp Eng Leaving Project=
Efct of Sch Pressure on Attrition Rate Table(Long Run Schedule Pressure)
Units: 1/Week

Frac Influence of Per Comp Rate=
Influence of Per Comp Rate Table(Fraction Work Completed)
Units: Dmnl

Frac Init Staff Shortfall=
0
Units: Dmnl

Frac Init Staff Shortfall c=
0
Units: Dmnl

Fraction Perceived Complete=
Work Perceived Satisfactory/Initial Tasks
Units: **undefined**

Fraction Resources to Coordination=
zidz(Coordination Hours to be Done,Total Hours to be Done Outstanding)
Units: Dmnl

Fraction Resources to New Work=
zidz(Design Hours to be Done,Total Hours to be Done Outstanding)
Units: Dmnl

Fraction Resources to Rework=
zidz(Rework hours to be Done,Total Hours to be Done Outstanding)
Units: Dmnl

Fraction Tasks Failing Test=
zidz(Work not approved,Approved Work Rate+Work not approved)
Units: Dmnl

Fraction Work Completed=
zidz(Work Released,Initial Tasks)
Units: Dmnl

FTE Starting Engineers=
(1-Starting Engineer Switch)*Init Est FTE Required Engineers*(1-Frac Init Staff Shortfall
)+Starting Engineer Switch*Starting Engineer Choice
Units: engineer

Hiring Rate=
Max(Desired Gross Increase in Staff,0)
Units: engineer/Week

Hours Remaining for other Activities=
Total Available Engineering Hours-Hours to Approval
Units: engineer*hour/Week

Hours to Approval=
MIN(Needed Hours for Approval,Total Available Engineering Hours)
Units: engineer*hour/Week

Hours to Coordination Work=
Fraction Resources to Coordination*Hours Remaining for other Activities
Units: engineer*hour/Week

Hours to New Design Work=
Hours Remaining for other Activities*Fraction Resources to New Work
Units: engineer*hour/Week

Hours to Rework Work=
Hours Remaining for other Activities*Fraction Resources to Rework
Units: engineer*hour/Week

Inc in Cum Cost of Project=
Engineering Cost*(1-Project Finish)
Units: $/Week

Inc in Total Tasks Done=
Design Work Completion Rate+Design Work Resubmitted
Units: Design Tasks/Week

Indicated Budget Growth=
Perceived Total Budget Required/Sanction budget
Units: Dmnl

Indicated Compeltion Rate=
Work Released/Comp Rate Evaluation Period
Units: Design Tasks/Week

Indicated Desired Project Staff=
Desired Project Staff*Desired Staff Multiplier
Units: engineer

Indicated Required Remaining Budget=
Budgetted Cost per task*Work Remaining
Units: $

Indicated Task Completion Schedule=
MIN(Initial Tasks, Initial Desired Work Rate*Time)
Units: Design Tasks

Indicated Value Earned=
zidz(Work Released, Cumulative Cost of Project)
Units: Design Tasks/$

Influence of Per Comp Rate Table(
[(0,-0.006)-(0.25,1)],(0,0),(0.0618467,0.0055632),(0.0949477,0.265736),(0.155052
,0.751391),(0.188153,0.942184),(0.25,1),(1,1))
Units: Dmnl

Init Est FTE Required Engineers=
ST/(Normal Work Week*Unit Engineer)
Units: engineer

Initial Completion Date=
100
Units: Week

Initial Completion Date c=
120
Units: Week

Initial Defect Fraction=
Normal New Work Defect Fraction*Total Error Multiplier
Units: Defects/Design Tasks

Initial Desired Work Rate=
Initial Tasks/(Initial Completion Date-Schedule Delay)
Units: Design Tasks/Week

Initial Exp Staff= INITIAL(
FTE Starting Engineers/(1+Rookie Discount*(Time to Gain Experience*Frac Exp Eng Leaving Project
)))
Units: engineer

Initial New Project Staff= INITIAL(

Initial Exp Staff*(Time to Gain Experience*Frac Exp Eng Leaving Project))
Units: engineer

Initial Project Duration=
Initial Completion Date-Schedule Delay
Units: Week

Initial Tasks=
10000
Units: Design Tasks

Initial Tasks c=
5000
Units: Design Tasks

INITIAL TIME = 0
Units: Week

Internal Concurrence(
[(0,0)-(1,1)],(0,0.1),(0.9,1))
Units: **undefined**

Long Run Avg Work Week=
Smooth3(Current work week,Time to average work week)
Units: hour/(engineer*Week)

Long Run Schedule Pressure=
Smooth3(Sustained Schedule Pressure,Time to Avg Long Run Sch Pressure)
Units: Dmnl

Max Approval Rate=
Design Work Being Checked/Minimum Time for Approval
Units: Design Tasks/Week

Max Error Multiplier=
1/(Normal New Work Defect Fraction*Design Tasks to Defects Ratio)
Units: Dmnl

Minimum Rework Time=
2
Units: Week

Minimum Time for Approval=
6
Units: Week

Minimum Time for Coordination=
3
Units: Week

Minimum Time per Task=
4
Units: Week

Needed Hours for Approval=
Max Approval Rate/Approval Task Productivity

Units: engineer*hour/Week

New Design Work Initiation Rate=
Task Commencement Rate
Units: Design Tasks/Week

New Project Staff= INTEG (
+Hiring Rate-Experience Gain Rate-New Staff Attrition,
Initial New Project Staff)
Units: engineer

New Staff Attrition=
0*(New Project Staff/Time to Transfer)*Willingness to transfer
Units: engineer/Week

New Work Capacity from Concurrence=
Work Available to complete/Minimum Time per Task
Units: Design Tasks/Week

New Work Capacity from Resources=
Design Task Productivity*Hours to New Design Work
Units: Design Tasks/Week

Normal Approval Task Productivity=
0.1
Units: Design Tasks/(engineer*Hours)

Normal Coordination Task Productivity=
0.025
Units: Design Tasks/(engineer*Hours)

Normal Design Task Productivity=
0.025
Units: Design Tasks/(Hours*engineer)

Normal Frac Defects Identified=
0.9
Units: Dmnl

Normal New Work Defect Fraction=
0.25
Units: Defects/Design Tasks

Normal Rework Defect Fraction=
0.25
Units: Defects/Design Tasks

Normal Rookie Fraction=
Initial New Project Staff/(Initial Exp Staff+Initial New Project Staff)
Units: Dmnl

Normal Tasks per Week=
Normal Work Week*Normal Design Task Productivity
Units: **undefined**

Normal Work Rate=

Normal Work Week*(Total FTE Project Staff/Total Normal Hours Per Task)*Unit Engineer
Units: Design Tasks/Week

Normal Work Week=
40
Units: hour/Week/engineer

Normalized Rookie Fraction=
Rookie Fraction
Units: Dmnl

OT Hours=
Max(Current work week-Normal Work Week,0)
Units: hour/(Week*engineer)

OT Premimum=
1
Units: Dmnl

OT Staff Cost=
Total Staff On Project*OT Hours*Engineer Cost Per Hour*OT Premimum
Units: $/Week

Overtime Hours Per Week=
GAME(Max(Normal Work Week*(Overtime Multiplier-1),0))
Units: hour/(Week*engineer)

Overtime Multiplier=
IF THEN ELSE(Overtime Sanction=1, Table for Overtime Multiplier(Sustained Schedule Pressure
), 1)
Units: Dmnl

Overtime Sanction=
1
Units: Dmnl

Overtime Sanction c=
0
Units: Dmnl

Perceived Comp Rate= INTEG (
Chng in Perc Comp Rate,
Initial Desired Work Rate)
Units: Design Tasks/Week

Perceived Required Remaining Budget= INTEG (
Change in Perc Required Budget,
Sanction budget)
Units: $

Perceived Rework Fraction=
SMOOTHI(Fraction Tasks Failing Test, Time to Perceive Defect Fraction , Initial Defect Fraction
*Frac Defects Identified)
Units: **undefined**

Perceived Total Budget Required=

Cumulative Cost of Project+Perceived Required Remaining Budget
Units: $

Perceived Value Earned= INTEG (
Change in Perc Value Earned,
Expected Value Earned)
Units: Design Tasks/$

Percent of Project Budget Spent=
(Cumulative Cost of Project/Sanction budget)*100
Units: Dmnl

Percentage of the Scheduled Work Complete=
XIDZ(Work Released,Scheduled Work Completed, 0)*100
Units: 1

Plan Spend Rate=
Budget Remaining/Time Remaining
Units: **undefined**

Pressure Ratio=
Sustained Schedule Pressure*Schedule Pressure Switch+(1-Schedule Pressure Switch
)
Units: 1

Project Finish=
Project Finish Level
Units: Dmnl

Project Finish Level= INTEG (
IF THEN ELSE(:NOT: Project Finish Level :AND: Work Released >= (Initial Tasks
*0.975),4,0/TIME STEP),
0)
Units: Dmnl

Ratio Approval Task Prod to Normal=
Approval Task Productivity/Normal Approval Task Productivity
Units: Dmnl

Ratio Design Task Prod to Normal=
Design Task Productivity/Normal Design Task Productivity
Units: Dmnl

Ratio of Approval Time to Total=
(1/Normal Approval Task Productivity)/Total Normal Hours Per Task
Units: Dmnl

Regular Staff Cost=
Total Staff On Project*Normal Work Week*Engineer Cost Per Hour
Units: $/Week

Released Work Returned to Coordination=
Defects in Tasks Moved from Release to Coordinate*Design Tasks to Defects Ratio
Units: Design Tasks/Week

Rework Capacity from tasks=

Design Rework/Minimum Rework Time
Units: Design Tasks/Week

ReWork defect fraction=
Normal Rework Defect Fraction*Total Error Multiplier
Units: Defects/Design Tasks

Rework hours to be Done=
Design Rework/Normal Design Task Productivity
Units: engineer*hour

Rework Work Capacity from Resources=
Design Task Productivity*Hours to Rework Work
Units: Design Tasks/Week

Rookie Discount=
0.5
Units: Dmnl

Rookie Discovery Multiplier=
Efct of Rookies on Discovery(Normalized Rookie Fraction)
Units: **undefined**

Rookie Error Multiplier=
Efct of Rookies on Errors Switch*Efct of Rookies on Errors Table(Normalized Rookie Fraction
)+(1-Efct of Rookies on Errors Switch)
Units: Dmnl

Rookie Fraction=
New Project Staff/(Experienced Project Staff+New Project Staff)
Units: Dmnl

Sanction budget=
Scheduled Engineering Cost*Initial Completion Date
Units: $

SAVEPER = 1
Units: Week [0,?]

Schedule Delay=
6
Units: Week

Schedule Pressure=
1+Anticipated Lateness as Frac of Init Project Duration
Units: Dmnl

Schedule Pressure change rate=
(Schedule Pressure-Sustained Schedule Pressure)/Time to average the schedule pressure
Units: 1/Week

Schedule Pressure Productivity Multiplier=
Table for Schedule Pressure Multiplier(Pressure Ratio)
Units: Dmnl

Schedule Pressure Switch=

1
Units: Dmnl

Schedule Pressure Switch c=
1
Units: Dmnl

Schedule Variance=
BCWP-BCWS
Units: $

Scheduled Budget Accumulation=
MIN(Sanction budget,Scheduled Engineering Cost*Time)
Units: $

Scheduled Cost Incurred= INTEG (
Plan Spend Rate,
0)
Units: **undefined**

Scheduled Engineering Cost=
Engineer Cost Per Hour*Total Initial Staff*Normal Work Week
Units: $/Week

Scheduled Percent total work completed=
(Scheduled Work Completed/Initial Tasks)*100
Units: 1

Scheduled Work Comp Rate=
IF THEN ELSE(Time<Schedule Delay , 0, Scheduled Work Remaining/Time Remaining
)
Units: **undefined**

Scheduled Work Completed= INTEG (
Scheduled Work Comp Rate,
0)
Units: Design Tasks

Scheduled Work Remaining= INTEG (
-Scheduled Work Comp Rate,
Initial Tasks)
Units: **undefined**

SPI=
XIDZ(BCWP,BCWS,1)
Units: **undefined**

ST=
Estimated Engineering Hours/Initial Project Duration
Units: hour*engineer/Week

Starting Engineer Choice=
0
Units: **undefined**

Starting Engineer Switch=

0
Units: **undefined**

Sustained Schedule Pressure= INTEG (
Schedule Pressure change rate,
1)
Units: 1

Table for Fatigue Multiplier(
[(0,0)-(2,1)],(0,1),(0.5,1),(1,1),(1.10801,0.91954),(1.31707,0.724138),(1.54704
,0.609195),(2,0.5))
Units: Dmnl

Table for Overtime Multiplier(
[(0,0.8)-(3,2)],(0,1),(0.4,1),(0.6,1),(1,1),(1.2,1.1),(1.4,1.3),(1.6,1.6),
(1.8,1.725),(2,1.75),(2.54007,1.75),(3,1.75))
Units: Dmnl

Table for Schedule Pressure Multiplier(
[(0,0)-(3,2)],(0,0.75),(0.25,0.789474),(0.5,0.833333),(0.752294,0.9),(1,1)
,(1.1682,1.10526),(1.29664,1.35965),(1.40061,1.46),(1.52294,1.5),(1.70642,1.5
),(2,1.5),(3,1.5))
Units: Dmnl

Task Commencement Rate=
0
Units: Design Tasks/Week

Task Completion Schedule=
DELAY FIXED(Indicated Task Completion Schedule,Schedule Delay,0)
Units: Design Tasks

Task Shortfall=
Scheduled Work Completed-Work Released
Units: Design Tasks

Task Shortfall as Percent of Total=
Task Shortfall/Initial Tasks
Units: Dmnl

Tasks Approved with Defects=
Approval Work Rate*Defects Per Task in Check*(1-Frac Defects Identified)*Design Tasks to Defects Ratio
Units: Design Tasks/Week

Tasks per Week Ratio=
Long Run Avg Work Week/Normal Work Week
Units: Dmnl

Tasks to be Coordinated= INTEG (
+Discovery of Coordination Work+Released Work Returned to Coordination-Coordination to Rework Flow
,
0)
Units: Design Tasks

Tasks with Defects Moved to Rework=
Approval Work Rate*Defects Per Task in Check*Frac Defects Identified*Design Tasks to Defects Ratio
Units: Design Tasks/Week

Tasks without Defects Approved=
Approval Work Rate*(1-Defects Per Task in Check)*Design Tasks to Defects Ratio
Units: Design Tasks/Week

Time Remaining=
Max(Completion Date-Time,1)
Units: Week

TIME STEP = 0.25
Units: Week [0,?]

Time To Adj Comp Rate=
26
Units: Week

Time to Adjust Budget Perception=
4
Units: Week

Time to Adjust Value Perception=
12
Units: Week

Time to average the schedule pressure=
4
Units: Week

Time to average work week=
24
Units: Week

Time to Avg Long Run Sch Pressure=
48
Units: Week

Time to Gain Experience=
52
Units: Week

Time to Hire New Staff=
12
Units: Week

Time to Perceive Defect Fraction=
12
Units: Week

Time to Transfer=
52
Units: Week

Time to Transfer Exp Staff=

4
Units: **undefined**

Time to Update Desired Staff=
12
Units: Week

Total Attrition=
Experienced Attrition+New Staff Attrition
Units: engineer/Week

Total Available Engineering Hours=
Current work week*Total FTE Project Staff*Unit Engineer
Units: engineer*hour/Week

Total Error Multiplier=
MIN(Design Speed Error Multiplier*Rookie Error Multiplier*Fatigue Error Multiplier
,Max Error Multiplier)
Units: Dmnl

Total FTE Project Staff=
Experienced Project Staff+New Project Staff*Rookie Discount
Units: engineer

Total Hours to be Done Outstanding=
Coordination Hours to be Done+Design Hours to be Done+Rework hours to be Done
Units: engineer*hour

Total Initial Staff=
Initial Exp Staff+Initial New Project Staff
Units: engineer

Total Normal Hours Per Task=
(1/Normal Approval Task Productivity)+(1/Normal Design Task Productivity)
Units: hour*engineer/ Design Tasks

Total Percent work completed=
(Work Released/Initial Tasks)*100
Units: Dmnl

Total Staff On Project=
Experienced Project Staff+New Project Staff
Units: engineer

Total Tasks Done= INTEG (
Inc in Total Tasks Done,
0)
Units: Design Tasks

Unit Engineer=
1
Units: engineer

Willingness to Slip Deadline=
1
Units: Dmnl

Willingness to Slip Deadline c=
0
Units: Dmnl

Willingness to transfer=
1
Units: Dmnl

Work Available Based on Int Concurrence=
Initial Tasks*Frac Available Based on Int Concurrence
Units: **undefined**

Work Available to complete=
Max(0,Work Available Based on Int Concurrence-Work Initiall Complete)
Units: Design Tasks

Work Estimated Complete=
Design Work Being Checked+Design Work Completed and Approved+Work Released
Units: Design Tasks

Work Initiall Complete=
Initial Tasks-Design Work To be Done
Units: **undefined**

Work not approved=
Tasks with Defects Moved to Rework
Units: Design Tasks/Week

Work Perceived Satisfactory=
Work Released+Design Work Being Checked-Design Work Completed and Approved
Units: Design Tasks

Work Release Rate=
Design Work Completed and Approved/Avg Time to Release Tasks
Units: Design Tasks/Week

Work Released= INTEG (
+Work Release Rate-Released Work Returned to Coordination,
0)
Units: Design Tasks

Work Remaining=
Initial Tasks-Work Released
Units: Design Tasks

THIS PAGE INTENTIONALLY LEFT BLANK

Appenix B

Equation Listing of BP NPD MFS VENAPP

```
!-------------------------------------------------------------------!
! BP NPD FS.vcd        BP New Product Development Flight Simulator
!-------------------------------------------------------------------!
! Dan MacInnis January, 2004
!!COMMAND,"",0,0,0,0,,,SPECIAL>READCUSTOM|ADIgraph.VGD
!-------------------------------------------------------------------!
:SCREEN INTRO
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
!!
COMMAND,,,,,,,,"SPECIAL>SETTITLE|BP Project Academy New Product Development Flight Simulator"
COMMAND,"",0,0,0,0,,, SPECIAL>LOADMODEL|bp-mfs-12-09-03-model.vmf
COMMAND,"",0,0,0,0,,, SPECIAL>READCUSTOM|bp-mfs-12-09-03-graphics.vgd
!
!
!
COMMAND,"",0,0,0,0,,,SETTING>SHOWWARNING|0
TEXTONLY,"Project Management Flight Simulator",0,15,100,20,C|Arial|32|B|0-0-0,,"",
TEXTONLY,"0.1",0,25,100,20,C||12|
TEXTONLY,"Version of  12/09/03-pm",0,30,100,20,C||12|,,"",
TEXTONLY,"This is a pre-release version for demo and testing only",0,35,100,20,C|Arial|12||0-0-0,,"",
TEXTONLY,"- Press Any Key to Continue -",0,65,100,20,C||14|
TEXTONLY,"Designed by Nelson Repenning, Dan MacInnis and Nick McKenna",0,83,100,20,C|Arial|10||0-0-
        0,,"",
TEXTONLY,"Copyright 2003, All Rights Reserved",0,87,100,20,C|Arial|10||0-0-0,,"",
ANYKEY,,,,,,,,INSTRUCT
!
!-------------------------------------------------------------------!
:SCREEN INSTRUCT
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"How to Play",34,6,0,0,C|Times|24||0-0-0,,,
TEXTONLY,"Click on the report name below to see the",9,15,0,0,L|Arial|12||0-0-0,,,
TEXTONLY,"status of your company.  Click on each decision",9,25,0,0,L|Arial|12||0-0-0,,,
TEXTONLY,"(to the right of the screen) and type in your",9,35,0,0,L|Arial|12||0-0-0,,,
TEXTONLY,"entry.  When you are ready to advance to the",9,45,0,0,L|Arial|12||0-0-0,,,
TEXTONLY,"next time period, click on 'Advance 1 Week' or other desired time period",9,55,0,0,L|Arial|12||0-0-\
0,,,
!
TEXTONLY,"To return to the Main Screen from any report, type '<alt> M'",9,65,0,0,L|Arial|10||0-0-0,,,
TEXTONLY,"For help, type '<alt> H'; to exit, type '<alt> X';",9,70,0,0,L|Arial|10||0-0-0,,,
!
LINE,"",75,0,,75,C||||,,"",
!
BUTTON,"Help",80,0,10,5,L,Hh,,MainScreenHelp
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
!
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
SHOWVAR,"Time",90,5,9,6,L|Arial|14|B|0-0-0,,"",
```

```
!
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
RECTANGLE,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
!
!
!
RECTANGLE,"Overtime Hours Per Week",83,49,10,5,H,,"",
!
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
RECTANGLE,"Completion Date",83,71,10,5,H,,"",
!
!
!
!
!
BUTTON,"Advance 1 Week",80,80,20,5,,,"",
!
BUTTON,"Restart",62,0,0,0,,,,Intro
!
!
BUTTON,"Human Resources",0,90,20,5,L,,,
BUTTON,"",0,95,20,5,L,,"",
!
!
BUTTON,"Project Status",20,90,20,5,L,,,
BUTTON,"",20,95,20,5,L,,"",
!
!
BUTTON,"Cost Status",40,90,20,5,L,,,
!
!
BUTTON,"Work Flow",60,90,20,5,L,,,
BUTTON,"Decision History",40,95,20,5,L,,,
BUTTON,"Main Screen >",60,95,20,5,L,,"",
!
!-------------------------------------------------------------------!
BUTTON,"Advance 2 Weeks",80,85,20,5,,,,
BUTTON,"Advance 1 Month",80,90,20,5,,,"",
BUTTON,"Advance 1 Quarter",80,95,20,5,,,"",
!
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
BUTTON,"Click Here to Change Parameters",53,80,30,8,C,,SIMULATE>RUNNAME|?Name for new game
        output,INPUT1
!
BUTTON,"Click Here to Start Game",20,80,30,8,C,,SIMULATE>RUNNAME|?Name for new game
        output,StartGame
!
!--------------------!
:SCREEN INPUT1
SCREENFONT,Arial|10|B|0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Scenario Assumptions",50,5,0,0,C|Arial|24|B|0-0-255,,,
TEXTONLY,"Initial Number of Tasks",13,65,0,0,L,,,
TEXTONLY,"Initial Completion Date (days)",13,74,0,0,L,,,
MODVAR,"Initial Completion Date",58,74,20,0,L,|[50|500],,
TEXTONLY,"Overtime Sanction",13,20,0,0,L,,,
```

```
TEXTONLY,"Schedule Pressure Switch",13,29,0,0,L,,,
TEXTONLY,"Fatigue Switch",13,36,0,0,L,,,
TEXTONLY,"Fraction Initial Staff Shortfall",13,56,0,0,L,,,
SWITCHVAR,"Overtime Sanction",36,19,0,0,,,,
SWITCHVAR,"Schedule Pressure Switch",36,28,0,0,,,,
SWITCHVAR,"Fatigue Switch",36,36,0,0,,,,
MODVAR,"Initial Tasks",58,64,20,0,L,|[1000|10000],,
MODVAR,"Frac Init Staff Shortfall",58,55,20,0,L,,,
TEXTONLY,"Initial Staff (Make sure switch turned on)",13,47,,,L,,"",
MODVAR,"Starting Engineer Choice",58,46,20,,L,,"",
BUTTON,"Record changes and return to Main Screen ",20,86,50,5,L,Rr,,StartGame
!
!
BUTTON,"Return to Main Screen (cancel changes)",20,92,50,5,L,EeXx,"CANCEL",MainScreen
TEXTONLY,"Efct of Sch Pres on Disc Switch",53,19,,,L,,"",
SWITCHVAR,"Efct of Sch Pressure on Discovery Switch",82,19,0,0,,,,
TEXTONLY,"Efct of Sch Pres on Error Rate Switch",53,28,,,L,,"",
SWITCHVAR,"Efct of Sch Pressure on Error Rate Switch",82,27,,,,,"",
TEXTONLY,"Efct of Rookies on Errors Switch",53,37,,,L,,"",
SWITCHVAR,"Efct of Rookies on Errors Switch",82,36,,,,,"",
!
!
!--------------------------------------------!
SWITCHVAR,"Starting Engineer Switch",51,46,,,,,"",
!
:SCREEN StartGame
SCREENFONT,Times New Roman|10||0-0-0|-1--1--1
PIXELPOS,0
COMMAND,"",0,0,,,,,GAME>GAMEINTERVAL|1
COMMAND,"",0,0,,,,,MENU>GAME|0
CLOSESCREEN,"",0,0,0,0,,,,MAINSCREEN
!
!
!--------------------------------!
!
:SCREEN MainScreen
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
!!
TEXTONLY,"Main Screen",67,-5,0,0,L|Arial|14|B|0-0-0,,,
!
!
!
TEXTONLY,"Hiring - People per Week",40,25,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hiring Rate - Engineers per Week",40,31,,,L|Arial|12||0-0-0,,"",
TEXTONLY,"=======>",70,25,0,0,L|Arial|14||0-0-0,,,
!
!
!
TEXTONLY,"Overtime Authorization",40,49,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hours of authorized overtime",40,54,27,6,L|Arial|12||0-0-0,,"",
TEXTONLY,"=======>",70,49,,,L|Arial|14||0-0-0,,"",
!
TEXTONLY,"Schedule Slip",40,71,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"New Completion Date - Weeks",40,76,,,L|Arial|12||0-0-0,,"",
TEXTONLY,"=======>",70,71,,,L|Arial|14||0-0-0,,"",
```

```
!
!
!
!
LINE,"",75,0,,75,C||||,,"",
!
BUTTON,"Help",80,0,10,5,L,Hh,,MainScreenHelp
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
BUTTON,"Restart",61,0,0,0,,,,Intro
!
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
SHOWVAR,"Time",90,6,0,0,L|Arial|14|B|0-0-0,,,
!
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
MODVAR,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
MODVAR,"Overtime Hours Per Week",83,49,10,5,H,,"",
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
MODVAR,"Completion Date",83,71,10,5,H,,"",
!
!
!
!
!
!
!
!
!
BUTTON,"Advance 1
        Week",80,80,20,5,,,"GAME>GAMEINTERVAL|1.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&BRANCH>BR2",
!
!
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH&IFTHENELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
!
!LINE,"",72,0,0,35,C||||
!
!TEXTONLY, "Step", 78,10,,5,C||14|
!TEXTONLY, "Forward", 78,14,,5,C||14|
!
BUTTON,"Human Resources",0,90,20,5,L,,,HumanResource
BUTTON,"",40,95,20,5,L,,"",
!
!
BUTTON,"Project Status",20,90,20,5,L,,,Project
!
!"",1,2,0,0,,,,
BUTTON,"Cost Status",40,90,20,5,L,,,Cost
!"",7,-8,0,0,,,,
BUTTON,"Work Flow",60,90,20,5,L,,,Workflow
BUTTON,"Decision History",60,95,20,5,L,,,RecDecisions
```

```
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1 Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON",TESTING
BUTTON,"",20,95,20,5,L,,"",
BUTTON,"",0,95,20,5,L,,"",
WIPTOOL,"GR1",0,1,33,39,4,,CUSTOM>WIP1,
WIPTOOL,"GR2",0,40,33,37,,,CUSTOM>WIP2,
!
!
!COMMAND,"",0,0,0,0,,,IFTHENELSE&TEST>Project Finish=1&Branch>BR1&Branch>BR2
!BRANCH,"BR1",0,0,0,0,,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now clic\
!k the O!K button to go to the Status Screen
!BRANCH,"BR2",0,0,0,0,,,,Continue
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
TEXTONLY,"Required Completion Date = ",2,78,,,|Arial|12|B|0-0-0,,"",
TEXTONLY,"Anticipated Finish Date = ",2,82,0,0,|Arial|12|B|0-0-0,,,
SHOWVAR,"Completion Date%8.0f",27,78,,,|Arial|12|B|0-0-0,,"",
SHOWVAR,"Anticipated Finish Date%8.0f",27,82,,,|Arial|12|B|0-0-0,,"",
!
:SCREEN MainScreenHel
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Help for Main Screen",0,15,100,20,C||18|
TEXTONLY,"To return to the Main Screen from any report, type '<alt> M'",15,75,,,L||10|
TEXTONLY,"For help, type '<alt> H'; to exit, type '<alt> X';",15,80,,,L||10|
ANYKEY,,,,,,,,MainScreen
!
!--------------------------------------------------------------------!
:SCREEN TESTING
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
COMMAND,"",0,0,0,0,,,IFTHENELSE&TEST>Project Finish Level=0&Branch>BR1&Branch>BR2
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,MainScreen
!
!
!--------------------------------------------------------------------!
:SCREEN Cost
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
LINE,"",80,0,0,75,C||||
!
BUTTON,"Help",80,0,10,5,L,Hh,"",CostStatHelp
```

```
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
!
!
BUTTON,"Advance 1
        Week",80,80,20,5,,,"GAME>GAMEINTERVAL|1.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2"",
!
!
!
!
!
!
!
!
TEXTONLY,"Cost Status",4,0,0,0,L|Arial|14|B|0-0-0,,,
!
!
!
!
!---------------------------------------------------------------------!
BUTTON,"",0,95,20,5,L,,"",
BUTTON,"",20,95,20,5,L,,"SPECIAL>SETWBITEM|&WORKBENCH>TABLE&WORKBENCH>GRAPH",
BUTTON,"Estimated Total",40,95,20,5,L,,"SPECIAL>SETWBITEM|Perceived Total Budget
        Required&WORKBENCH>GRAPH",
!
TEXTONLY,"Cumulative Cost of Project ($)",9,20,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Cumulative Cost of Project",48,20,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Engineering Cost ($/week))",9,25,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Engineering Cost",48,25,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Regular Staff Cost ($/week)",9,30,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Regular Staff Cost",48,30,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Overtime Staff Cost ($/week)",9,35,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"OT Staff Cost",48,35,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Overtime Hours",9,40,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"OT Hours%8.0f",48,39,0,0,R|Arial|10||0-0-0,,,
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
WIPTOOL,"WIP2",2,44,40,40,,,CUSTOM>WIP2,
WIPTOOL,"EVMS",57,43,22,41,,,CUSTOM>EVMS,
TEXTONLY,"Budget",30,12,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Actual",43,12,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Estimated Total",67,6,13,12,|Arial|14|B|0-0-0,,,
SHOWVAR,"Scheduled Budget Accumulation",31,20,0,0,,,,
!
SHOWVAR,"Perceived Total Budget Required",75,20,0,0,R|Arial|10||0-0-0,,,
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
>Project Finish Level=0&Branch>BR1&Branch>BR2",
```

```
SHOWVAR,"Plan Spend Rate",31,25,0,0,,,,
SHOWVAR,"Plan Spend Rate",31,30,0,0,,,,
BUTTON,"Human Resources",0,90,20,5,L,,,HumanResource
BUTTON,"Project Status",20,90,20,5,L,,,Project
BUTTON,"Cost Status",40,90,20,5,L,,,Cost
BUTTON,"Work Flow",60,90,20,5,L,,,Workflow
BUTTON,"Decision History",40,95,20,5,L,,,RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH,
!
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
!
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
TEXTONLY,"Sanctioned Budget",53,6,14,12,|Arial|14|B|0-0-0,,,
SHOWVAR,"Sanction budget",61,20,0,0,R|Arial|10||0-0-0,,,
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
TEXTONLY,"SPI = ",42,46,9,12,|Arial|14|B|0-0-0,,"",
SHOWVAR,"SPI",55,51,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"CPI = ",42,66,9,12,|Arial|14|B|0-0-0,,"",
SHOWVAR,"CPI",55,71,0,0,R|Arial|10||0-0-0,,,
!
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
MODVAR,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
MODVAR,"Overtime Hours Per Week",83,49,10,5,H,,"",
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
MODVAR,"Completion Date",83,71,10,5,H,,"",
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
SHOWVAR,"Time",90,6,0,0,L|Arial|14|B|0-0-0,,,
BUTTON,"",20,95,20,5,L,,"SPECIAL>SETWBITEM|&WORKBENCH>TABLE&WORKBENCH>GRAPH",
!
:SCREEN CostStatHelp
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Help for Cost Status",0,15,100,20,C||18|,,"",
TEXTONLY,"To return to the Main Screen from any report, type '<alt> M'",15,75,,,L||10|
TEXTONLY,"For help, type '<alt> H'; to exit, type '<alt> X';",15,80,,,L||10|
ANYKEY,,,,,,,,Cost
!
!--------------------------------------------------------------------!
:SCREEN Project
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
!!
!!
LINE,"",80,0,0,75,C||||
!
BUTTON,"Help",80,0,10,5,L,Hh,"",ProjectHelp
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
!
!
```

```
!
BUTTON,"Advance 1
        Week",80,80,20,5,,,"GAME>GAMEINTERVAL|1.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
!
!
!SETWBITEM|Task Shortfall&WORKBENCH>TABLE&WORKBENCH>GRAPH"
!
!
BUTTON,"",0,95,20,5,L,,"",
!
!
!
!
TEXTONLY,"Project Status",10,0,,,L||14|B|,,"",
!
!
TEXTONLY,"Cost Performance Index (CPI)",45,11,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"CPI",77,11,,,R|Arial|10||0-0-0,,"",
!
!
TEXTONLY,"Schedule Performance Index (SPI)",45,17,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"SPI",77,17,,,R|Arial|10||0-0-0,,"",
!
TEXTONLY,"% Total Project Completed",45,23,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Total Percent work completed",77,23,0,0,R|Arial|10||0-0-0,,,
!
TEXTONLY,"% of Scheduled Work Completed",45,30,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Percentage of the Scheduled Work Complete",77,30,0,0,R|Arial|10||0-0-0,,,
!
!-------------------------------------------------------------------!
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
TEXTONLY,"Estimated Total Budget",2,34,,,L|Arial|10||0-0-0,,"",
TEXTONLY,"Work Released",2,17,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Work Released%8.0f",42,17,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Work Remaining (tasks)",2,11,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Work Remaining%8.0f",42,12,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Budget to Date",2,28,0,0,L|Arial|10||0-0-0,,,
WIPTOOL,"GR4",0,38,39,51,,,CUSTOM>WIP4,
!
TEXTONLY,"Actual",36,5,,,|Arial|14|B|0-0-0,,"",
TEXTONLY,"Scheduled",22,5,,,|Arial|14|B|0-0-0,,"",
SHOWVAR,"Scheduled Work Remaining%8.0f",30,12,,,R|Arial|10||0-0-0,,"",
SHOWVAR,"Scheduled Work Completed%8.0f",30,17,,,R|Arial|10||0-0-0,,"",
WIPTOOL,"GRWork Release",40,38,39,51,,,CUSTOM>WIP5,
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
```

```
>Project Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Human Resources",0,90,20,5,L,,,HumanResource
BUTTON,"Project Status",20,90,20,5,L,,,Project
BUTTON,"Cost Status",40,90,20,5,L,,,Cost
BUTTON,"Work Flow",60,90,20,5,L,,,Workflow
BUTTON,"Decision History>",40,95,20,5,L,,"",RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH,
!
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
!
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
TEXTONLY,"Completion Date",2,22,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Completion Date%8.0f",30,22,,,R|Arial|10||0-0-0,,"",
SHOWVAR,"Anticipated Finish Date%8.0f",42,22,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
MODVAR,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
MODVAR,"Overtime Hours Per Week",83,49,10,5,H,,"",
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
MODVAR,"Completion Date",83,71,10,5,H,,"",
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
!
SHOWVAR,"Time",90,6,0,0,L|Arial|14|B|0-0-0,,,
BUTTON,"% Work Failing Checking",20,95,20,5,L,,"SPECIAL>SETWBITEM|Fraction Tasks Failing
        Test&WORKBENCH>GRA\
PH",
SHOWVAR,"Cumulative Cost of Project",42,29,0,0,R|Arial|10||0-0-0,,,
SHOWVAR,"Scheduled Budget Accumulation",30,29,,,R,,"",
SHOWVAR,"Perceived Total Budget Required",42,35,0,0,R|Arial|10||0-0-0,,,
SHOWVAR,"Sanction budget",30,35,,,R|Arial|10||0-0-0,,"",
!
:SCREEN ProjectHelp
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Help for Project Status",0,15,100,20,C||18|,,"",
TEXTONLY,"To return to the Main Screen from any report, type '<alt> M'",15,75,,,L||10|
TEXTONLY,"For help, type '<alt> H'; to exit, type '<alt> X';",15,80,,,L||10|
ANYKEY,,,,,,,,Project
!
!-------------------------------------------------------------------!
:SCREEN HumanResource
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
!!
!!
LINE,"",80,0,0,75,C|||
!
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
!
```

```
!
!
!
!
!
!
!
!
!
BUTTON,"Advance 1
        Week",80,80,20,5,,,"GAME>GAMEINTERVAL|1.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
!
!BUTTON,"Advance 1
        Year",80,75,20,5,,,GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH
!
!BUTTON,"Advance to
        End",80,80,20,5,,,GAME>GAMEINTERVAL|120.0&Game>GAMEON&SPECIAL>REFRESH
!
!LINE,"",72,0,0,35,C||||
!
!TEXTONLY, "Step", 78,10,,5,C||14|
!TEXTONLY, "Forward", 78,14,,5,C||14|
!
BUTTON,"Average Work Week",0,95,20,5,L,,SPECIAL>SETWBITEM|Long Run Avg Work
        Week&WORKBENCH>GRAPH,
!
BUTTON,"Total FTE Project Staff",14,109,20,5,L,,SPECIAL>SETWBITEM|Total FTE Project
        Staff&WORKBENCH>GRAPH,
BUTTON,"Rookie Fraction",20,95,20,5,L,,SPECIAL>SETWBITEM|Rookie Fraction&WORKBENCH>GRAPH,
!
!
!
!
!
TEXTONLY,"Human Resources",10,0,,,L||14|B|
!
TEXTONLY,"Experienced Project Staff (Engineers)",5,14,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Experienced Project Staff%8.0f",48,14,0,0,R|Arial|10||0-0-0,,,
!
TEXTONLY,"Hiring Rate (Engineers/week)",5,27,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Hiring Rate%8.0f",48,27,0,0,R|Arial|10||0-0-0,,,
!
!
!
!
!
!TEXTONLY, "Product Development",55,10,,,L||10|B|
!
!
!
!
!
!
!
```

```
!TEXTONLY, "Administration",12,10,,,L||10|B|
!
!
!
!
!
!
!--------------------------------------------------------------------!
TEXTONLY,"Total Project Staff (Engineers)",5,22,0,0,L|Arial|10|B|0-0-0,,,
SHOWVAR,"Total Staff On Project%8.0f",48,22,0,0,R|Arial|10|B|0-0-0,,,
TEXTONLY,"New Project Staff (Engineers)",5,9,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"New Project Staff%8.0f",48,9,0,0,R|Arial|10||0-0-0,,,
TEXTONLY,"Experienced Attrition Rate (Engineers/Week)",5,32,0,0,L|Arial|10||0-0-0,,,
SHOWVAR,"Experienced Attrition%8.0f",48,32,0,0,R|Arial|10||0-0-0,,,
WIPTOOL,"GR3",0,40,79,49,,,CUSTOM>WIP3,
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
SHOWVAR,"Long Run Avg Work Week%8.1f",44,37,0,0,,,,
!
TEXTONLY,"Average Work Week (Hours/eng-week)",5,37,0,0,L|Arial|10||0-0-0,,,
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
>Project Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Human Resources",0,90,20,5,L,,,HumanResource
BUTTON,"Project Status",20,90,20,5,L,,,Project
BUTTON,"Cost Status",40,90,20,5,L,,,Cost
BUTTON,"Work Flow",60,90,20,5,L,,,Workflow
BUTTON,"Decision History>",40,95,20,5,L,,,RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH,
!
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
!
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
SHOWVAR,"Initial Exp Staff%8.0f",39,14,,,R|Arial|10||0-0-0,,"",
SHOWVAR,"FTE Starting Engineers%8.0f",39,22,,,R|Arial|10|B|0-0-0,,"",
SHOWVAR,"Initial New Project Staff%8.0f",39,9,,,R|Arial|10||0-0-0,,"",
TEXTONLY,"Initial",35,5,0,0,|Arial|12|B|0-0-0,,,
TEXTONLY,"Current",44,5,,,|Arial|12|B|0-0-0,,"",
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
MODVAR,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
MODVAR,"Overtime Hours Per Week",83,49,10,5,H,,"",
```

```
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
MODVAR,"Completion Date",83,71,10,5,H,,"",
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
SHOWVAR,"Time",90,6,0,0,L|Arial|14|B|0-0-0,,,
!
:SCREEN HumanResHelp
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Help for Human Resources",0,15,100,20,C||18|
TEXTONLY,"To return to the Main Screen from any report, type '<alt> M'",15,75,,,L||10|
TEXTONLY,"For help, type '<alt> H'; to exit, type '<alt> X';",15,80,,,L||10|
ANYKEY,,,,,,,,HumanResource
!
!-------------------------------------------------------------------!
:SCREEN RecDecisions
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
!!
!!
LINE,"",80,0,0,75,C|||
!
BUTTON,"Help",80,0,10,5,L,Hh,"",DecisionHelp
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
!
BUTTON,"Advance 1
        Week",80,80,20,5,,,"GAME>GAMEINTERVAL|3.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
!
!BUTTON,"Advance 1
        Year",80,75,20,5,,,GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH
!
BUTTON,"Advance to
        End",80,80,20,5,,,GAME>GAMEINTERVAL|120.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
```

```
!LINE,"",72,0,0,35,C|||
!
!TEXTONLY, "Step", 78,10,,5,C||14|
!TEXTONLY, "Forward", 78,14,,5,C||14|
!
BUTTON,"Hiring Rate:Desired Staff Increase",48,25,30,5,L,,"SPECIAL>SETWBITEM|Hiring
        Rate&WORKBENCH>GRAPH",
!
!
!
!
!
!
!
!
!
BUTTON,"Overtime Hours Per Week",48,49,30,,L,,"SPECIAL>SETWBITEM|Overtime Hours Per
        Week&WORKBENCH>GRAPH",
BUTTON,"Schedule Slip:Completion Date",48,71,30,5,L,,"SPECIAL>SETWBITEM|Completion
        Date&WORKBENCH>GRAPH",
!
TEXTONLY,"Record of Decisions",10,0,,,L||14|B|,,"",
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
!
!
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
>Project Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Human Resources",0,90,20,5,L,,,HumanResource
BUTTON,"Project Status",20,90,20,5,L,,,Project
BUTTON,"Cost Status",40,90,20,5,L,,,Cost
BUTTON,"Work Flow",60,90,20,5,L,,,Workflow
BUTTON,"Decision History>",40,95,20,5,L,,"",RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH&IFTHENELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
!
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
!
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
```

```
MODVAR,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
MODVAR,"Overtime Hours Per Week",83,49,10,5,H,,"",
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
MODVAR,"Completion Date",83,71,10,5,H,,"",
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
TEXTONLY,"Hiring - People per Week",5,25,0,0,L|Arial|14|B|0-0-0,,,
TEXTONLY,"Hiring Rate - Engineers per Week",5,31,0,0,L|Arial|12||0-0-0,,,
TEXTONLY,"=======>",35,25,0,0,L|Arial|14||0-0-0,,,
TEXTONLY,"Overtime Authorization",5,49,0,0,L|Arial|14|B|0-0-0,,,
TEXTONLY,"Hours of authorized overtime",5,54,27,6,L|Arial|12||0-0-0,,,
TEXTONLY,"=======>",35,49,0,0,L|Arial|14||0-0-0,,,
TEXTONLY,"Schedule Slip",5,71,0,0,L|Arial|14|B|0-0-0,,,
TEXTONLY,"New Completion Date - Weeks",5,76,0,0,L|Arial|12||0-0-0,,,
TEXTONLY,"=======>",35,71,0,0,L|Arial|14||0-0-0,,,
SHOWVAR,"Time",90,6,0,0,L|Arial|14|B|0-0-0,,,
BUTTON,"",20,95,20,5,L,,"",
BUTTON,"",0,95,20,5,L,,,
!
:SCREEN DecisionHelp
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Help for Decision Records Screen",0,15,100,20,C||18|,,"",
TEXTONLY,"To return to the Main Screen from any report, type '<alt> M'",15,75,,,L||10|
TEXTONLY,"For help, type '<alt> H'; to exit, type '<alt> X';",15,80,,,L||10|
ANYKEY,,,,,,,,RecDecisions
!
:SCREEN Workflow
SCREENFONT,Arial|10||0-0-0|192-255-192
PIXELPOS,0
!!
!!
!!
!!
!!
!!
!!
!!
!!
!!
!!
!!
!!
LINE,"",80,0,0,75,C||||
!
BUTTON,"Help",80,0,10,5,L,Hh,,MainScreenHelp
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
!
!
!
!
!
!
!
!
!
!
```

```
!
BUTTON,"Advance 1
        Week",80,80,20,5,,,"GAME>GAMEINTERVAL|1.0&Game>GAMEON&IFTHENELSE&TEST>Project
        Finish Leve\
l=0&Branch>BR1&BRANCH>BR2",
!
!BUTTON,"Advance 1
        Year",80,75,20,5,,,GAME>GAMEINTERVAL|12.0&Game>GAMEON&IFTHENELSE&TEST>Project
        Fini\
sh Level=0&Branch>BR1&Branch>BR2",
!
!
!LINE,"",72,0,0,35,C|||
!
!TEXTONLY, "Step", 78,10,,5,C||14|
!TEXTONLY, "Forward", 78,14,,5,C||14|
!
!
!
!
!
!
!
!
!
!--------------------------------------------------------------------!
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
>Project Finish Level=0&Branch>BR1&Branch>BR2",
TEXTONLY,"Uncompleted Tasks Remaining",16,8,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Tasks Requiring Testing",16,14,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Tasks Requiring Rework",16,19,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Tasks Awaiting Approval",16,24,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Tasks Released",16,30,0,0,|Arial|14|B|0-0-0,,,
!
!TEXTONLY
SHOWVAR,"Design Work To be Done%8.0f",56,9,0,0,|Arial|14|B|0-0-0,,,
SHOWVAR,"Design Work Being Checked%8.0f",56,14,0,0,|Arial|14|B|0-0-0,,,
SHOWVAR,"Design Rework%8.0f",56,19,0,0,|Arial|14|B|0-0-0,,,
SHOWVAR,"Design Work Completed and Approved%8.0f",56,24,0,0,|Arial|14|B|0-0-0,,,
SHOWVAR,"Work Released%8.0f",56,30,0,0,|Arial|14|B|0-0-0,,,
WIPTOOL,"New_Tasks",0,52,25,32,,,"CUSTOM>New_Tasks",
WIPTOOL,"WIP1",53,52,25,32,,,"CUSTOM>WIP1",
WIPTOOL,"Rework",26,52,26,32,,,CUSTOM>Rework,
TEXTONLY,"Fraction of Tasks Failing Testing",16,39,0,0,|Arial|14|B|0-0-0,,,
SHOWVAR,"Perceived Rework Fraction",56,39,0,0,|Arial|14|B|0-0-0,,,
BUTTON,"Human Resources",0,90,20,5,L,,,HumanResource
```

```
BUTTON,"Project Status",20,90,20,5,L,,,Project
BUTTON,"Cost Status",40,90,20,5,L,,,Cost
BUTTON,"Work Flow",60,90,20,5,L,,,Workflow
BUTTON,"Decision History",40,95,20,5,L,,,RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH,
!
BUTTON,"Analysis",51,0,0,0,,,,ANALYSIS
!
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
!
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
MODVAR,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
MODVAR,"Overtime Hours Per Week",83,49,10,5,H,,"",
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
MODVAR,"Completion Date",83,71,10,5,H,,"",
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
SHOWVAR,"Time",90,6,0,0,L|Arial|14|B|0-0-0,,,
BUTTON,"",20,95,20,5,L,,"",
BUTTON,"",0,95,20,5,L,,,
!
:SCREEN OUTPUT1
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
PIXELPOS,0
TOOL,"GR1",5,5,90,70,,,WORKBENCH>Graph
BUTTON,"Show table",17,75,25,10,C,,,OUTPUT2
!BUTTON,"Modify and Rerun Last Scenario",75,82,0,6,L,,SIMULATE>READRUNCHG|!,SETUPSIM
BUTTON,"Select a new variable",42,75,25,0,C,,SPECIAL>VARSELECT|New variable to use,OUTPUT1
BUTTON,"Perform detailed
        analysis",67,75,25,0,C,,SPECIAL>ALIASSCREEN|ARETURN|OUTPUT1,ANALYSIS
BUTTON,"Advance 1
        Week",80,80,20,5,,,GAME>GAMEINTERVAL|1.0&Game>GAMEON&IFTHENELSE&TEST>Project
        Finish Level\
=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
>Project Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Human Resources",0,85,20,5,L,,"",HumanResource
BUTTON,"Project Status",20,85,20,5,L,,"",Project
BUTTON,"Cost Status",40,85,20,5,L,,"",Cost
```

```
BUTTON,"Work Flow",60,85,20,5,L,,"",Workflow
BUTTON,"Decision History",60,90,20,5,L,,"",RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH&IFTHENELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
!
:SCREEN OUTPUT2
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
PIXELPOS,0
TOOL,"GR1",5,5,90,70,,,WORKBENCH>Table
BUTTON,"Show graph",17,75,25,10,C,,,OUTPUT1
BUTTON,"Select a new variable",42,75,25,0,C,,SPECIAL>VARSELECT|New variable to use,OUTPUT2
BUTTON,"Perform detailed
        analysis",68,75,25,0,C,,SPECIAL>ALIASSCREEN|ARETURN|OUTPUT2,ANALYSIS
BUTTON,"Advance 1
        Week",80,80,20,5,,,GAME>GAMEINTERVAL|1.0&Game>GAMEON&IFTHENELSE&TEST>Project
        Finish Level\
=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
>Project Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Human Resources",0,85,20,5,L,,"",HumanResource
BUTTON,"Project Status",20,85,20,5,L,,"",Project
BUTTON,"Cost Status",40,85,20,5,L,,"",Cost
BUTTON,"Work Flow",60,85,20,5,L,,"",Workflow
BUTTON,"Decision History",60,90,20,5,L,,"",RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH,
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
!
:SCREEN ANALYSIS
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
```

```
PIXELPOS,0
TEXTONLY,"Analysis Control",50,5,0,0,C|Arial|24|B|0-0-255,
BUTTON,"Select a variable for analysis",50,13,60,5,C,4,"SPECIAL>SETWBITEM|Anticipated Finish
        Date&SPECIAL>V\
ARSELECT|New variable to use",
BUTTON,"Load, unload, or reorder previous runs",50,20,60,5,C,1,MENU>LOAD_RUN,
BUTTON,"Change subscript selection",50,27,60,5,C,8,SPECIAL>SUBSCRIPT|?Choose a subscript to control
        selecti\
on on,
TEXTONLY,"Results",0,36,100,0,C|Times New Roman|24||0-0-255
BUTTON,"List differences between the first two loaded runs",50,43,60,5,C,8,,DIFF
BUTTON,"Display a predefined graph or report",50,50,60,5,C,,,RESULT
BUTTON,"Trace underlying causes using Trees",50,57,60,5,C,5,,CAUSE1
BUTTON,"Trace underlying causes using Graphs",50,64,60,5,C,6,,CAUSE2
BUTTON,"Trace the Uses of a variable",50,71,60,5,C,7,,USE
BUTTON,"Advance 1
        Week",80,80,20,5,,,GAME>GAMEINTERVAL|1.0&Game>GAMEON&IFTHENELSE&TEST>Project
        Finish Level\
=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON&SPECIAL>REFRESH&IFT
        HENELSE&TEST\
>Project Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Human Resources",0,85,20,5,L,,"",HumanResource
BUTTON,"Project Status",20,85,20,5,L,,"",Project
BUTTON,"Cost Status",40,85,20,5,L,,"",Cost
BUTTON,"Work Flow",60,85,20,5,L,,"",Workflow
BUTTON,"Decision History",60,90,20,5,L,,"",RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Restart",61,0,0,0,,,,Intro
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH&IFTHENELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
!
:SCREEN DIFF
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Constant and table differences between first two loaded scenarios",0,2,100,0,C32
TOOL,"D1",5,10,90,80,,,WORKBENCH>RUNS COMPARE
BUTTON,"Print",30,92,20,6,C,Pp,PRINT>D1
BUTTON,"Analysis Control",70,92,20,6,C,,,ANALYSIS
ANYKEY,"",0,0,0,0,0,,,ANALYSIS
!
:SCREEN RESULT
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
PIXELPOS,0
```

```
TOOL,"GR1",2,5,96,80,,,CUSTOM>?Graph to display
BUTTON,"Select another graph or report",2,90,34,6,L,,CUSTOM>?Other graph|GR1
BUTTON,"Print",38,90,10,6,L,,PRINT>GR1
BUTTON,"Copy",50,90,10,6,L,,EXPORT>GR1
BUTTON,"? Help ?",62,90,12,6,L,,SPECIAL>WINHELP|VRHELP.HLP|1600
BUTTON,"Analysis Control",76,90,22,6,L,,,ANALYSIS
!
!
!
:SCREEN CAUSE1
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Causal tracing -",0,2,50,0,R
WBVAR,"",50,2,0,0,L
TOOL,"TR1",2,6,96,42,,,WORKBENCH>CAUSES TREE
BUTTON,"Graph based",2,50,40,0,L,Cc,,CAUSE2
BUTTON,"Definition...",43,50,0,0,L,Cc,WORKBENCH>DOCUMENT,
BUTTON,"Uses - of current variable",2,57,40,0,L,Cc,,USE
BUTTON,"Select a new variable to trace",2,64,40,0,L,Ss,SPECIAL>VARSELECT|New variable for tracing
BUTTON,"? Help ?",2,71,40,0,L,,SPECIAL>WINHELP|VRHELP.HLP|1700
BUTTON,"Analysis Control",2,78,40,0,L,EeXx□ ,,ANALYSIS
TOOL,"GR1",60,50,40,50,,,WORKBENCH>STRIP GRAPH
SETWB,"",0,0,0,0,,,,CAUSE1
!
:SCREEN CAUSE2
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Causal tracing -",2,2,0,0,L
WBVAR,"",20,2,0,0,L
TOOL,"TR1",60,0,40,100,,,WORKBENCH>CAUSES STRIP
BUTTON,"Tree based",2,50,40,0,L,Cc,,CAUSE1
BUTTON,"Uses - of current variable",2,57,40,0,L,Cc,,USE
BUTTON,"Select a new variable to
          trace",2,64,40,0,L,Ss,SPECIAL>SETWBITEM|POPULATION&SPECIAL>VARSELECT|New v\
ariable for tracing
BUTTON,"? Help ?",2,71,40,0,L,,SPECIAL>WINHELP|VRHELP.HLP|1800
BUTTON,"Analysis Control",2,78,40,0,L,EeXx□ ,,ANALYSIS
TOOL,"GR1",2,8,58,40,,,WORKBENCH>DOCUMENT
SETWB,"",0,0,0,0,,,,CAUSE2
!
!
:SCREEN USE
SCREENFONT,Times New Roman|12|B|0-0-0|192-255-192
PIXELPOS,0
TEXTONLY,"Uses of -",0,2,50,0,R
WBVAR,"",50,2,0,0,L
TOOL,"TR1",2,6,96,42,,,WORKBENCH>USES TREE
BUTTON,"Causes - of Current variable",2,50,40,0,L,Cc,,CAUSE1
BUTTON,"Definition...",43,50,0,0,L,Cc,WORKBENCH>DOCUMENT,
BUTTON,"Causes - Graph Based",2,57,40,0,L,Cc,,CAUSE2
BUTTON,"Select a new variable to
          trace",2,64,40,0,L,Ss,SPECIAL>SETWBITEM|POPULATION&SPECIAL>VARSELECT|New v\
ariable for tracing
BUTTON,"? Help ?",2,71,40,0,L,,SPECIAL>WINHELP|VRHELP.HLP|1900
BUTTON,"Analysis Control",2,78,40,0,L,EeXx□ ,,ANALYSIS
TOOL,"GR1",60,50,40,50,,,WORKBENCH>STRIP GRAPH
```

```
SETWB,"",0,0,0,0,,,,USE
!
:SCREEN COMPLETE
SCREENFONT,Times New Roman|10||0-0-0|192-255-192
PIXELPOS,0
!!
TEXTONLY,"Main Screen",67,-5,0,0,L|Arial|14|B|0-0-0,,,
!
!
!
!
!
!
!
!
!
!
!
LINE,"",80,0,0,75,C||||
!
BUTTON,"Help",80,0,10,5,L,Hh,,MainScreenHelp
BUTTON,"Exit",90,0,10,5,L,Xx,SPECIAL>ASKYESNO|Do you really want to exit?&MENU>EXIT,
BUTTON,"Restart",61,0,0,0,,,,Intro
!
!
!
!
!
!
!
!
!
!
!
BUTTON,"Advance 1
        Week",80,80,20,5,,,GAME>GAMEINTERVAL|1.0&Game>GAMEON&SPECIAL>REFRESH&IFTHEN
        ELSE&TEST>Proj\
ect Finish Level=0&Branch>BR1&Branch>BR2",
!
!
BUTTON,"Run to End",69,0,0,0,,,GAME>GAMEINTERVAL|FINAL
        TIME&Game>GAMEON&SPECIAL>REFRESH&IFTHENELSE&TEST>Pro\
ject Finish Level=0&Branch>BR1&Branch>BR2",
!
!LINE,"",72,0,0,35,C||||
!
!TEXTONLY, "Step", 78,10,,5,C||14|
!TEXTONLY, "Forward", 78,14,,5,C||14|
!
BUTTON,"Human Resources",0,90,20,5,L,,,HumanResource
!
!
BUTTON,"Project Status",20,90,20,5,L,,,Project
!
!"",1,2,0,0,,,,
BUTTON,"Cost Status",40,90,20,5,L,,,Cost
```

```
!"",7,-8,0,0,,,,
BUTTON,"Work Flow",60,90,20,5,L,,,Workflow
BUTTON,"Decision History>",40,95,20,5,L,,"",RecDecisions
BUTTON,"Main Screen >",60,95,20,5,L,,"",MainScreen
BUTTON,"Advance 2
        Weeks",80,85,20,5,,,"GAME>GAMEINTERVAL|2.0&Game>GAMEON&SPECIAL>REFRESH&IFTH
        ENELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1
        Month",80,90,20,5,,,"GAME>GAMEINTERVAL|4.0&Game>GAMEON&SPECIAL>REFRESH&IFTHE
        NELSE&TEST>Pr\
oject Finish Level=0&Branch>BR1&Branch>BR2",
BUTTON,"Advance 1 Quarter",80,95,20,5,,,"GAME>GAMEINTERVAL|12.0&Game>GAMEON",TESTING
WIPTOOL,"GR1",2,46,33,39,4,,CUSTOM>WIP1,
WIPTOOL,"GR2",45,45,33,39,,,CUSTOM>WIP2,
!
!
!COMMAND,"",0,0,0,0,,,IFTHENELSE&TEST>Project Finish=1&Branch>BR1&Branch>BR2
!BRANCH,"BR1",0,0,0,0,,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now clic\
!k the O!K button to go to the Status Screen
!BRANCH,"BR2",0,0,0,0,,,,Continue
BRANCH,"BR2",0,0,0,0,,,SPECIAL>MESSAGE|3|Project is Finished|Project is finished.  You may now click the
        OK\
 button to go to the Status Screen,COMPLETE
BRANCH,"BR1",0,0,0,0,,,,
BUTTON,"Analysis",52,0,0,0,,,,ANALYSIS
BUTTON,"Output",44,0,0,0,,,,OUTPUT1
TEXTONLY,"GAME COMPLETION SCREEN",22,7,0,0,|Arial|20|B|0-0-0,,,
TEXTONLY,"Completion Date (Weeks)",7,25,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Initial",40,16,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Final",61,16,0,0,|Arial|14|B|0-0-0,,,
TEXTONLY,"Budget ($)",7,40,0,0,|Arial|14|B|0-0-0,,,
SHOWVAR,"Sanction budget",44,40,0,0,R|Arial|12|B|0-0-0,,,
SHOWVAR,"Perceived Total Budget Required",65,40,0,0,R|Arial|12|B|0-0-0,,,
!
SHOWVAR,"Initial Completion Date%8.0f",44,25,0,0,R|Arial|12|B|0-0-0,,,
SHOWVAR,"Time%8.0f",65,25,0,0,R|Arial|12|B|0-0-0,,,
TEXTONLY,"Changed",50,16,0,0,|Arial|14|B|0-0-0,,,
SHOWVAR,"Completion Date%8.0f",55,25,0,0,R|Arial|12|B|0-0-0,,,
!
TEXTONLY,"Week:",80,6,,,L|Arial|14|B|0-0-0,,"",
TEXTONLY,"Hiring Rate (Engineers per Week)",77,12,21,15,L|Arial|12|B|0-0-0,,"",
MODVAR,"Desired Gross Increase in Staff%8.0f",83,25,10,5,H,,"",
MODVAR,"Overtime Hours Per Week",83,49,10,5,H,,"",
TEXTONLY,"Schedule Slip (Completion Date-Weeks)",76,57,24,17,L|Arial|12|B|0-0-0,,"",
MODVAR,"Completion Date",83,71,10,5,H,,"",
TEXTONLY,"Overtime Authorization - (Hours/week)",77,36,21,14,L|Arial|12|B|0-0-0,,"",
SHOWVAR,"Time",90,6,0,0,L|Arial|14|B|0-0-0,,,
BUTTON,"",20,95,20,5,L,,"",
BUTTON,"",0,95,20,5,L,,,
!
```