System Dynamics Modeling for the Exploration of Manpower Project Staffing Decisions in the Context of a Multi-Project Enterprise

by

Gregory M. Herweg

M.S. Computer Engineering
University of Southern California 1992

B.S. Computer Science
Loyola Marymount University 1987

Karl E. Pilon

M.S. Mechanical Engineering
Rensselaer Polytechnic Institute 1992

B.S. Mechanical Engineering
Worcester Polytechnic Institute 1984

Submitted to 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 2001

Signature of Authors……………………………………………………………………………..
System Design and Management Fellows – January 19, 2001

Certified by………………………………………………………………………………………
Thesis Supervisor: Eric Rebentisch, Ph.D.
Research Associate Lean Aerospace Initiative
Center for Technology, Policy, and Industrial Development

Certified by……………………………………………………………………………….…
Thesis Supervisor: Nelson P. Repenning
Robert N. Noyce Career Development Assistant Professor
Sloan School of Management

Accepted by………………………………………………………………………………………
LFM/SDM Co-Director: Stephen C. Graves
Abraham J. Siegel Professor of Management & Engineering Systems
Sloan School of Management

Accepted by………………………………………………………………………………………
LFM/SDM Co-Director: Paul A. Lagace
Professor of Aeronautics & Astronautics and Engineering Systems
School of Engineering

(This page intentionally left blank for duplex printing.)

System Dynamics Modeling for the Exploration of Manpower Project Staffing Decisions in the Context of a Multi-Project Enterprise

by

Gregory M. Herweg and Karl E. Pilon

Submitted to System Design and Management Program
on January 19, 2001 in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Engineering and Management

ABSTRACT

At the Sikorsky Aircraft and Xerox Corporations, project decisions may be made on a project to project basis and often neglect to account for the complex interactions that exist between projects. Often the current decision process results in a less than optimal return to the corporation. This return may be measured in terms of organizational capability or intellectual capital. Exploration of the effects of decisions with regard to the allocation of manpower in a multiple engineering project setting is required. Also, an understanding is required as to how these daily decisions effect the development of organizational capability or intellectual capital.

The authors propose to describe and model the product development processes currently in use at the Sikorsky Aircraft and Xerox Corporations. The system dynamics method has been employed extensively in the development and application of single project models. However, the application of the system dynamics method in the understanding of multi-project systems is limited. The authors developed an original multi-project model, utilizing the Vensim toolkit, which permits the exploration of manpower resource allocation decisions based on experience of current practice at the Sikorsky Aircraft and Xerox Corporations. This model was exercised to determine the effects of proactive and reactive resource allocation decisions on an organization's ability to complete projects and expand intellectual capital. This learning environment will further the understanding of management at both organizations.

Methodologies learned from the various aspects of the System Design and Management curriculum served the purpose of problem framing and provided validation that a multi-project system dynamics model would serve as a valuable decision support tool. This holistic perspective provides a basis for process improvement and the development of recommendations applicable to the Sikorsky Aircraft and Xerox Corporations.

Recommendations for policy change are categorized with respect to anticipated payback in the near-term, intermediate and long-term time horizons. Near-term recommendations relate to overtime and project prioritization. Policies regarding resource allocation at the beginning and end of projects, as well as a project cancellation policy comprise the intermediate term recommendations. Long-term recommendations include policy that emphasizes scheduling projects with a gap between and a policy that seeks portfolio balance based on intellectual capital growth as well as monetary return on investment.

Thesis Supervisor: Eric Rebentisch
Title: Research Associate Lean Aerospace Initiative

Thesis Supervisor: Nelson P. Repenning
Title: Robert N. Noyce Career Development Assistant Professor

Author Biographies

Gregory Herweg has been employed in the document imaging industry for over 17 years and has been involved with multiple facets of computer software and hardware development projects. He is currently Strategic Planning Manager with the Xerox Corporation in El Segundo, California where he is responsible for the planning associated with large scale printing systems.

Greg holds a Bachelor of Science in Computer Science from Loyola Marymount University and a Master of Science in Computer Engineering from the University of Southern California. He is a member of the National Engineering Honor Society, Tau Beta Pi.

Greg and his family reside in Redondo Beach, California.

Karl Pilon has been employed in the high technology computer consulting and aerospace industries for over 16 years. His most recent experience is with Sikorsky Aircraft Corporation, a United Technologies Company. He currently serves as Heavy Lift Product Line Team Leader responsible for business plan execution.

Karl received a Bachelor of Science in Mechanical Engineering from Worcester Polytechnic Institute and a Master of Science in Mechanical Engineering from Rensselaer Polytechnic Institute. He is a member of the National Engineering Honor Society, Tau Beta Pi and the National Mechanical Engineering Honor Society, Pi Tau Sigma.

Karl resides in Derby, Connecticut with his family.

TABLE OF CONTENTS

LIST OF FIGURES

(This page intentionally left blank for duplex printing.)

LIST OF TABLES

(This page intentionally left blank for duplex printing.)

Chapter 1: Introduction

Managers at the Sikorsky Aircraft and Xerox Corporations must make decisions daily with respect to resource allocation across a range of projects within an overall portfolio. Project decisions made on a project to project basis do not always account for how single decisions can help one project while simultaneously hurting several more projects. This single project focus can lead to less than optimal returns to the corporation. This return may be measured in terms of organizational capability or intellectual capital. An understanding is required as to how firefighting, the fractional rate of movement of people, between projects effects the ability of the corporation to deliver projects on time and the development of corporate intellectual capital. A brief description of these two measures of success is as follows:

Projects Delivered on Time

Projects must be delivered on time in order to exceed customer expectations. Also, the ability of an organization to deliver projects on time is an overall indication of the effectiveness of company processes. Completed projects also contribute to organizational learning through the success stories that they leave as a legacy.

Corporate Intellectual Capital

Growth in organizational capability is necessary in order for firms involved with advanced technologies to be competitive in the future. Organizational structure and product development process should enable learning that permits the achievement of strategic goals.

Competitive pressures in both the document imaging and the helicopter industries require that customer expectations be exceeded every time. Employee expectations must also be met through continued development of skills. However, project extensions as a result of missed delivery dates as well as employee attrition due to a lack of challenging work are increasingly more commonplace. This trend exhibited at both Sikorsky Aircraft and Xerox is opposite the goal required in order to remain profitable, viable businesses.

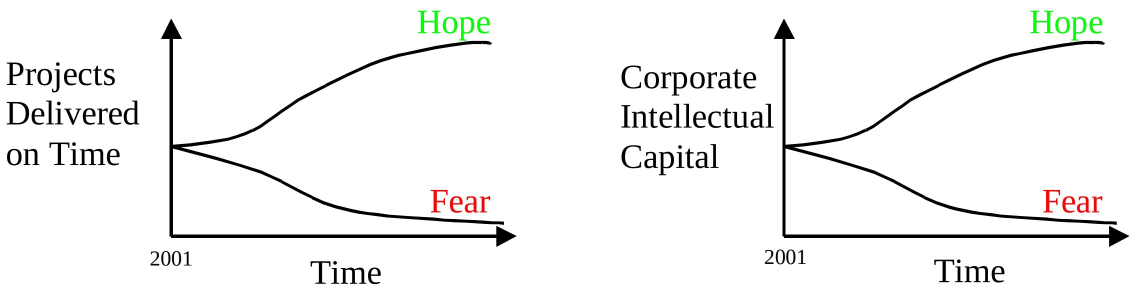


Figure 1 - Reference Modes

Reference modes[1] are developed in order to characterize the problem dynamically. These reference modes will be utilized to describe how the pattern of behavior might evolve in the future. Figure 1 depicts the two possibilities that exist for on-time product delivery and intellectual capital. Let us first consider projects delivered on time. Historically, projects at both companies are increasingly missing the target delivery date. The hope is that from this point forward that improvement will result, the trend may be reduced, and projects delivered on time every time. The fear is that the trend will continue.

The second portion of Figure 1 refers to the development of intellectual capital. The focus here is on the development of human capital as measured by the advancement of individuals from the novice skill ranking to intermediate and then to expert. The hope is that individual skills develop such that the corporation is capable of performing on more technologically advanced projects. The fear is that technical skill, hence organizational capability, will continue to diminish leading to the inability to complete projects.

[1] Reference Appendix C for a primer on the System Dynamics Method.

Chapter 2: Problem Statement

"Shhh, do you hear that!" exclaimed one manager as he was being interviewed. A cheer was heard faintly in the background. "That's the Falcon Team[2]" the manager went on to say. "They are in the process of a major milestone review with the customer. The news must be good and the customer pleased." Later as we walked past the conference area, we could see that indeed the customer and project manager were smiling. However, the major participants from the project team were not. This was not a troubling site, however. It has become more and more commonplace in the work environment. Employees being driven like a square peg into a round hole trying to catch up ground such that the project milestones will be met. We later learned that the Falcon Team review was a success with the customer. However the team attrition rate tells a different story with respect to the employees.

High attrition, employee and customer dissatisfaction are often the result of good projects gone bad. No one intentionally planned for the bad performance. Although the reasons for poor performance may differ, post mortem reviews usually tell the same result. Ineffective and inefficient technology development work caused by a lack of focus. This lack of focus results from insufficient time allocation to tasks. But the workers assigned to perform the tasks often estimate the time to complete. Could they be so wrong each time?

The project manager, who compiles these estimates into the program plan, is often faced with difficult decisions regarding pushing the technical professionals to do a greater number of tasks under time pressure. This compression of schedule and stretching of the workforce is the reality of the competitive document imaging and aerospace marketplaces of today. A study of over two hundred technology development and transition projects at nine companies in the automotive and aerospace industries reveals that over commitment of technical professionals and under-representation of key skills is present in 40 percent of the projects studied. These practices seriously impair team performance. So much so that the weak staffing is found to be associated with a doubling of the failure rate to reach full production. However, when the managers were

[2] Project name disguised.

interviewed, they felt that they provided just the right amount of resources to complete the job.[3] Why does the project manager see things differently?

Perhaps some of the explanation rests with the tools available to the manager for project planning. These tools, such as Pert and Critical Path Method, are static in nature. They provide the best deterministic means for project planning and are widespread in use. The success in the application and use of these tools in addressing singular project plans is well documented. The breakdown in success lies in the use of these tools within this singular project mindset. Each project manager believes that he is doing his best in assuring success for his team through this myopic view. It is this lack of understanding of the dynamics that exist between projects and the lack of perspective regarding total enterprise resource allocation requirements that is the root of the problem.

Tools that permit a systemic and dynamic view of the resource allocation decisions on project outcomes need to become more widespread in use. The benefit will be a paradigm shift for managers and new organizational synergy.

Investigation Scope

The goal of this research is to be qualitative in nature regarding the current mental models that are applied in the planning and management of technology development projects at the Sikorsky Aircraft and Xerox corporations. The leading theories suggest that a systems approach should be used that considers resource allocation decisions in the context of the enterprise, not on a project to project basis. This research seeks to extend the application of these theories with specific application to the product development process that exists within the military project environment at Sikorsky Aircraft and within the commercial setting at Xerox.

A system dynamics model that represents the current product development processes at both corporations will be developed. This model will serve as a tool to enable any manager in these environments to readily visualize the impact on the total project portfolio as a result of their

[3] Lucas, William A., et al. "The Wrong Kind of Lean: Over-Commitment and Under-represented Skills on Technology Teams", The LeanTEC Project, Sloan School of Management Massachusetts Institute of Technology, May 2000., pp. 1 - 23.

singular project resource allocation decisions. The impact will be measured with respect to completion of projects and development of human intellectual capital.

The overall objective of this research will be to develop policy recommendations that encourage well managed resource allocation processes. Also, the implications of these policy recommendations with respect to the effective development of knowledge will be presented. The work that follows will demonstrate the potential for improvement in the mental models regarding project management. The combination of commercial and military project experience in the model development leads to more universal application of the policy recommendations.

Key Questions

The assertion of the authors is that a systemic approach to resource allocation is required for improved project performance and development of human capital. Structural enablers that involve organizational form or processes are not sufficient to ensure project success. These structural enablers when coupled with an understanding of project interdependencies that result from the application of scarce resources are what is required for success, success in terms of project completion and advancement of individual skills. Key questions being addressed by this research are;

- What factors are important to consider in the project portfolio when making singular project decisions?

- What factors affect the development of human intellectual capital?

- How sensitive is the product development process to project complexity, employee experience level and employee fatigue?

Deliverables

There are two primary products of this research.

- A system dynamics computer model. This computer model developed with the Vensim software envelops both the structure and processes that represent technology product development at the Sikorsky Aircraft and Xerox corporations. This model will facilitate understanding of resource allocation decisions and the associated interdependencies that result in a multi-project setting. The model is stylized to represent four projects with varying duration and technical complexity. A user-friendly front end in spreadsheet format permits the manipulation of project characteristics for scenario analysis.

- Recommendations for policy change at Sikorsky Aircraft and Xerox. Also, qualitative implications with regard to the policy impact on intellectual capital.

Summary

An efficient product development process that leads to successful project completion is desirable given the competitive nature of the document imaging and helicopter marketplaces. Projects completed successfully contribute to individual skill advancement and corporate intellectual capital growth through the legacy data, both tacit and explicit, retained in the organization. The contribution of effective resource allocation to this efficiency needs to be explored. The interdependencies that exist between projects as a result of allocation of scarce resources are difficult to understand without a project portfolio perspective.

The development of a system dynamic model of the product development process is an attempt to provide that project portfolio perspective. Application of the model will demonstrate how the current system really works in a multi-project environment. This model aims to provide the decision-maker with a tool such that existing policies may be evaluated.

Chapter 3: Background

Overview

In this chapter, the human, structural and customer capital components of intellectual capital are defined. Increased strategic importance is being placed on a skilled and motivated workforce at a time when the traditional sources of competitive advantage are easily imitated. The evolution of the role of human capital in strategy is creating a need to measure the value to the organization. A value scheme that represents the relationship between the types of capital is presented. This value scheme brings our attention to the renewal and development process as a key to future business success. This value measurement will be translated into practice within the systems dynamic model.

Workflow is discussed next. Maintaining the information flow throughout the product development process by utilizing standardized work practices that result from structured decision making methodologies ensures value and is an artifact that reflects lean practice.

Background is presented regarding the structure of the product development organization that attracts, develops, and retains human capital at both Sikorsky and Xerox. This discussion provides a basis for understanding the efforts associated with creating the structural capital necessary to extract value from the knowledge workers within the enterprise. The discussion follows a past and present presentation format. This format was chosen such that the reader may gain an understanding of the organizational dynamics that result from a shift in structure. An understanding of the organizational structures is important in that the system dynamics model is developed with these structures and processes as a basis. The discussion regarding the change in structure at Sikorsky Aircraft is presented first followed by that of Xerox. The Sikorsky structural change is relatively recent and is discussed in more detail in an attempt to capture the intent of the organizational architects to evade the possibility of eroding functional expertise and encourage cross discipline communication.

The renewal and development process may be placed in jeopardy as a result of management practices that are reactionary. These practices favor resource allocation decisions that encourage investment in current projects at the expense of future projects. The static nature of current project management techniques that treat development projects as independent entities may very well be the primary impetus in the formulation of the managers' mental models. A system dynamics approach is discussed as a paradigm shift mechanism that can serve as a daily decision support tool in a multi-project development environment.

Intellectual Capital

In this section intellectual capital is introduced. The relationship between the different types of intellectual capital; human, structural and customer is discussed. The emphasis is placed on human capital as it is the source of innovation and renewal for the corporation. Inter-project dynamics that effect the ability of the individual to learn will be explored through the use of a system dynamics model. This learning impact will be measured as growth in human intellectual capital, the advancement of individuals from novice to expert skill levels, and serve as an indicator of the organization's ability to compete for future projects.

The market value of a company is often measured in terms of its financial capital reflected in tangible assets. However, many companies are coming to realize that the real value rests in what is considered hidden, those assets that are not traditionally measured in the corporate bookkeeping scheme. These hidden assets are predominantly characterized as the knowledge of employees and customer relationships that lead to brand loyalty. It is individuals, not the company, who own and control the chief source of competitive advantage - the knowledge of organizational members. The greatest challenge for the manager is to keep the information flowing through value creating processes and organizational infrastructure in order to leverage employee knowledge. At this point one may expand the previous definition of intellectual capital to include what is in the heads of the employees as well as what is left in the company when they leave.[4]

The subtleties regarding the interplay between the sources of intellectual capital need to be understood when considering the development of high performance work systems. In order to realize the opportunity that a shift to knowledge intensive services provides; individuals must supply skills to meet the customers' needs, skills and experience must be capitalized and leveraged through organizational structure and processes, and the strength of the franchise or brand must be exploited.[5]

---

[4] Roos, Goran, and Roos, Johan, "Measuring your Company's Intellectual Performance", Long Range Planning, Vol. 30, No. 3, 1997., pp. 413 - 415.

[5] Stewart, Thomas A., "Your Company's Most Valuable Asset: Intellectual Capital", Fortune, October 3, 1994., pp. 71 - 73.

Human capital is considered to be the skills and experience that serve as the source of innovation from within the firm. Growth in human capital is created through hiring, training and applied experience. Structural capital is the tools that capture the knowledge. These range from information systems, customer databases, tools, internal processes and management focus. In short, all things that can be used again and again to create value. Customer capital is the relationship with the customer and the perception of the brand in the marketplace. The relationships between the three types of intellectual capital are best represented pictorially through the use of the Skandia Value Scheme shown in Figure 2.

Skandia, the largest financial services group in Scandinavia, has performed much of the pioneering work with regard to classification and measurement of intellectual capital. This Value Scheme depicts the balance between the financial and non-financial issues. The past is represented

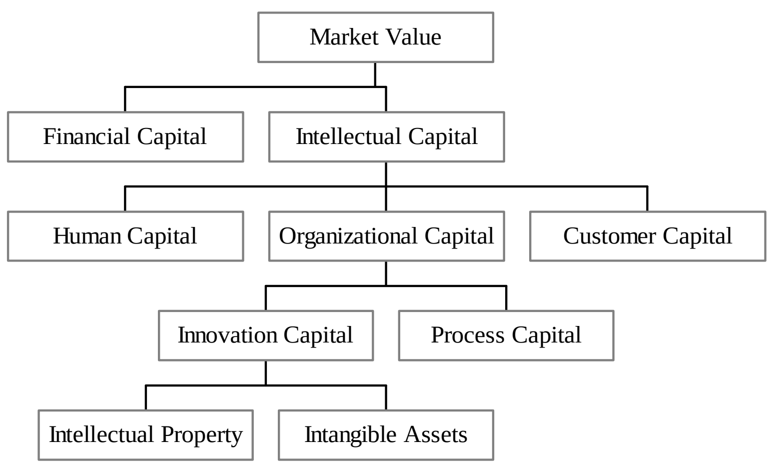


Figure 2 - Skandia Value Scheme

through the financial measures. The present is characterized through a focus on the customer, human resources and processes. The future is depicted as a focus on renewal and development. This renewal and development focus involves innovation capital and may be considered as the foundation for the long-term sustainability of the enterprise.[6] An additional study performed by the Ernst and Young Center for Business Innovation and the Wharton Research Program on

---

[6] Edvinsson, Leif, "Developing Intellectual Capital at Skandia", Long Range Planning, Vol. 30, June 1997., pp. 369 - 371.

Value Creation in Organizations confirms that corporate value is driven through innovation and the ability of the firm to attract talented employees.[7]

The work of Skandia and others teaches us that knowledge assets can be identified and that intellectual performance can be measured. Measurement methods assign indicators to categories of intellectual capital and are in static, a balance sheet approach, as well as dynamic forms. The dynamic form measures intellectual capital growth or decline over time. Regardless of the method, it is through this measurement that subsequent financial performance may be predicted.

Intellectual capital is an important factor in this research in that it serves as an indicator of the organization's ability to address potential future work. Organizational design and processes that impair innovation should be identified and corrected. However, intellectual capital needs to be measured first. This measurement will be made specifically through the application of a systems dynamic model that will capture the growth or decline in human intellectual capital. This growth or decline measurement will be reflective of human resource allocation decisions and allow for improvement in mental models regarding the dynamics associated with a multi-project environment.

Intellectual capital creates value through activity and process, which includes the structure of the engineering process. The section that follows will proceed to discuss how those engineering processes function.

---

[7] Baum, Geoff, et al. "Introducing the New Value Creation Index ", Forbes ASAP, April 3, 2000. pp. 140 - 143.

Workflow in the Engineering Enterprise

This discussion stimulates model development as it is based around the flow of work in the enterprise and is of primary importance in the completion of projects. Model structure will be created to allow for the simulation of work completion, both correctly and incorrectly, and its movement through the four phases of a representative product development process.

The multi-disciplinary, cross-functional product development teams that exist at both Sikorsky and Xerox were architected with the intention of reduced coordination and improved information flow. This organizational design reflects current practice in the area of lean thinking. In the quest for reducing waste the value stream is analyzed to minimize handoffs and increase knowledge retention. This leads teams to standardize work that results in continuous improvement of design methodology. The design moves forward and rework is eliminated through the application of decision-making methodologies that ensure value.[8] The Quality Function Deployment method is an example that is employed at both companies to insure information flow in the product development process. Figure 3 represents value and product development information flow at both the Sikorsky and Xerox organizations and is an adaptation of the basic framework established by Slack.[9] This information flow framework serves as a visual tool that depicts the creation of value at Sikorsky Aircraft and Xerox through the transformation of customer requirements into product form.

Product development information flow is represented through a series of processes. These processes however, also translate into functional disciplines. At Xerox virtual product teams are created to address specific project requirements. The teams consist of functional core specialists in the areas of requirements, development and test. The development function comprises two disciplines, high-level design and implementation. The product platform teams at Sikorsky as discussed earlier comprise co-located functional specialists. Requirements specialists address the

---

[8] Womack, James P., and Jones, Daniel T., Lean Thinking: Banish Waste and Create Wealth in Your Corporation. New York: Simon & Shuster, 1996., p. 54.

[9] Slack, R., "The Application of Lean Principles to the Military Aerospace Product Development Process", Unpublished Master's Thesis. Cambridge, MA.: Massachusetts Institute of Technology, December 1998., p. 31.

requirements definition with overlap in participation from high level design and test functions. Specialists from the advanced design and development function carry out high level design. Detail design is accomplished by the myriad of team specialists resident. The project type determines the required mix from airframe, dynamic systems, electrical, avionics, system integration and so forth. This information flow process mapped with the functional makeup of the virtual and co-located teams will serve as the foundation that describes engineering value adding activity in the simulation model that follows.

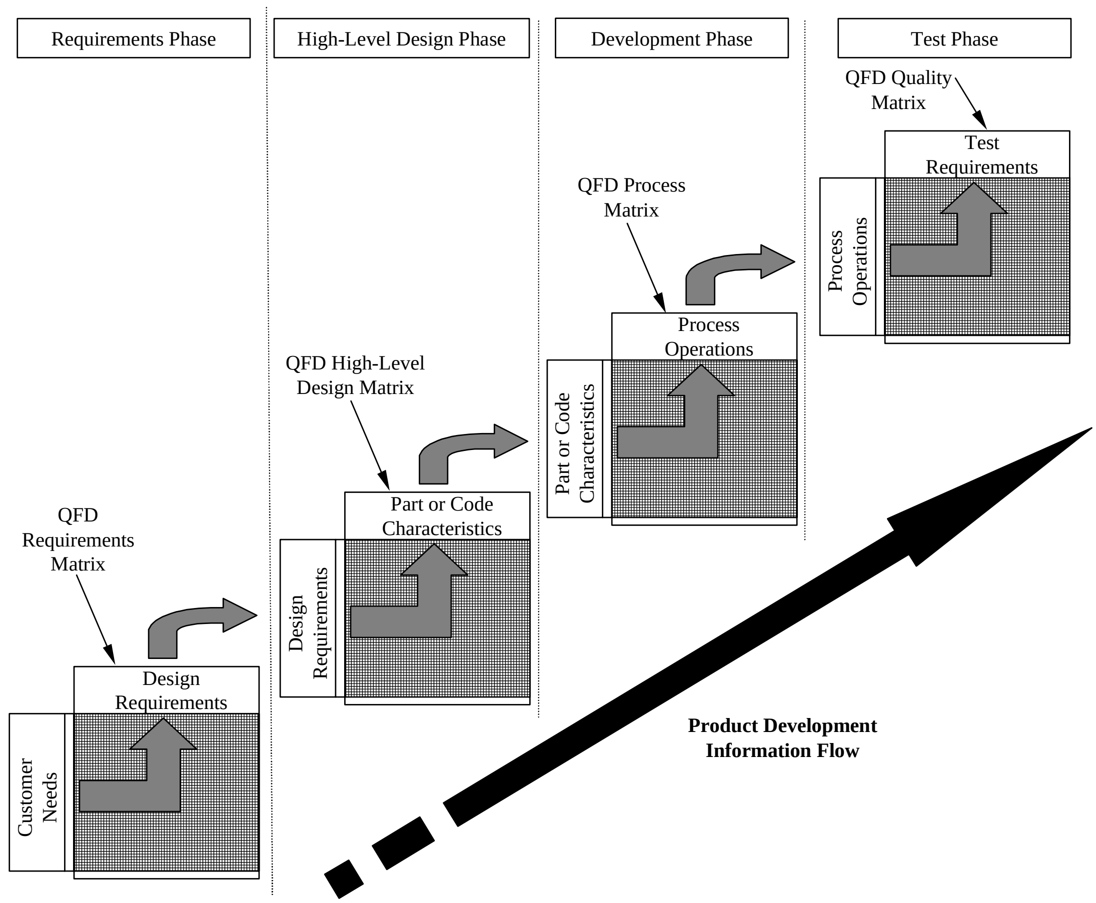


Figure 3 - Quality Function Deployment Information Flow Framework

Background: Sikorsky Aircraft

Sikorsky Aircraft, a division of United Technologies Corporation, is a world leader in the technology of vertical flight. This status has been challenged in recent years due to a decline in United States Military sales and increased competitive pressures in the International Military and Commercial markets.

In response to this challenge, Sikorsky Aircraft has committed to a change intended to realign division resources in order to maximize value to the customer and improve competitive advantage. This change, consisting of a shift from a functional organization to a platform team organization, was initiated within the engineering department in February 1998.

Platform Team Change Initiative

The platform team concept is not new to Sikorsky Aircraft. During the 1960's this was the organizational structure of the company. However, in the early 1970's, a concern for the standardization of technology among product lines as well as the requirement to increase efficiency with groups of technical specialties led to a change to the functionally based organization that existed until February 1998.

The functional organization was effective in addressing customer needs and requirements characteristic of large United States Military aircraft production orders. However, the cost plus type of contracts inherent to this environment offered little incentive or necessity for efficient operation. This became apparent as the market shifted to smaller U.S. military and international orders. As Sikorsky increased efforts to compete in the global market place where fixed price contracts, small quantities, varied configurations and short lead times are the norm, the need for decisive change became apparent.

Recognizing this need, Donald Gover, the Vice President of Production Engineering at the time, had advocated a dramatic change that was challenging the employee's view of how Sikorsky Aircraft will compete in the future. On three separate occasions, the entire engineering staff of over one thousand individuals, was assembled for a series of presentations where he communicated his vision of the new engineering organization. Coincident with the change initiative a new slogan, from Ken Blanchard's book "Raving Fans", was adopted. This slogan, which was intended to represent our new corporate vision statement, was expressed as follows;[10]

Sikorsky Vision

"Make every customer a raving fan"

- By understanding our Customers' needs as well as they do to deliver products and services that exceed expectations
- By implementing a team-based design, development and production process that achieves decisive market speed, cost, and quality advantage
- By attracting and retaining the best people
- By using technology to maximize value to the customer
- By fostering and embracing change

The three meetings with the entire engineering staff took place over a three-month period. Don Gover explained the five elements of his vision at the first meeting in February 1998. The impending organizational change from a functional base to a team base, as emphasized in tenet two above, was explained at length. The second meeting took place in March 1998 after most of the platform teams had been formed and collocation of individuals from the dispersed engineering groups had begun. Here the Vision was re-emphasized and platform team definition was presented. The third meeting, in April 1998, addressed the co-location concerns and reaffirmed the competitive necessity for the platform team change.

Functional Organization

The engineering functional organization as it had evolved over the twenty-year period prior to implementation of the subject change initiative is depicted in Figure 4. Seven functional branches were required to address all aspects of the production engineering process. Each branch consisted of groupings of similar functional competencies. Within a functional branch, individual functional groups often had common employee skill requirements. Despite this

[10] Gover, D.; "Vice President of Production Engineering, New Engineering Organization Address to Engineering Staff". Sikorsky Aircraft Corporation, February 1998.

commonality of requirements within the functional branch, resources were seldom shared among different disciplines.

To attain a high level of expertise within a particular functional competency, long tenure was normally required. Generally, advancement within the functional group was directly associated with tenure and competence. Because of this incentive system, individuals rarely moved across functional groups thus developing strong group loyalty. Broad and effective informal communication networks were established across functional groups as a result of this constancy of employees within the differing functional disciplines. In general, an atmosphere of cooperation existed between functional groups within a branch, however, communication between branches was often less than satisfactory. The functions that were largest in scope and number of employees were given the primary allocation of resources.

Within this organizational structure, direct communication between the Engineering community and the customer was virtually nonexistent. Customer requirements and objectives were relayed to the functional groups by the Product Line Program Engineering Management (PEM) department. This was the singular engineering link to the customer. Individuals received direction from both their functional supervision as well as the PEM. Conflicting instructions from functional management and the PEM were a common occurrence. Since the functional manager controlled incentives, functional group or branch instructions were often given precedence over customer requirements.

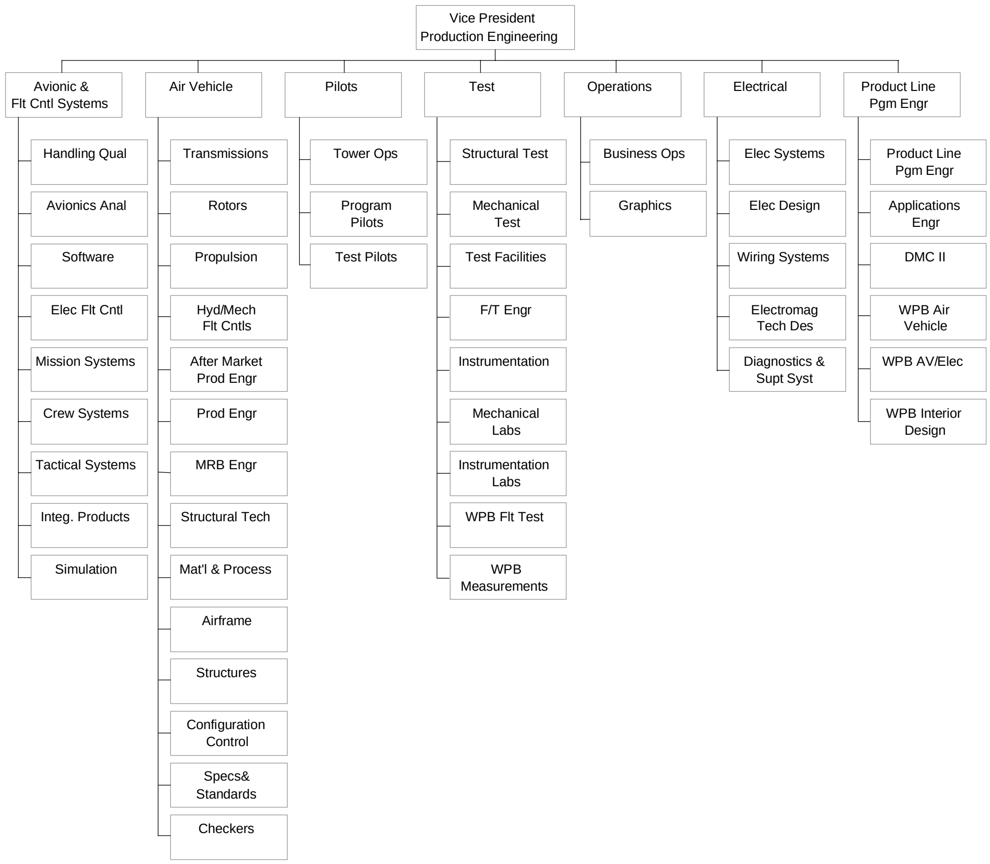


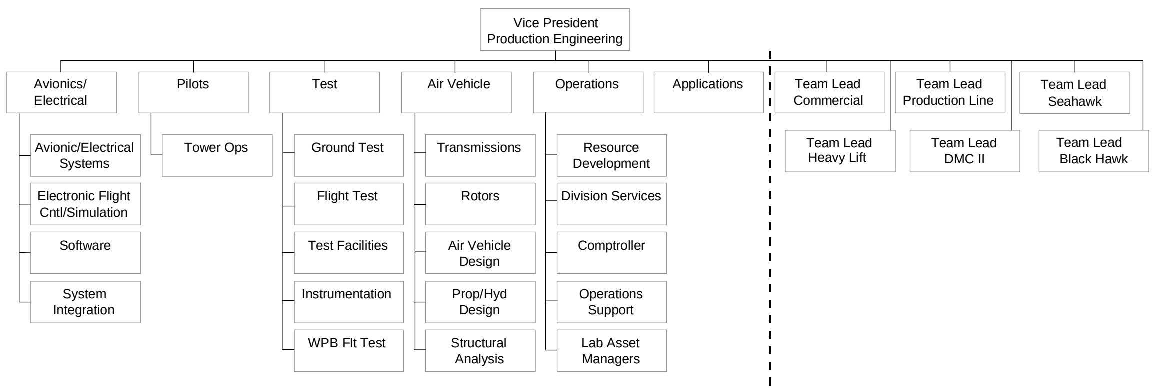
Figure 4 - Functional Product Development Organization[11]

Platform Teams

The product platform team process was developed in order to provide a single point of focus for the customer. Additionally, the collocated platform team should eliminate confusion, by enabling team members to focus their efforts on a specific set of customer requirements and team objectives. The Product Platform Teams represent the full-scale implementation of a prototype platform team that was established within the Development Manufacturing Engineering department. The goal of this prototype effort was to create an autonomous team, comprised of highly skilled individuals from each functional branch that would be responsible for all aspects of the entire aircraft development process from requirement definition to product delivery. This team would also interact directly with the customer throughout the entire project cycle. A process benchmark of industry competitors served as the basis for this prototype platform team.

[11] Ambrose, M., et al; "Organizational Initiative Analysis, Sikorsky Aircraft Reengineering". MIT Sloan School of Management Organizational Processes Course, March 1998.

The new team based organization depicted in Figure 5 comprises functional core competency groups as well as product platform teams. Readily apparent is an approximately fifty-percent reduction in the number of functional groups within the functional branches. The most notable change is the reduction of resource groups within the Air Vehicle branch from fourteen to five. The intent was to eliminate the duplication of skills and consolidate resources that required extensive interface. In certain instances the existing functional core group leader assumed the new responsibility of technical consultant and a replacement group leader was introduced from outside of the group competence. The new role of the functional core competencies will be to provide a skilled manpower base, through core competency development, for deployment to the product platform teams.



Functional Core Competencies **Product Platform Teams**

Figure 5 - Production Engineering Organization 1998 [12]

The organizational basis for the individual platform teams is a specific product line. The intent of this arrangement is to enhance team focus and management control to ensure that customer expectations are met or exceeded. Collocation of the product team resources enables the team leader to efficiently utilize member skills without the limitations and restrictions normally imposed by functional boundaries. Improved cross-functional communication results as functional "stovepipes" are eliminated. Collocation provides opportunities for better communication between individuals of interfacing departments throughout the design process. The "over the fence" handoff effect of the previous functional organization is eliminated thus resulting in a better-integrated product. Collocation also provides a means of cross training team

[12] Ambrose, M., et al; "Organizational Initiative Analysis, Sikorsky Aircraft Reengineering". MIT Sloan School of Management Organizational Processes Course, March 1998.

members in functional core areas that they may not have been exposed to previously. Reallocation and relocation of resources also signifies an important shift in authority from the Functional Group Manager to the Platform Team Leader. Eugene Buckley, Chairman and Chief Executive Officer of Sikorsky Aircraft at that time, characterized the role of the product platform teams, as having the sole responsibility for each aircraft program and delivery of the product to the customer.[13]

The new engineering organization was designed with careful consideration of the interrelationships of the entire enterprise. Figure 6 is a representation of the dynamic interaction of the core competency functional groups and product platform teams within the engineering department, and other areas of the corporation with enterprise leadership providing the required strategic direction.

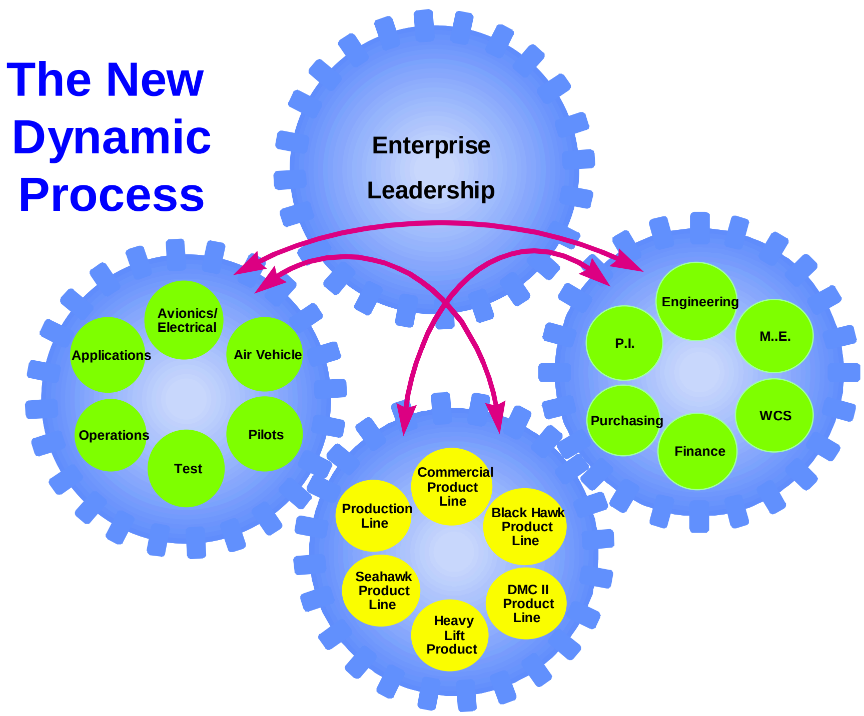


Figure 6 - The New Dynamic Process [14]

---

[13] Buckley, E.; "Corporate Chairman Commentary Regarding Product Platform Team Responsibility". Executive Management Council Meeting, Sikorsky Aircraft Corporation, March 1999.

[14] Ambrose, M., et al; "Organizational Initiative Analysis, Sikorsky Aircraft Reengineering". MIT Sloan School of Management Organizational Processes Course, March 1998.

Background: Xerox

Xerox is considered to be a primary source of innovation in the document imaging, document management and document reproduction marketplace. Xerox has a global presence in manufacturing, distribution and marketing and sales. Product development activities are divided primarily between facilities located in Rochester New York and El Segundo California. This discussion will focus on product development structures in these two locations.

The history of the organization at Xerox differs from that of Sikorsky in that the product development organization has transitioned from teams centered around single Product focus, to that of a Platform team centered around platforms that drive several devices within a family of products. The platform business team more closely resembles the functional organizational past of Sikorsky.

Product Teams

The engineering product development organization that existed from 1970 to mid 1990's is depicted in Figure 7. Groups of functional specialists were co-located and organized around product lines. Each product team was comprised of functional specialists from requirements, high level design, development and implementation, and test. Product organizations could be 250 people or more in size. The product teams functioned as independent autonomous units with specific focus on a particular set of customers.

Because of the diverse nature of the product lines, functional specialists of the same discipline worked independently from their peers. This resulted in organizational duplication as well as produced little transfer of knowledge between products. This organization was effective however, at developing solutions to address specific customer desires. Thus the result was high customer satisfaction as suggestions for product features were often implemented to the fullest extent. The byproduct of this behavior was products that lacked the same look and feel.

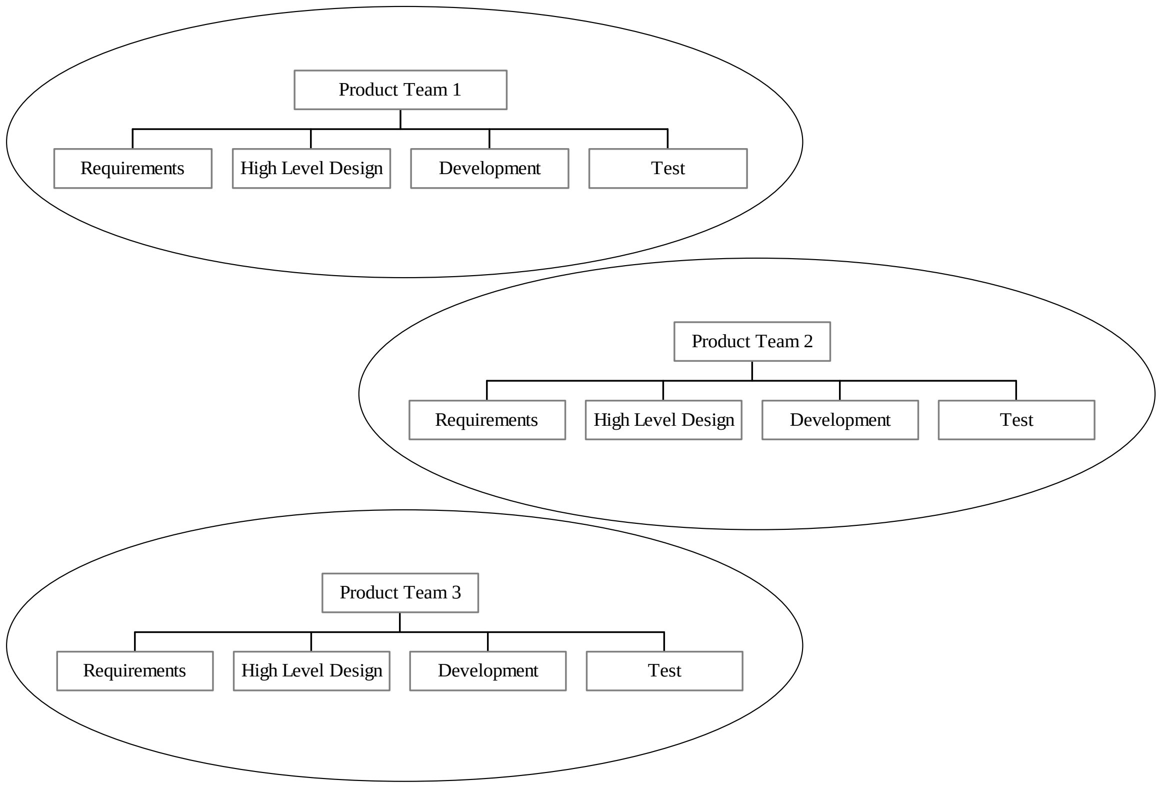


Figure 7 - The Xerox Product Team Organization

Platform Team

In the mid 1990's the product teams began to be combined to form platform teams. This was a result of efforts to reduce the structural inertia within the product development process. An

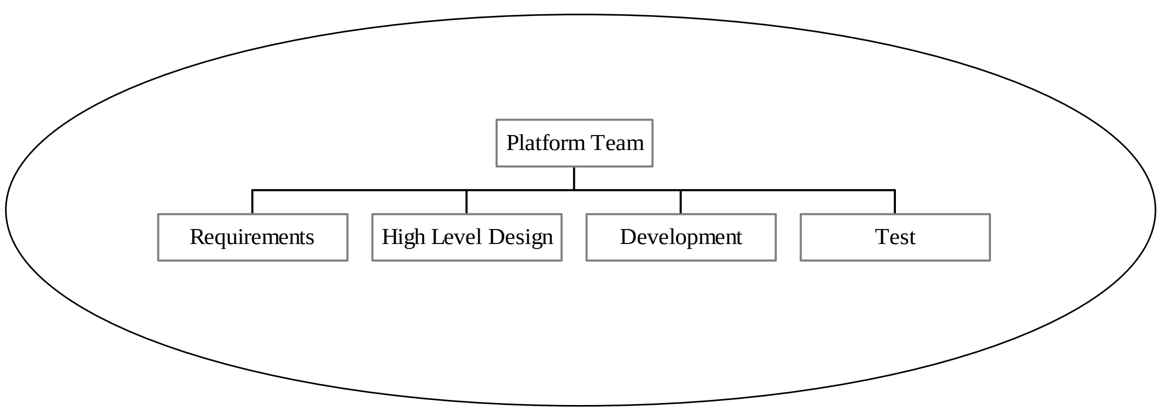


Figure 8 - The Xerox Platform Team

additional impetus was the need for commonality across all product lines that could provide the benefit of software reuse leading to reductions in time to market. The new organization has approximately half of the manpower as the old. The platform team resembles an engineering functional organization with four functional branches required to address all aspects of the production engineering process, reference Figure 8. However, all customer requirements are addressed concurrently within this singular organization. Deep rooted functional competence that insures innovation may be developed yet communication is encouraged across functions. Product families with common features now result. Commonality in employee skill requirements within the functional branches encourages resource sharing among the different disciplines.

Summary

The different approaches to organizational design at Sikorsky Aircraft and Xerox reflect the strategic significance of identifying core competence, focus on the products as a system, and attention to the customer. However, the combination of core functional and product platform organizations at Sikorsky or the platform teams at Xerox are not enough to evade the possibility of eroding functional expertise. Competition for scarce resources creates project interdependencies. It is these interdependencies coupled with the organizational structure that must be understood. Application of the system dynamics method may help to refresh existing mental models with a dynamic perspective.

Dynamic Modeling

Many authors have recognized that product development processes are complex systems with interdependent elements. The traditional tools available to describe development activities in terms of precedence and duration reinforce a deterministic and myopic view of project management. It is documented that managers have the greatest ability to influence product outcomes through early involvement in decisions. Unfortunately, their involvement in programs is inversely proportional to their ability to influence the outcome, reference Figure 9.[15] The end effect is disruptive reactionary practices late in the product development cycle. More effective tools are required that provide for a systemic as well as dynamic approach to project management.

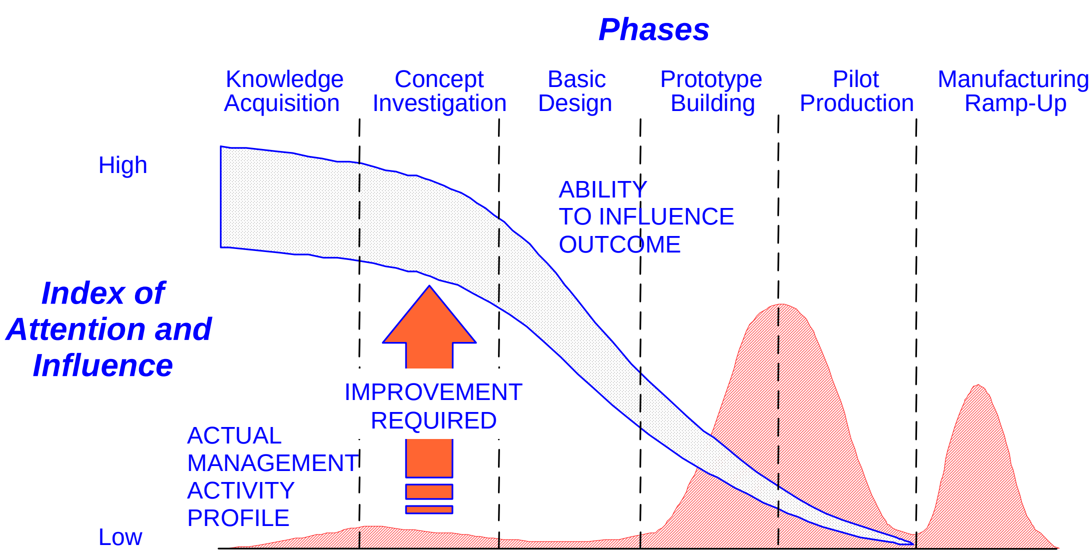


Figure 9 - Management Attention and Influence

The system dynamics method has been applied to the single project environment with great success. Mental models regarding the drivers of project performance have been significantly influenced as a result of this work. Ford and Sterman[16] have explored the multi-phase project

[15] Hayes, R.H., Wheelwright, S.C. and Clark, K.B. Dynamic Manufacturing. New York: The Free Press, 1988., p. 279.

[16] Ford, David N., and Sterman, John D. "Dynamic modeling of product development processes", System Dynamic Review, Vol. 14, No. 1, Spring 1998., pp. 31 - 68.

model in order to provide insight into the dynamics of development processes. Task sequencing constraints that result from within a single phase or from upstream development phases coupled with iteration in the work flow and the effect on resources and delivery dates capture how development processes affect project performance. These intra-phase effects that exist within a project signal the importance of exploration of the effects that are a result of interactions between multiple projects within the company development portfolio.

There exist few formal models that explore project interdependence in the development environment. Allocation of scarce resources is of particular interest as a source of interdependence. It is understood in practice, that in this multi-project environment, policy favors allocation of scarce resources to projects in the later stages of development at the expense of those projects in early phases of development. Repenning[17] explores the self-reinforcing effects of this type of policy. The cascading effect of resource allocation decisions that address the immediacy of existing projects at the expense of future work is demonstrated. The failure of managers in this type of decision-making environment, where the dynamic effects are predominant, reinforces the need for a systemic perspective regarding human resource allocation.

[17] Repenning, Nelson P., "A Dynamic Model of Resource Allocation in Multi-Project Research and Development Systems", Sloan School of Management Massachusetts Institute of Technology, Version 2.0, September 1999., pp. 1 - 49.

Summary

The strategic importance of intellectual capital, specifically human capital, has been presented. In order to determine value, the need for measurement is identified. This measurement within the framework of a system dynamics model will serve as a useful outcome indicator for comparison of project resource allocation decisions in a multi-project environment.

A review of the Sikorsky and Xerox product development organizations and the flow of work was presented to serve as a basis for model development. The flow of work and the structure of the organization contribute to the development of intellectual capital. However, the presence of structure and processes do not guarantee advances in technical capability.

The long-term investment in renewal and development is placed in jeopardy at the expense of short-term gain. The Repenning model, based on simplifying assumptions, will be extended to reflect the product development processes common to Sikorsky Aircraft and Xerox. The expanded model can be exercised to determine the effects of proactive and reactive resource allocation decisions on an organization's ability to complete projects and expand intellectual capital. This learning environment will further the understanding of management at both organizations.

(This page intentionally left blank for duplex printing.)

Chapter 4: Dynamic Hypothesis

Hypothesis Development

The cumulative project management experience of the authors is the primary source of insight into the difficulty surrounding the execution of desired product design processes. This insight is supplemented with qualitative data collection in the form of informal interviews with individuals in the requirements, high level design, development, and test functions. Also, additional insight is provided as a result of process participant observation and archival data in the form of project manpower planning charts.

Time after time large engineering projects are planned for perfect execution despite early warning from engineers and developers that live in the trenches. More often than not, large engineering projects fail to deliver either on time or within the originally allocated budget. In fact, many projects get canceled before completion. During post-mortem examination of failed projects, managers respond with “how could this have happened” while the engineers associated with the project typically reply with “I told you so”. This phenomenon has repeated itself many times at both Sikorsky and Xerox.

Exploration into the cause of project failure suggests that an unfavorable allocation of resources persists. The product development process involves time-phased activities that, in order for successful completion, may require allocation of desired manpower resources early within the development cycle with a reduction as the project approaches completion. The contrary may be true however in that the allocation of resources up front is often less than optimal. This may result in a reactive manpower allocation practice in later phases of the product development cycle. This reactive practice, often referred to as firefighting, may create a cascading effect across all projects within the engineering project portfolio.

Dynamic Hypotheses

The dynamic hypotheses[18] relating to Projects Delivered on Time and Corporate Intellectual Capital are listed below. Each hypothesis is developed from the system-of-projects perspective and further defined in this chapter.

Attrition

Productivity

Project on Time Completion

Resource Progression from Novice to Intermediate to Expert Skill Level

Timing of Hiring

Attrition

The first dynamic hypothesis is that firefighting drives the attrition of good people. Firefighting is described as the fractional rate of movement of people between projects. This causes an overall decrease in corporate intellectual capital.

This attrition is predominantly the result of fatigue due to increased workloads. Due to the long product cycles in the low volume high dollar product document imaging and helicopter industries, change requests are often introduced into the work cycle that increase the total workload. These change requests may be the result of rework to correct design issues, address safety concerns or customization to meet customer expectations. The causal structure is represented in Figure 10. There are three elements to this causal structure; work to do, work errors, and fatigue.

As the work-to-do increases, the workforce required increases. The new-hires into the workforce are often inexperienced and produce more errors in their work as they learn while doing. The increase in errors leads to an increase in rework that requires additional staffing to address. Since the additional staff takes time to hire the rework needs to be addressed through the enforcement of overtime. As overtime increases this balancing loop is closed as more work will be completed which leads to less work-to-do.

[18] Reference Appendix C for a primer on the System Dynamics Method.

A reinforcing loop is introduced as a result of an increase in work errors that produce more rework and overtime. Working overtime creates fatigue. The overworked and tired workers continue to produce more errors closing out the loop.

Continued fatigue encourages the formulation of the mental model that the work environment will not change which leads to attrition and an overall reduction in the workforce. This completes the second balancing loop.

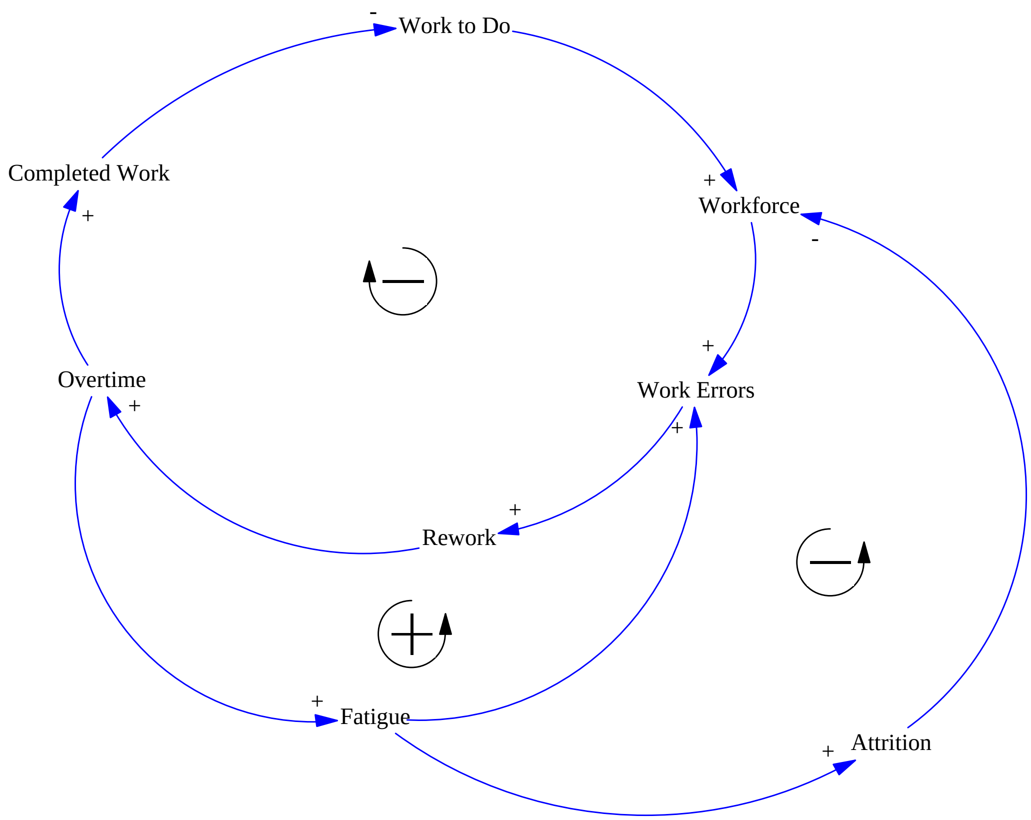


Figure 10 - Attrition Causal Loop

Productivity

The second dynamic hypothesis is that firefighting drives the loss of productivity. The loss of productivity is primarily the result of a loss of focus created by overburdening employees. This leads to a decrease in the corporate intellectual capital because employees are unproductive and not advancing their skills. The productivity causal loop is depicted in Figure 11.

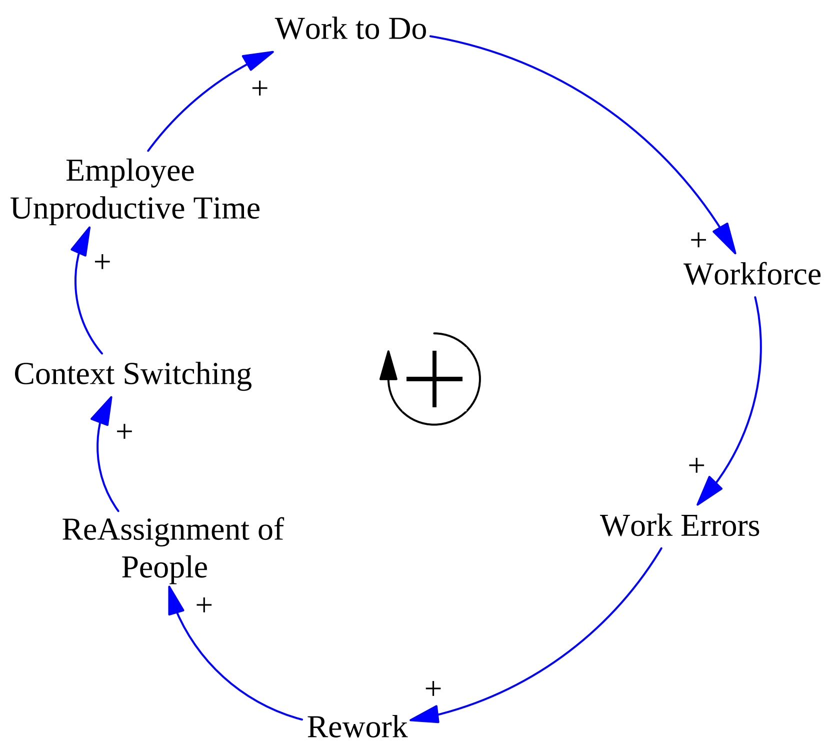


Figure 11 - Productivity Causal Loop

An increase in work-to-do requires additional workforce. The new workforce practices learning while doing and produces an increase in errors. This increase in errors leads to additional work classified as rework. In order to keep up, people are reassigned from other projects to address this rework. This reassignment is one of the central mechanisms for enabling the interaction between multiple projects within a multi-project portfolio. In addition, this reassignment requires that the employees adjust their skill reference frame to get in synch with the new project. Essentially the employee is unproductive as a result of a potential physical change in location or while waiting for enabling information regarding the new assignment. Work continues to go unattended while employees are unproductive. This completes the productivity reinforcing loop.

Project On-Time Completion

The third dynamic hypothesis is related to project on-time completion. Project on-time completion is negatively affected due to schedule slips that result from delays due to errors discovered in completed work. This causal structure comprises five loops, three reinforcing and two balancing.

The first loop is the balancing work-to-do loop that was previously described in the attrition section above. No additional explanation is necessary.

Three different loops are represented by work errors. The first of the three loops is a reinforcing loop that reflects more rework and overtime as a result of error production. The additional overtime increases employee fatigue and increases attrition that reduce intellectual capital. Reduced intellectual capital means less skill available and a greater production of errors.

The second work error loop is also a reinforcing loop. Project schedule slip is the result of the increased amount of errors. As schedule slip increases, project on-time completion decreases. Completed projects serve as a means for learning. Therefore, intellectual capital is decreased as well. A decrease in skills produces a organization that is more error prone.

A balancing loop, the third loop associated with work errors, is created as a result of learning when work is completed. Improved learning increases intellectual capital and subsequently reduces work errors.

The fifth loop is reinforcing and a result of employee attrition due to fatigue. Senior level employees are required to perform more multi-tasking that reduces productive work time and leads to work errors. This increases fatigue as more hours must be worked to accommodate for the shortfall due to time spent on rework. The continual catch-up mode of operation causes attrition that produces more hiring in order to offset. More hiring increases the workforce. Reference Figure 12.

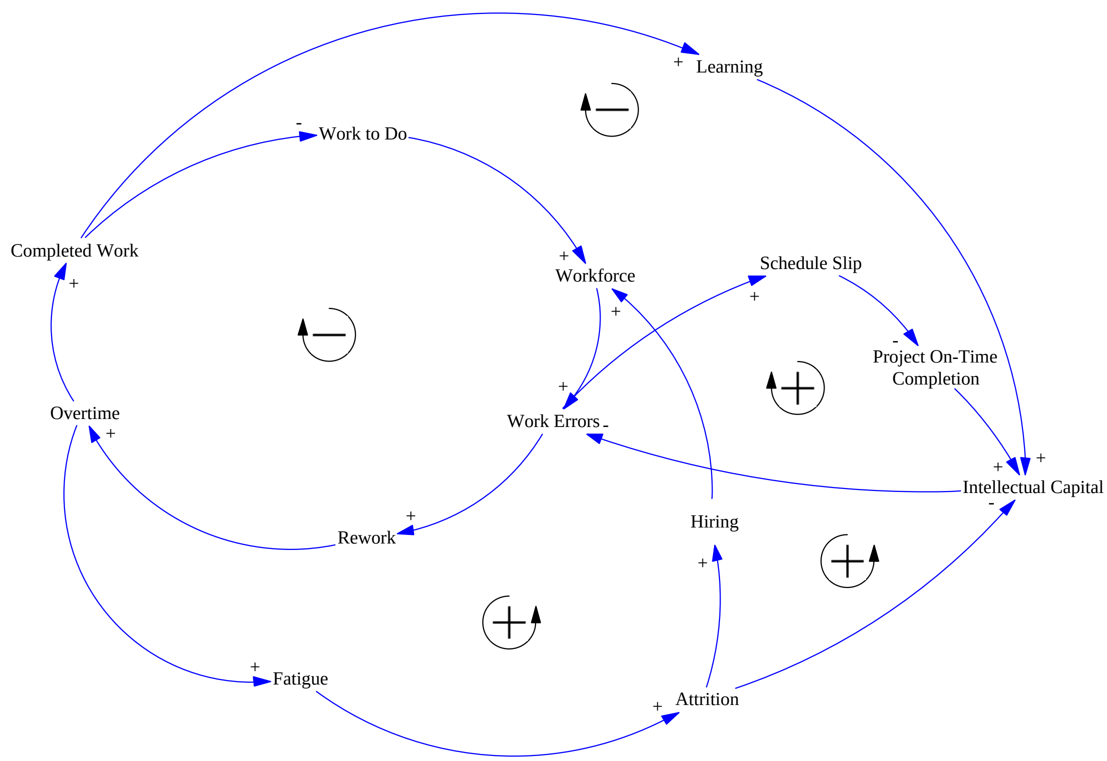


Figure 12 - Project Completion Causal Loop

Resource Progression from Novice to Intermediate to Expert Skill Level

Resource progression is the basis for the fourth dynamic hypothesis. One method for intellectual capital growth is through employee learning. This learning can be a result of informal on-the-job application of existing skill sets while adapting to new work processes. Mentoring serves as a more structured transference of tacit knowledge from senior level employees to the junior level. Cost reduction measures at both Sikorsky Aircraft and Xerox have created an increased reliance on personnel with the highest levels of experience. The attrition losses at this level combined with a lack of time and a lack of focus that results from context switching all lead to reduce efficiency and reductions in intellectual capital and project on-time completion.

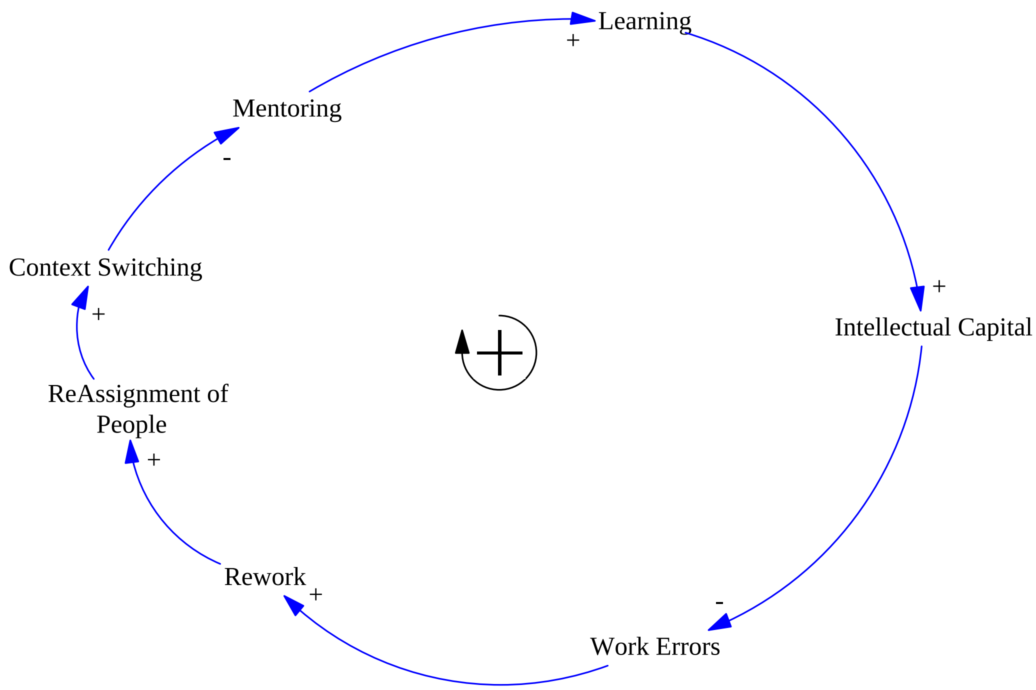


Figure 13 - Resource Progression Causal Loop

Figure 13 is the causal loop that represents changes in intellectual capital as a result of resource progression from novice to intermediate to expert skill levels. This reinforcing loop starts with an increase in work errors which leads to increased levels of rework that require the reassignment of staff from other projects to the project in trouble. This reassignment dilutes employee focus and reduces the time for senior level employees to mentor junior level staff. Reductions in learning and intellectual capital produce a workforce that is even more error prone than before.

Timing of Hiring

The final hypothesis to be addressed focuses on the timing of hiring. The hiring process is often initiated to address manpower shortfalls in a crisis situation. However, the hiring cycle is often lengthy and less than satisfactory in meeting the instantaneous demand required of the crisis. Intellectual capital may actually be reduced rather than increased in this situation. This potential reduction may be the result of the lack of time available to senior level staff to perform mentoring. This balancing loop is depicted in Figure 14.

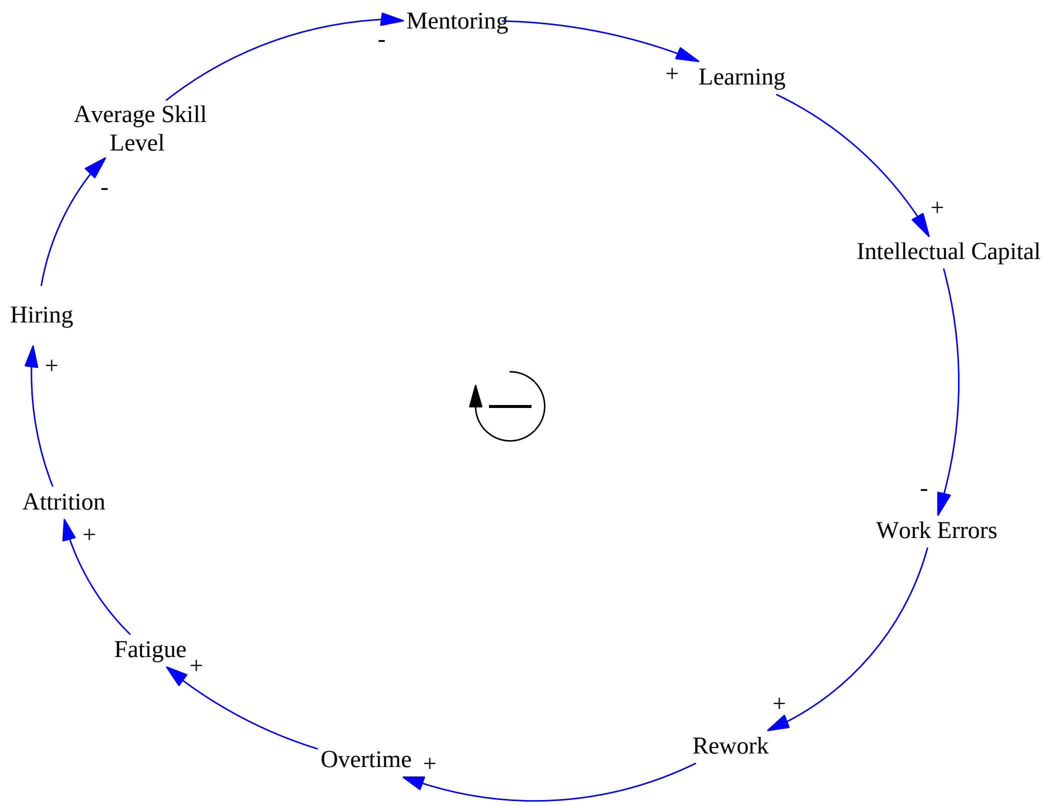


Figure 14 - Timing of Hiring Causal Loop

Hiring reduces the average skill level. This increases the need for mentoring, increase learning and intellectual capital. The more the intellectual capital, the less error prone the employees are. Less errors translates into less rework, fatigue, and attrition. Fewer people leaving has a balancing effect and slows hiring.

Summary

Combining all loops developed in addressing the dynamic hypotheses produces a single causal loop diagram. This is represented in Figure 15.

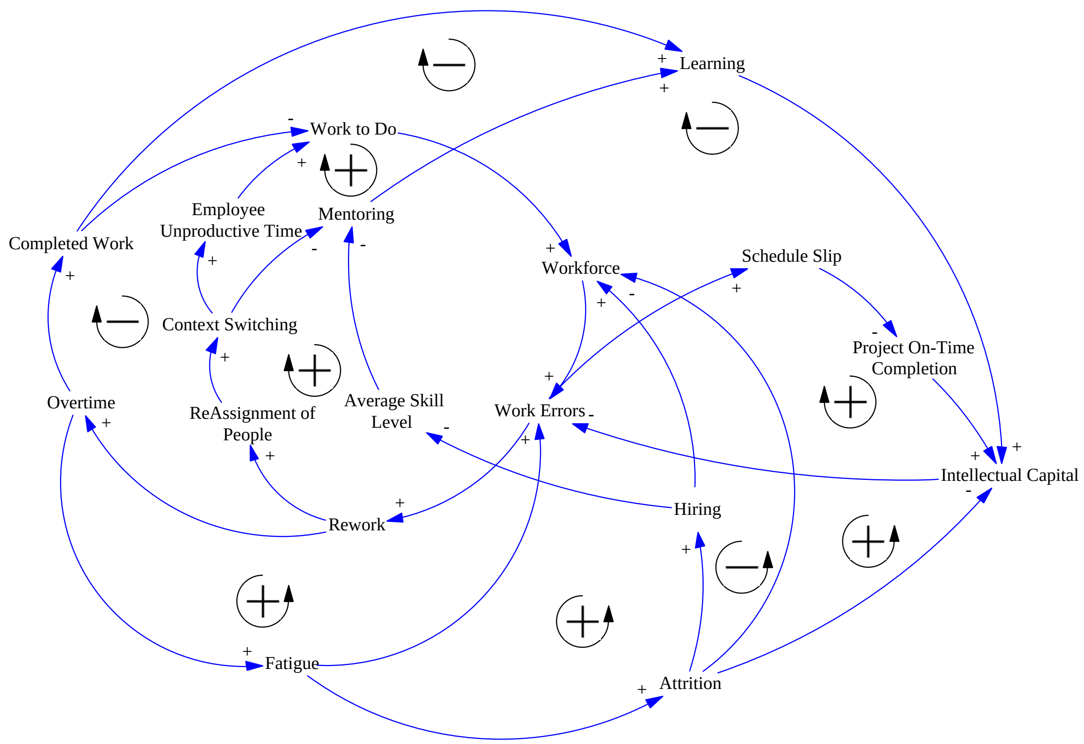


Figure 15 - Combined Causal Loop

The potential fear and hope behaviors described in the reference modes may be produced as a result of the combined loops represented. In order to assess the validity of these dynamic hypotheses a system dynamics model has been formulated with the combined causal loops serving as the basis for model structure. Chapter Five presents the architecture and development of this system dynamics model that will produce insight that leads to recommendations that improve project on-time completion and intellectual capital.

(This page intentionally left blank for duplex printing.)

Chapter 5: System Dynamics Model

Model Structure

A Vensim computer simulation model[19] model was created in order to explore our hypotheses. Its overall architecture consists of the three basic components necessary to examine the dynamic behavior of the workflows, project resources tradeoffs, and people skill enhancement issues described in Chapter 2 and Chapter 3. Figure 16 illustrates how all three components are interrelated in the model. Each of the three pieces is described throughout the chapter and included in a summary at the end.

"Work To Do" structure – The basic workflow within an organization including how overtime, fatigue, complexity, and differing worker skills levels can contribute to the timeliness of original work as well as the rework of discovered work errors.

"People Skill Advancement" structure – A simple mapping of how workers move from novice to expert levels of competency along with how project reassignment, attrition, and mentoring impact the availability of skilled workers for projects within the overall portfolio.

"Portfolio Resource Assignment" structure – A representation of how workers are hired and allocated to projects as well as how things like attrition, retiring, and portfolio reprioritization impact the redistribution of resources.

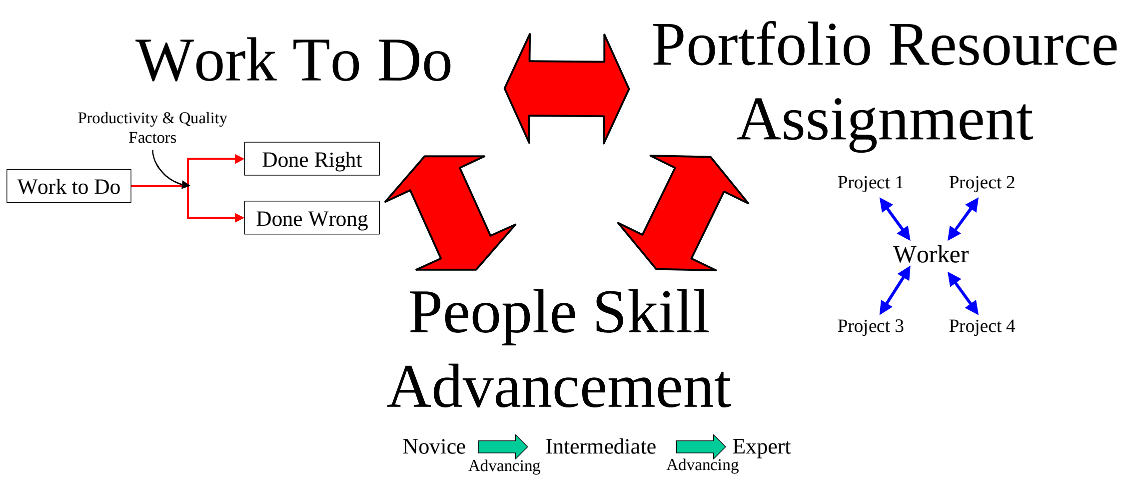


Figure 16 - Overall Model Architecture

[19] A computer model environment by Ventana Systems where variables and their logical interconnections can be simulated over time in order to examine the dynamic behavior of complex systems.

Work To Do

A simplified workflow concept in represented in Figure 17. Some quantity of work to do, represented by a stock[20], is worked upon at some predefined rate. In most typical work environments, some work gets completed correctly and some incorrectly. The "Done Right" and "Done Wrong" stocks represent accumulations of these two conditions. The rate at which work is accomplished is impacted by the productivity of the workforce. Additionally, the probability of completing work correctly is impacted by the work quality at a given instant. In our system dynamics model, both the worker productivity and quality of workmanship can change over time.

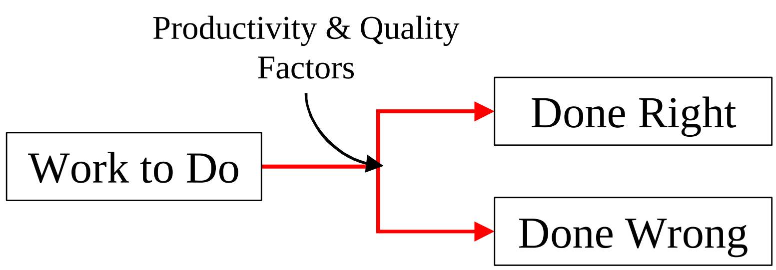


Figure 17 - Work To Do Concept

While Figure 17 serves as a simplified conceptual model of how work flows with a system, Figure 18 shows a more detailed representation of workflow utilizing more conventional system dynamics icons. In this representation, WorkToDo is a stock that is initialized at some starting level. As time moves forward, work get completed at a rate equal to that of Doing. The work rate of Doing is impacted by the current Productivity of the workforce. When work gets completed some flows into the Done Right stock and some to the Hidden Errors stock. The probability that work gets completed correctly is impacted by the current work Quality. Just as is the case with Productivity, the Quality of work can change over time. Another important concept within the model is that work that flows into the Hidden Errors stock is not simply forgotten. Any work that is competed with errors must be reworked. The timeliness of doing work a second time is controlled by the rate of Finding Errors. An important note regarding this model formulation is that in situations of poor work quality, rework cycle may be repeated several times. Thus the combination of lower than expected Productivity and/or Quality can lead to scenarios where

[20] In system dynamic models a stock is an accumulation of things. Stocks create delays by accumulating the difference between inflows and outflows and are the source of disequilibrium in dynamic systems.

completing all the work within the WorkToDo stock takes much longer than originally anticipated.

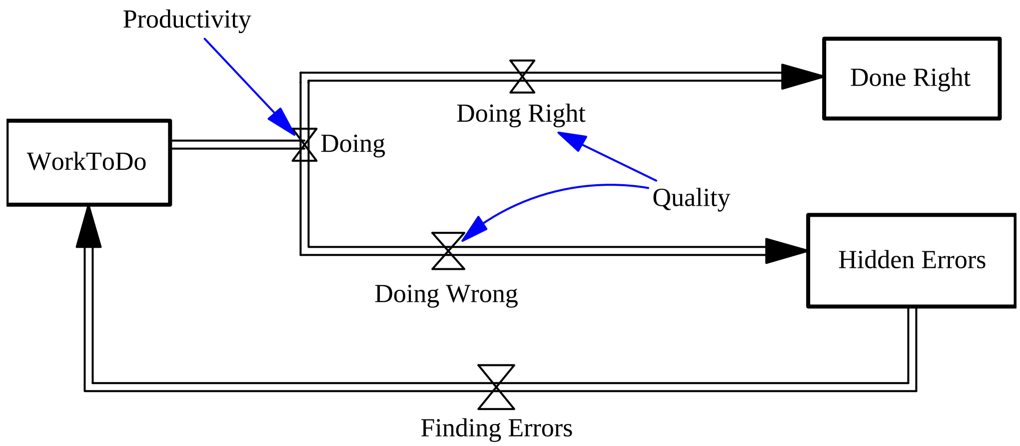


Figure 18 - Simple Work To Do Model

In order to more completely represent the workflow described in Chapter 3, our model cascades four Work To Do submodels together. Just as in a typical development environment, some quantity of Requirements work must be completed before High-Level Design work can complete. While each phase within the development lifecycle has its own stocks of work, the submodels are connected such that the rates of completion in one phase are dependant upon work successfully done in the previous phase. Figure 19 illustrates how these submodels are connected beginning with the Requirements phase and ending with the Test phase. The model contains a separate one of these workflow chains for each of the projects that is active.

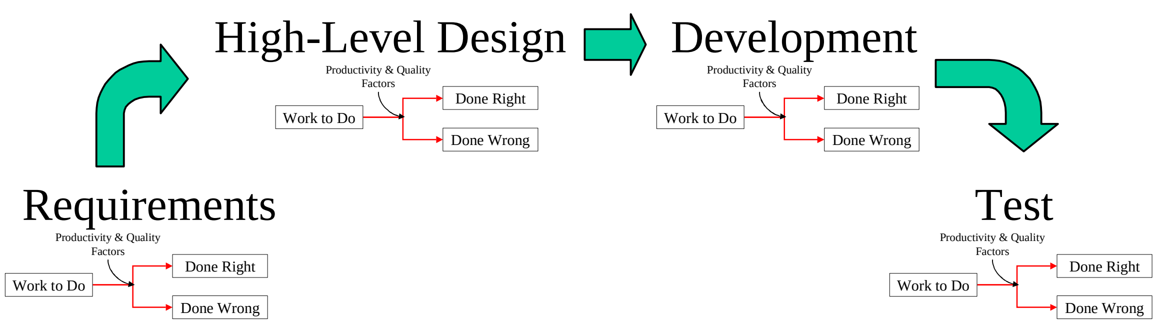


Figure 19 - Overall Project Work Flow

While the simplified representations in Figure 17 and Figure 18 can serve to communicate the basic workflow concept, the actual Vensim representation is much more complete. Figure 20 shows a Vensim view of each Work To Do submodel[21] structure. In addition to the central WorkToDo stock structure, the model comprises other segments that allow for a more realistic simulation of the actual development process. The People portion is utilized to compute the current productivity expected at any given time based upon the number of workers, their skill levels, and their level of fatigue. The Effects portion of the model is used to calculate how things such as project complexity impact the likelihood of work being done correctly. Finally, the Attractiveness[22] section allows for computing the urgency of placing more workers on this phase of the project. It is also the same section that is used for computing whether or not this particular project must roll people off to another project that is currently more important.

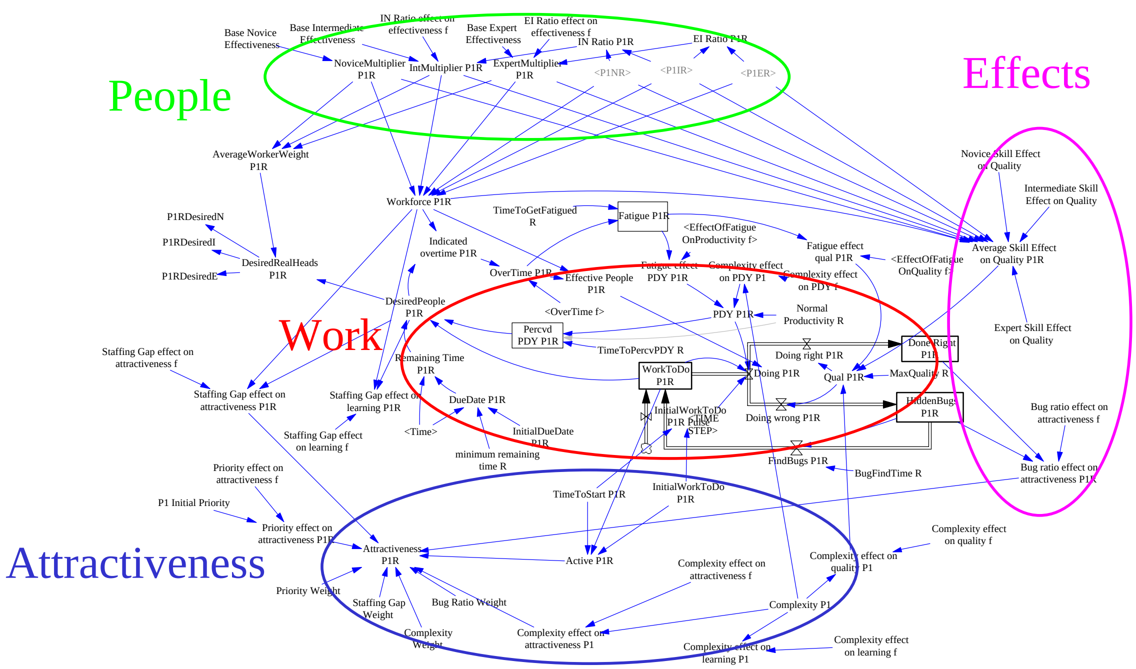


Figure 20 - Complete Work To Do Model

[21] All equations for the model are included in Appendix B.

[22] Attractiveness for a project is a measure for how important the project currently is to managers in charge of allocating resources across the project portfolio. The measure comprises several real-time metrics like Priority, Complexity, and Staffing Gap that are weighted and combined to form a single project comparison factor.

People Skill Advancement

Our model assumes that there are three basic skill levels for workers, Novice, Intermediate, and Expert. Each phase within the project lifecycle utilizes a different combination of workers at these three skill levels. Figure 21 shows how there is a sequential migration of workers from one skill level to the next. Stocks of workers with differing skill levels are used to calculate things such as productivity rates or the amount of work done correctly the first time through the phase.

Figure 21 - Worker Skill Advancement

In addition to simply assuming a fixed number of workers within each phase, the level of each stock can dynamically increase or decrease over time. As shown in Figure 22, the model is able to adjust the number of workers based on hiring, retiring, and attrition. A simplifying assumption is that people are always hired at the Novice level and always retire from the Expert level. While it is certainly possible to hire workers with more subject-matter expertise, there is normally some passage of time and mentoring required until workers are fully productive in a given environment.

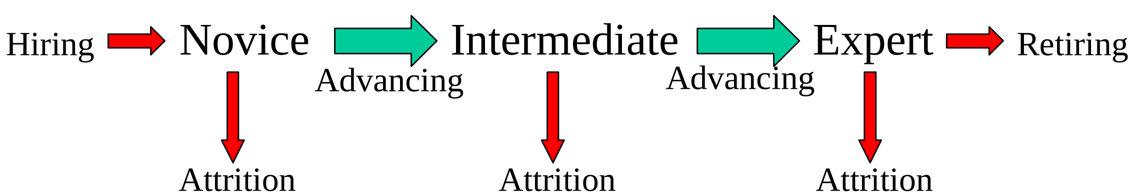


Figure 22 - Worker Hiring, Retiring, and Attrition

Figure 23 depicts the basic Vensim structure necessary for implementing the concept shown in Figure 22. The important thing to note is that each rate of flow between stocks is independently controlled. For instance, the amount of Experts that are leaving the organization due to attrition is different than the amount of Novices that may be leaving. In addition, each phase within each project has separate rate controls. It is possibly to experience increased attrition on one project, based on something like abnormally high levels of fatigue, while another project experiences very low levels of attrition brought on by workers being excited about a new or complex technical challenge.

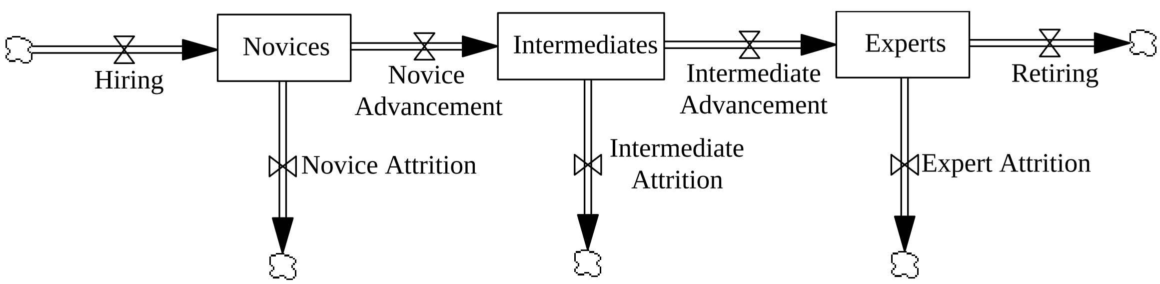


Figure 23 - Worker Skill Advancement Model

Portfolio Resource Assignment

At the same time as workers are progressing from one skill level to the next, there can be a competition for their work-effort between the different projects that may be active at a given time within the portfolio as shown in Figure 24. The Attractiveness measure for a project phase is used to compute which projects must give up resources and which projects are allowed to obtain new resources. An important construct within the model is that all newly hired workers or workers that are in transition between projects must temporarily reside within a core stock for a period of time. This core stock is used for accounting for a worker's unproductive time that is normally associated with context switching time that follows changing jobs or projects. In addition, workers do not get credit for advancing to the next skill level during their temporary stay in the core stock.

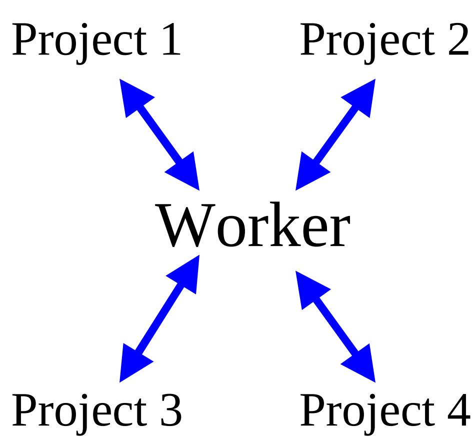


Figure 24 - Portfolio Resource Assignment

Figure 25 depicts the Vensim structure that is responsible for moving workers in and out of the organization as well as between projects. While the example shows the movement of Novice Requirements workers, this structure is utilized for the other phases and skills levels within the project portfolio as well. The flows between each project stock and the core stock are bi-directional allowing for the staff on a project to be increased of decreased. However, the hiring, attrition, and retiring flows are all unidirectional. The most significant contributor to the rate of movement of people is the current Attractiveness of a project phase.

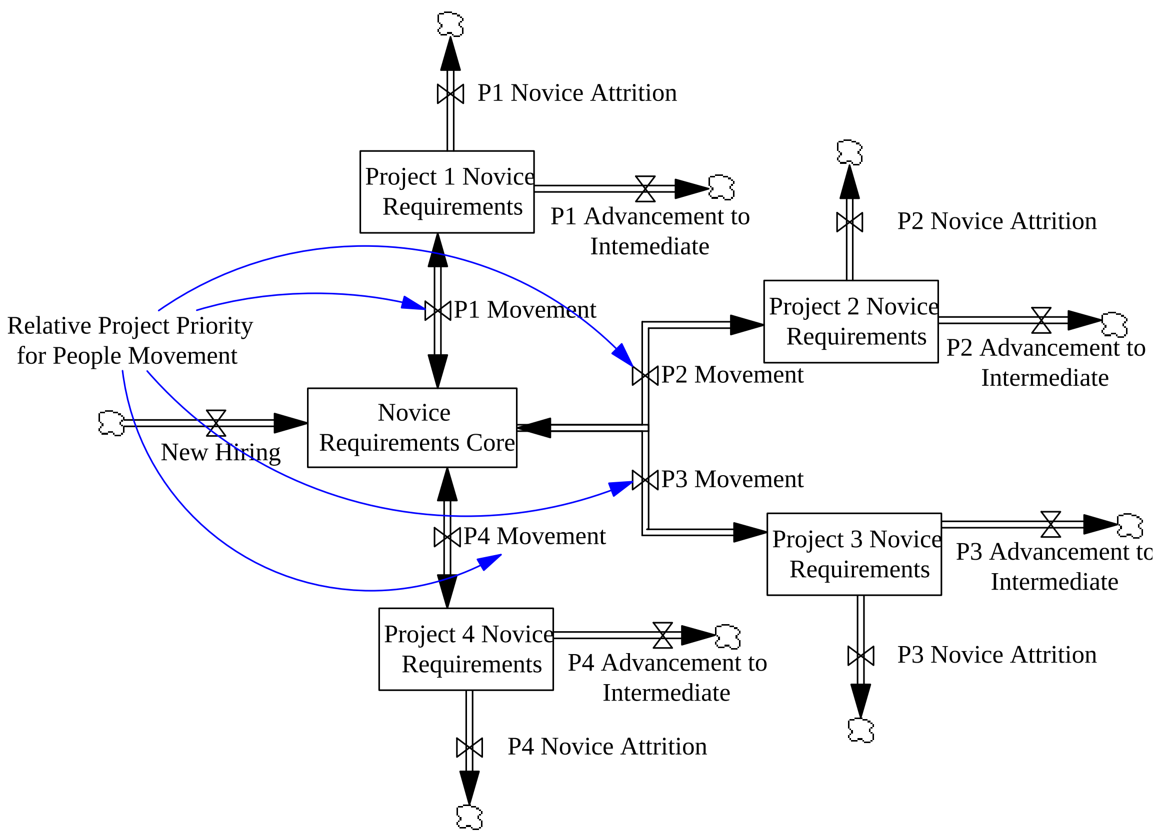


Figure 25 - Worker Assignment Model

Model Structure Summary

The concepts of worker hiring, firing, attrition, skill advancement, and project reassignment are combined within the model to form a complex set of possible movements for any given employee. Figure 26 shows the number of possible paths within a single project phase.

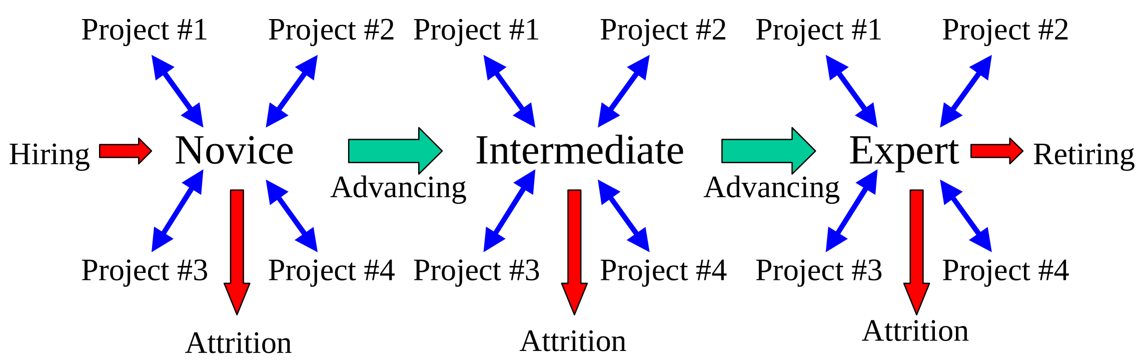


Figure 26 - Worker Skill Advancement with Reassignment

As discussed earlier, an important note is that while the model allows for several different paths for a particular worker, each phase within a project (i.e. Requirements, High-Level Design, Development, and Text) are all staffed separately. While it is certainly possibly within the real world for workers to change specialties, we used the more simplifying assumption that workers typically stay within one specialty for a period of time typically exceeding that of a single project. Figure 27 illustrates the overall worker flow structure for a single project within the model.

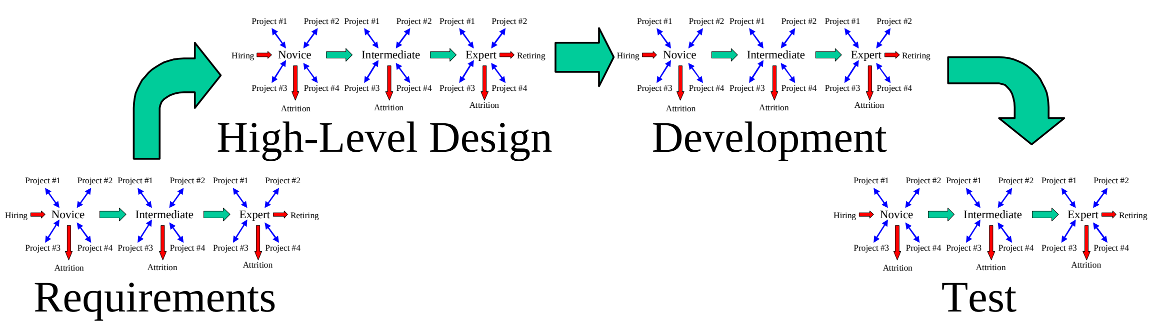


Figure 27 - Multiple Assignments and Advancement Across Work Phases

The model is constructed so that up to four concurrent projects can be simulated at a time. The Vensim structures could be appended so that more projects could be concurrently simulated but four concurrent projects enabled sufficient inter-project dynamics to provide useful insights. When viewed in total, the Vensim model consists of sixteen separate Work To Do submodels all interconnected by sequential phases as depicted in Figure 28. Differing projects can be set to fully overlap, partially overlap, or not overlap at all within the model. This allows for exploring various levels of parallelism in the project portfolio.

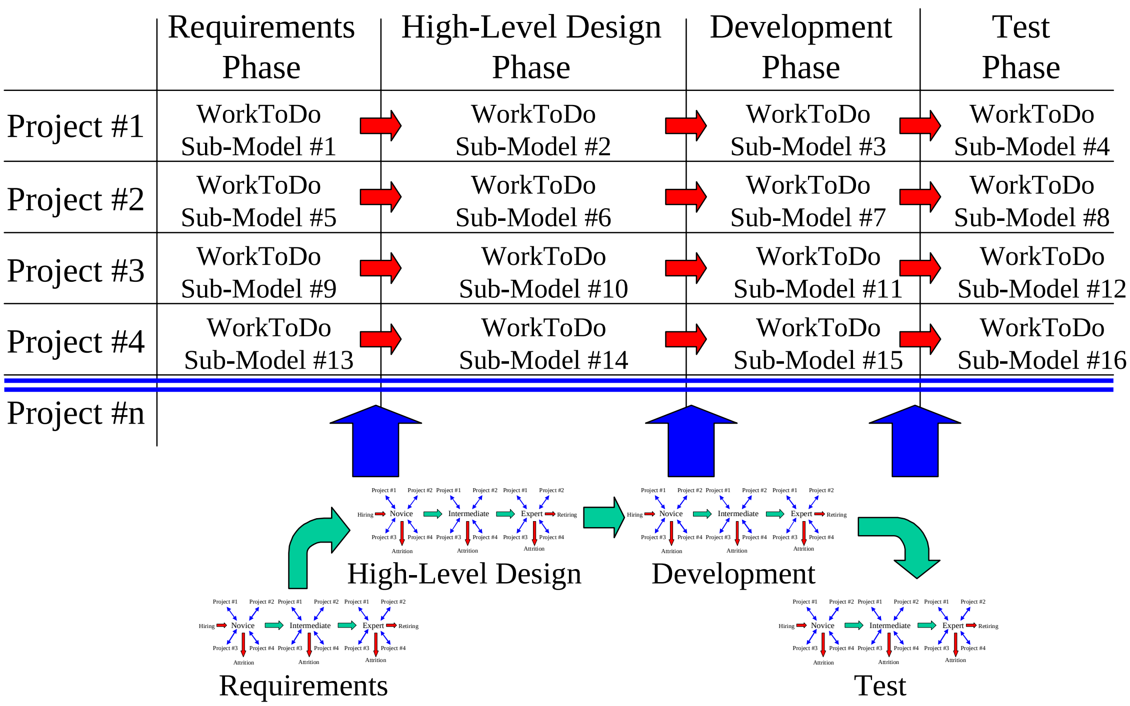


Figure 28 - Work Flow Over Multiple Projects

When viewed from the perspective of worker movement possibilities, the model takes on the shape of that represented in Figure 29. Each of the sixteen submodels is connected to its other three neighboring projects phases of the same type.

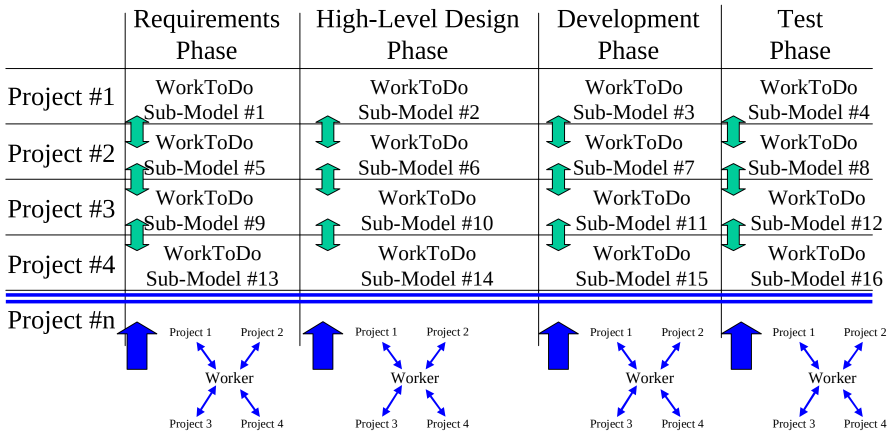
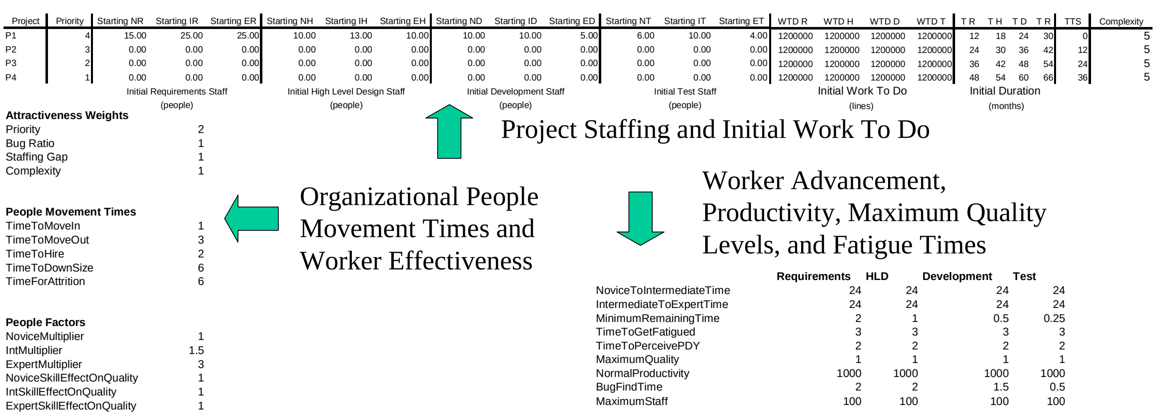


Figure 29 - Worker Reassignment Across Multiple Projects

Even though all equations and constant are modifiable within the Vensim simulations environment, we constructed the model such that all project portfolio configuration parameters are visible in a Microsoft Excel spreadsheet. Vensim reads the Excel file to gather initial values of all equation constants before the start of each simulation run. This architecture also allowed for easy separation of each set of simulation parameters that were explored since each set of data could be stored in its own Excel file. An example spreadsheet is shown in Figure 30. A few other examples for actual simulation runs are also included in Appendix A.



Project	Priority	Starting NR	Starting IR	Starting ER	Starting NH	Starting IH	Starting EH	Starting ND	Starting ID	Starting ED	Starting NT	Starting IT	Starting ET	WTD R	WTD H	WTD D	WTD T	T R	T H	T D	T R	TTS	Complexity
P1	4	15.00	25.00	25.00	10.00	13.00	10.00	10.00	10.00	5.00	6.00	10.00	4.00	1200000	1200000	1200000	1200000	12	18	24	30	0	5
P2	3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1200000	1200000	1200000	1200000	24	30	36	42	12	5
P3	2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1200000	1200000	1200000	1200000	36	42	48	54	24	5
P4	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1200000	1200000	1200000	1200000	48	54	60	66	36	5
			Initial Requirements Staff (people)			Initial High Level Design Staff (people)			Initial Development Staff (people)			Initial Test Staff (people)			Initial Work To Do (lines)			Initial Duration (months)				

Attractiveness Weights	
Priority	2
Bug Ratio	1
Staffing Gap	1
Complexity	1

People Movement Times	
TimeToMoveIn	1
TimeToMoveOut	3
TimeToHire	2
TimeToDownSize	6
TimeForAttrition	6

People Factors	
NoviceMultiplier	1
IntMultiplier	1.5
ExpertMultiplier	3
NoviceSkillEffectOnQuality	1
IntSkillEffectOnQuality	1
ExpertSkillEffectOnQuality	1

	Requirements	HLD	Development	Test
NoviceToIntermediateTime	24	24	24	24
IntermediateToExpertTime	24	24	24	24
MinimumRemainingTime	2	1	0.5	0.25
TimeToGetFatigued	3	3	3	3
TimeToPerceivePDY	2	2	2	2
MaximumQuality	1	1	1	1
NormalProductivity	1000	1000	1000	1000
BugFindTime	2	2	1.5	0.5
MaximumStaff	100	100	100	100

Figure 30 - Sample Simulation Parameter Spreadsheet

Model Formulations

The architecture described previously reflects the translation of the organizational structures and work processes, which are in use at Sikorsky Aircraft and Xerox, into the structure of a system dynamics model. This model captures real world experience through the use of functions that represent effects observed in the workplace. These functions may already be included as a component of the modeling software or modeler defined. Examples of functions that are readily available within the Vensim modeling software include representations for a delay in a process and allocation by priority.

Modeler defined effects are implemented in the system dynamics model through the use of specialized lookup functions. These specialized Lookup functions are simply a list of numbers representing an x-axis and a y-axis. The inputs are positioned relative to the x-axis and the outputs are read from the y-axis. A description of the important modeler defined Lookup functions that were utilized is presented in the discussion that follows.

The model formulations, both linear and non-linear, may be divided into two categories; formulations related to work and formulations related to workforce. These formulations are represented in Table 1.

Work	Workforce
Productivity • Effect of Fatigue • Complexity Effect	Effectiveness • Novice to Intermediate Ratio Effect • Intermediate to Expert Ratio Effect
Quality • Effect of Fatigue • Complexity Effect	Learning • Staffing Gap Effect • Complexity Effect
Overtime • Overtime Effect	Hiring • Gap Effect
Attractiveness • Bug Ratio Effect • Complexity Effect • Priority Effect • Staffing Gap Effect	Attrition • Fatigue Effect • Complexity Effect

Table 1 - Model Formulations for Work and Workforce

The formulations related to work involve primarily effects to productivity and quality. Also included are overtime and project attractiveness effects on the completion of work. Effects that impact workforce effectiveness, skill advancement through learning, hiring and attrition are also discussed. Lookup functions will be presented where necessary through the use of figures in the text in order to enhance the explanation.

Work

Work is the result of productive people. The amount of work performed correctly or incorrectly determines the quality. Both factors are represented in Figure 31 and will be discussed in more detail.

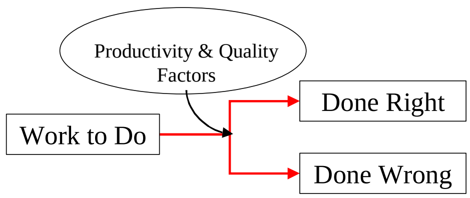


Figure 31 - Productivity & Quality Factors

Productivity

Productivity is a primary factor in the work section of the model. Productivity effects the rate at which work is completed. Employee fatigue and project complexity are two effects on productivity that were included in the system dynamics model. A normal level of productivity is reduced or improved through the application of these factors. The fatigue effect is depicted in Figure 32.

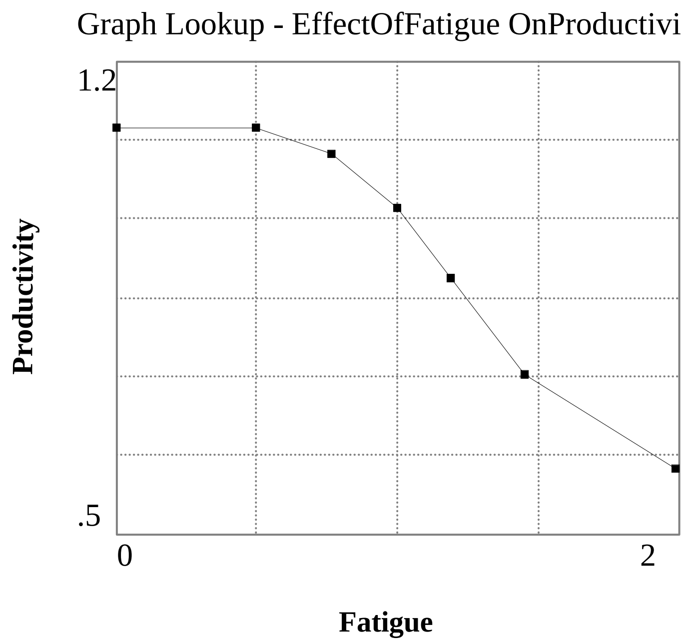


Figure 32 - Effect of Fatigue on Productivity

The fatigue effect on productivity is non-linear as shown. The time to get fatigued is set by project phase as different projects phases may require more or less effort depending on the amount of staff available to address the total level of work. Time to fatigue is currently set at three months for all four phases and is adjusted through the use of the model front-end data sheet. Once fatigue sets in, productivity is effected as shown in the curve. When fatigue is low workers perform at their full capacity. As fatigue increases, productivity is reduced gradually at first then more rapidly. Productivity is reduced to approximately sixty percent (60%) when the employee reaches a fatigue level that is twice that normally experienced in the workplace.

This fatigue is representative of work demands such as excessive multitasking and long hours of attention. The long hours may be the result of overtime policies and will be discussed later in this section.

The complexity effect on productivity is subtle. Project complexity leads to a gradual linear change in worker productivity. When project complexity is low workers operate at full levels of productivity. Projects that have the highest complexity rating reduce worker productivity to eighty percent (80%). This reduction in productivity may be the result of time spent learning new information or techniques that address the technical complexity introduced into the design solution.

Quality

Fatigue also effects the quality of work. As workers tire, more mistakes may be made. Quality remains linear as fatigue is increased to fifty percent (50%) greater than normally experienced. A rapid drop-off in quality to a level of sixty-five percent (65%) occurs when fatigue is double that normally experienced in the work environment.

The complexity effect on quality is linear and very slight. Quality is at normal level for projects with low complexity and drops to ninety percent (90%) when complexity is highest. Complexity may increase the desire to do the work such that more attention is given but because the work challenges existing skill sets, some mistakes will be made.

Overtime

When the amount of work exceeds the number of people available in the timeframe required, overtime is usually applied in order to catch up. This overtime will impact the fatigue level of the employee base. The effect is depicted in Figure 33. Indicated overtime is the amount of overtime necessary in order to finish on schedule. This amount of hours may exceed what the workforce is physically capable of or the number of hours in a day. For these reasons, the indicated overtime is conditioned through the non-linear function and limited to 1.5 times the standard workweek.

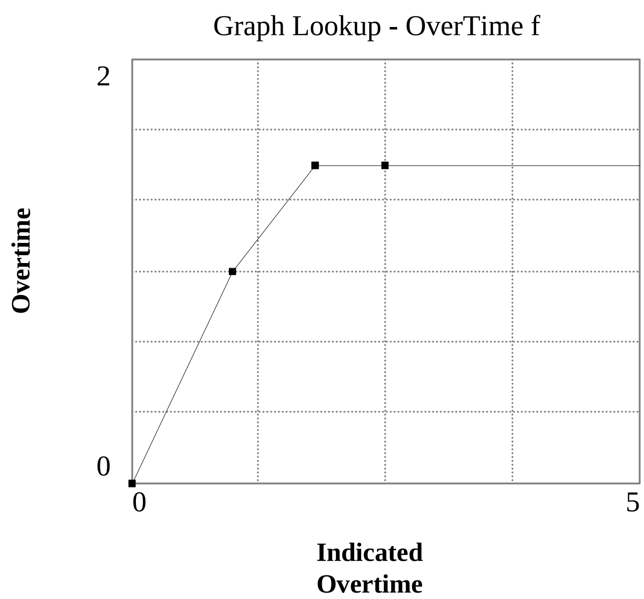


Figure 33 - Overtime Effect

Attractiveness

Projects are provided resources based on their relative attractiveness to one another. The number of errors produced, complexity of the technology involved, project priority level assigned and the shortfall of resources influence this measure. The effects are applied to weighting factors that the organization can adjust. For instance if the company only makes low technology projects the weighting on project attractiveness can be adjusted to lessen the effect of project technical complexity in resource allocation decisions.

Attractiveness is reduced linearly as errors are produced such that when there are no errors the project is very attractive. When there are a high number of errors the project attractiveness is zero. This serves as a means of penalizing a project that is performing poorly with respect to quality.

Project complexity is also a linear contribution to attractiveness. Projects of low complexity lead to a low attractiveness number because these types of projects have little influence in advancing worker skills and contributing to increased intellectual capital. Projects of a high degree of technical complexity produce the full effect of the complexity weighting factor and thus, high attractiveness numbers.

The project priority effect is applied linearly to the project priority weighting. Projects with low priority have low attractiveness and receive little attention in the competition for scarce resources. Attractiveness is a ten when project priority is highest.

Staffing levels that fall short of the amount required for the level of work and timeframe influence the attractiveness in a non-linear manner, reference Figure 34. Low staffing gap contribution to attractiveness results when the desired level of project staffing is achieved. As the staffing shortage grows, so too does the attractiveness. Attention is elevated, as projects fall short of manpower required to complete the work on schedule.

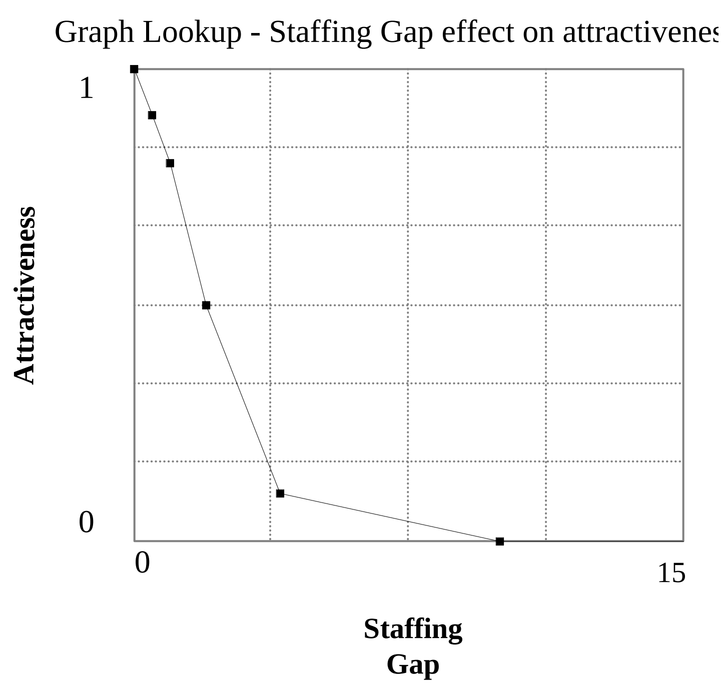


Figure 34 - Staffing Gap Effect on Attractiveness

Workforce

The system dynamics model represents the effectiveness of senior level employees as they mentor junior level employees, as well as worker skill advancement from novice to intermediate to expert skill levels. Changes in the workforce as a result of hiring and attrition are also present in the model structure.

Effectiveness

The productivity level of expert and intermediate skill employees is effected by the amount of mentoring performed. The model is structured such that expert skill level employees mentor intermediate skill level employees. Intermediate skill employees mentor novices. In each case the expert and intermediate employees are less effective as a result of the time spent away from productive work. This non-linear function maintains effectiveness at high level as long as the ratio of novices to intermediates is four-to-one or less. The reduction in effectiveness grows rapidly as the novice to intermediate mismatch gets larger than four. Intermediates are only ten percent (10%) as effective when novices exceed intermediates by a factor of 100. The same function is applied to expert skill level effectiveness as a result of mentoring to intermediate skill level employees.

Learning

Skill advancement takes place as a result of learning while doing. This skill advancement as structured in the system dynamics model is represented in Figure 35. Two effects impact the ability to learn, the shortage of people or staffing gap, and the technical complexity of projects.

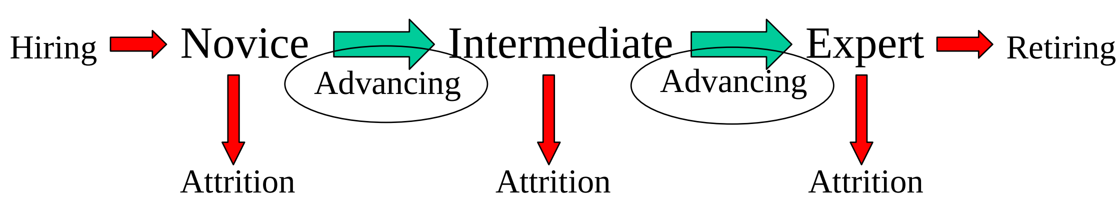


Figure 35 - Skill Advancement

Figure 36 represents the non-linear function for the effect on learning as a result of staffing gap. Learning is significantly impacted as the difference between desired and required staffing levels grows. This is attributed to the fact that there is much more work than people to do it. Workers shift into a reactionary mode and little time is dedicated to reflect on work completed or in process. Also, in situations such as this context switching is much more likely as workers are reallocated to different projects to address manpower shortfalls.

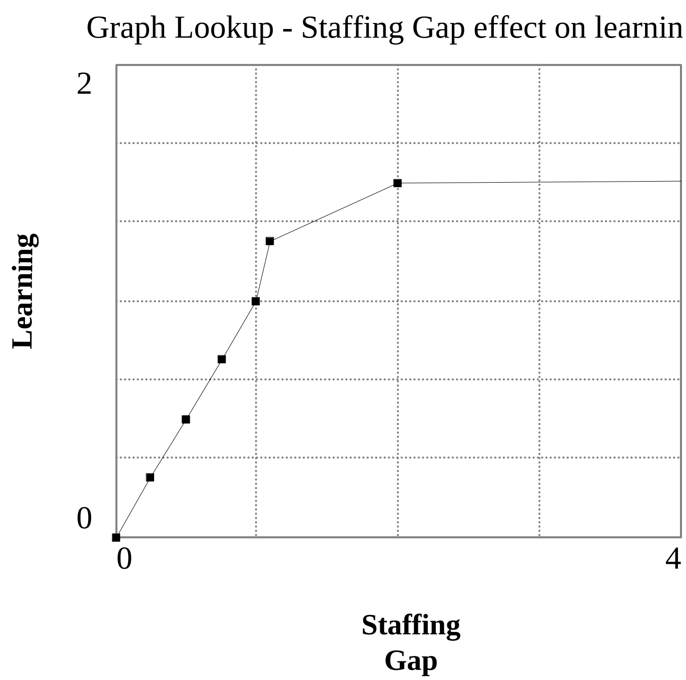


Figure 36 - Staffing Gap Effect on Learning

Project technical complexity also impacts learning and skill advancement in a linear manner. Projects with a high degree of technical complexity stimulate learning to a level fifty percent (50%) greater than for average levels of technical complexity. Mundane work, which results from projects with low technical complexity, contributes little to learning.

The normal hiring process is effected by shortages in staff. As the staff shortage increases, the hiring process needs to be adjusted in order to keep up. This non-linear effect is depicted in Figure 37. The normal rate of hiring is adjusted through the input data sheet to reflect organizational characteristics. Organizational as well as market factors may limit the amount of hiring that ultimately can take place. For these reasons, hiring is limited to twice the normal rate as represented by the function.

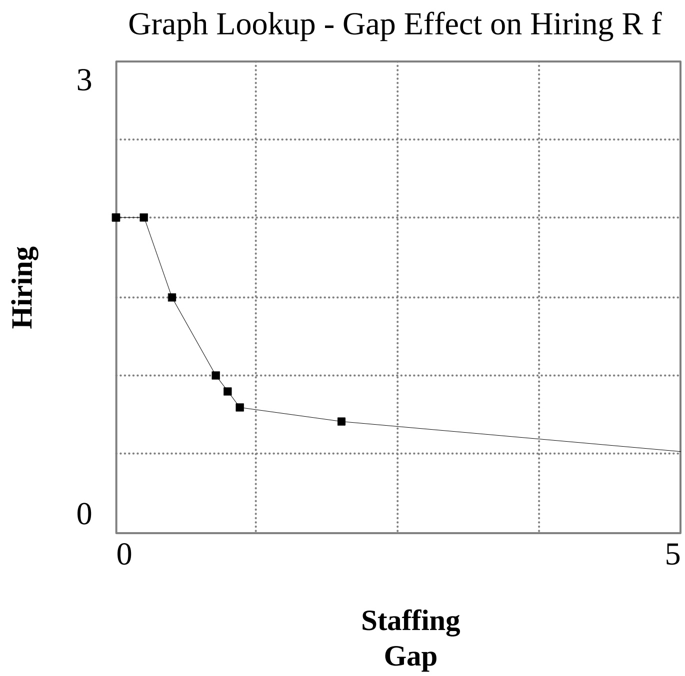


Figure 37 - Staffing Gap Effect on Hiring

Attrition

Workers may leave the workplace at any time. This is reflected in the model structure through attrition at the novice, intermediate and expert skill levels. Fatigue and complexity are the primary factors effecting attrition.

The fatigue effect on attrition is represented in Figure 38. Attrition is present at normal levels of fatigue. However, as fatigue is accelerated to levels twice that of the normal work environment attrition increases to ten times that of the normal rate.

.

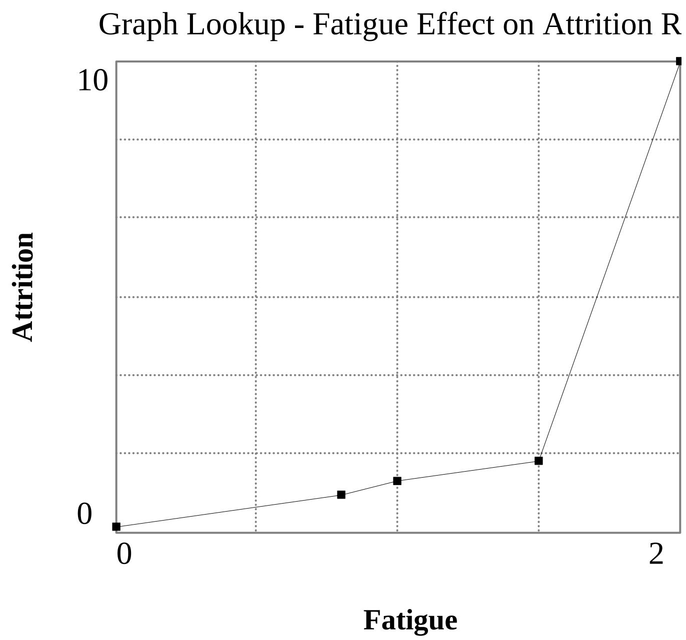


Figure 38 - Fatigue Effect on Attrition

Project technical complexity has just the opposite effect as fatigue. Low complexity routine work does not stimulate the workforce due to the low level of learning opportunities. This lack of learning increases attrition as people leave the company to pursue better work opportunities. Learning opportunities that lead to skill advancement encourage employees to stay. Increases in the technical complexity reduce attrition to seventy percent that of the normal levels of attrition.

Model Formulations Summary

Some of the characteristics that make Sikorsky Aircraft and Xerox unique have been captured and represented in a system dynamics model. This was achieved through the use of Lookup functions. These specialized functions were created to represent effects on work and the workforce as observed today. The importance of these effects in the formulation of mental models will be explored in the analysis that follows in Chapter 6.

Chapter 6: Model Analysis

The system dynamics model developed was exercised through twenty-one (21) cases that represent differing model parameter inputs. These model parameter input changes were made to explore the targeted behavior that each case was intended to represent. These cases are displayed in Appendix A. Each case is summarized with respect to case description, rational for use, and parameter changes with respect to a base case. The results are in a standard output format that is consistent for all cases. These results summarize total intellectual capital as measured in people, total project effort measured in people-months, total number of novice, intermediate and expert skill level people. In addition, project duration by total and phase for each project is also included.

The broad level of parameter permutations that these twenty-one cases represent provided for ample combinations when applied in support of a validity check for the hypotheses developed in Chapter 4. Detailed discussion of differences in perfect and imperfect quality, loss of productivity, on-time completion of projects, attrition, progression of workers between skill levels, and intellectual capital growth follow. Summary statements are included along with a table that maps the test cases of Appendix A to policy categories. These policy categories will be explored further in the Chapter 7 recommendations. A summary of the cases discussed in this chapter is shown in Table 2

Discussion	Cases Discussed
Differences in Perfect and Imperfect Quality	Case #1: Base - Perfect Quality Case #2: Base - Imperfect Quality
Loss of Productivity	Case #2: Base - Imperfect Quality Case #4: Complex Projects
On-Time Completion of Projects	Case #2: Base - Imperfect Quality Case #6: Inter-Project Gaps
Attrition	Case #2: Base - Imperfect Quality Case #8: Project 3 as High Priority
Progression of Workers between Skill Levels	Case #2: Base - Imperfect Quality Case #8: Project 3 as High Priority
Intellectual Capital Growth	Case #8: Project 3 as High Priority

Table 2 - Summary of Cases Discussed

Table 3 and Table 4 lists all the cases that are included in Appendix A along with a brief description of each and its rationale for being included in the documented set of analyzed simulations.

Category	Cases Discussed
Project Focus	Case #1: Four Sequential Projects, Same Duration, Perfect Quality • Base Case for Sequential Projects with Perfect Quality Representative of an Ideal World Where No Mistakes are Made by Employees. Case #2: Four Sequential Projects, Same Duration, Imperfect Quality • Serves as Base Case for Comparison to Other Cases and Demonstrates Existing Mental Model that Current Projects are More Important than Future Projects. Case #4: Four Sequential Projects, Same Duration, All Complex • Look at Extreme Case for Retaining & Developing Intellectual Capital. Case #10: Four Sequential Projects, Same Duration, Set Maximum Staff • Explore the Effect of a Constrained and Unconstrained Number of People on Project Completion Time. Case #11: Four Sequential Projects, Same Duration, Adjust Time to Hire • Explore Ability of Organization to Meet Resource Demand Case #13: Four Parallel Projects, Same Duration, Perfect Quality • Base Case for Parallel Projects with Perfect Quality Representative of an Ideal World Where No Mistakes are Made by Employees. • Also, Explore the Effect of Project Concurrency on Firefighting. Case #14: Four Parallel Projects, Same Duration, Imperfect Quality • Base Case for Parallel Projects with Perfect Quality Representative of an Ideal World Where No Mistakes are Made by Employees. Case #15: Four Parallel Projects, Same Duration, All Complex • Look at Extreme Case for Retaining & Developing Intellectual Capital. Case #16: Four Parallel Projects, Same Duration, • Complexity; Simple, Complex, Simple, Complex • Intermediate Complexity Case
Resource Allocation	Case #6: Four Sequential Projects, Same Duration, Equal Small Gaps Between Projects • Test effect of Allowing Space for Rework Discovery & Other Unforeseen Circumstances. Case #7: Four Sequential Projects, Same Duration, Vary Bug Find Time • Explore Effect of Projects with Delayed Rework Discovery on Other Projects. Case #9: Four Sequential Projects, Same Duration, Adjust Time for Resources to Move Off a Project • Test Effect of Manpower Availability for Discovery of Rework Case #11: Four Sequential Projects, Same Duration, Adjust Time to Hire • Explore Ability of Organization to Meet Resource Demand

Table 3 - Cases included in Appendix A (Part 1)

Category	Cases Discussed
Resource Allocation (cont.)	Case #11: Four Sequential Projects, Same Duration, Adjust Time to Hire • Explore Ability of Organization to Meet Resource Demand Case #12: Four Sequential Projects, Same Duration, Adjust Time to Downsize. • Explore Organizational Tolerance to Unassigned Staff and Impact to Learning. • Also, Explore Impact on Completing Rework as it is Discovered.
Project Scheduling	Case #3: Four Sequential Projects, Same Duration, Remove Project 3 • Impact of Removing the Overburden. Case #6: Four Sequential Projects, Same Duration, Equal Small Gaps Between Projects • Test effect of Allowing Space for Rework Discovery & Other Unforeseen Circumstances. Case #9: Four Sequential Projects, Same Duration, Adjust Time for Resources to Move Off a Project • Test Effect of Manpower Availability for Discovery of Rework
Project Priority	Case #3: Four Sequential Projects, Same Duration, Remove Project 3 • Impact of Removing the Overburden. Case #4: Four Sequential Projects, Same Duration, All Complex • Look at Extreme Case for Retaining & Developing Intellectual Capital. Case #8: Four Sequential Projects, Same Duration, Project 3 High Priority • Demonstrate Effect of Unexpected Priority Shifts. • Priority Shifts May be Due to Changes in Market Conditions. Case #13: Four Parallel Projects, Same Duration, Perfect Quality • Base Case for Parallel Projects with Perfect Quality Representative of an Ideal World Where No Mistakes are Made by Employees. Case #14: Four Parallel Projects, Same Duration, Imperfect Quality • Base Case for Parallel Projects with Perfect Quality Representative of an Ideal World Where No Mistakes are Made by Employees. Case #15: Four Parallel Projects, Same Duration, All Complex • Look at Extreme Case for Retaining & Developing Intellectual Capital. Case #16: Four Parallel Projects, Same Duration, • Complexity; Simple, Complex, Simple, Complex (Intermediate Complexity Case) Case #17: Four Parallel Projects, Same Duration, Cancel Lowest Priority Project 3, Re-allocate Workers to Highest Priority Project 1. • Test Impact of Reducing Competition for Resources by Eliminating a Low Priority Project and Re-allocating that Staff to the High Priority Project.
Portfolio Balance	Case #4: Four Sequential Projects, Same Duration, All Complex • Look at Extreme Case for Retaining & Developing Intellectual Capital. Case #5: Four Sequential Projects, Same Duration, Complexity; Simple, Complex, Simple, Complex • Intermediate Complexity Case

Table 4 - Cases included in Appendix A (Part 2)

Differences in Perfect and Imperfect Quality

Case #1 and Case #2 illustrate how imperfect work quality, the normal real world situation, can drive the overall completion time of projects. The completion time for Project 4, the last project in the portfolio, is completed 20 months earlier in Case #1 than in Case #2. While Case #1 assumes perfect work quality from all workers (Novices, Intermediates, and Experts), Case #2 sets the work quality levels to 92%, 95%, and 98% respectively for Novice, Intermediate, and Expert skill levels. Figure 39 and Figure 40 show the tremendous difference in the Work to Do[23] accumulation as well as the completion times.

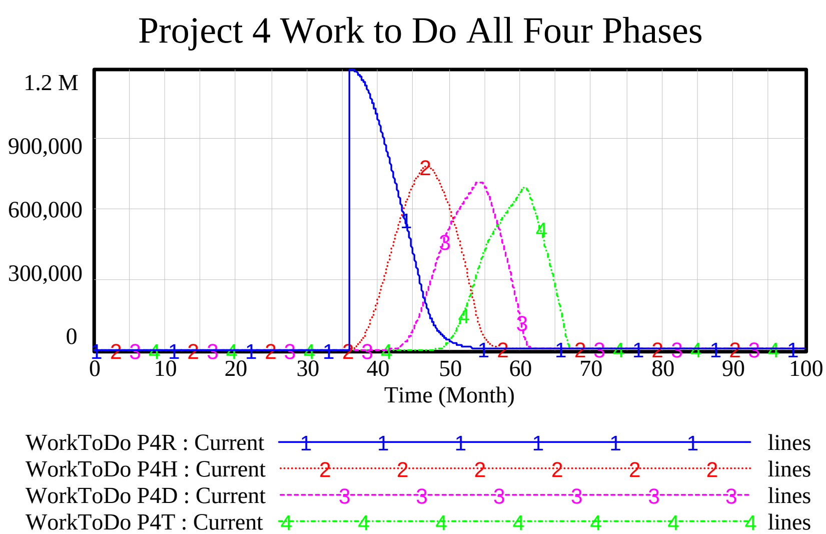


Figure 39 - Case #1 Work to Do (Base - Perfect Quality)

Project 4 in Case #2 accumulates much more Work to Do based on less than perfect work quality as well as from a lack of resources. The reasons are essentially three-fold. First, the workers on Project 4 in Case #2 generate more errors and thus add more rework to the Work to Do stock. Second, there are less worker resources available to Project 4 in Case #2 since they have been allocated to previous projects in the portfolio that have pending rework resulting from imperfect quality. Third, the fact the Project 4 is behind schedule results in more worker fatigue. This fatigue result in more attrition and later the hiring of less experienced workers.

[23] All projects utilize “lines” (short for “lines of code” as taken from the software industry) as the unit of work. However, “line” can be viewed in a broader sense a simply a generic work unit.

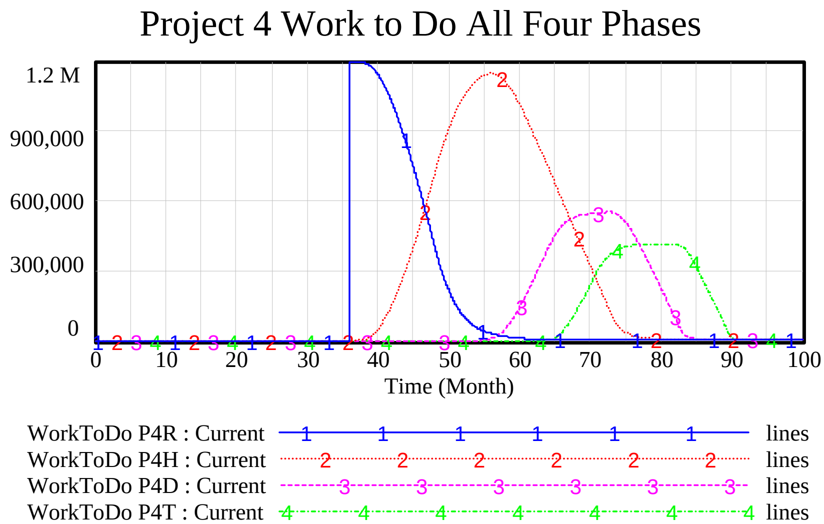


Figure 40 - Case #2 Work to Do (Base - Imperfect Quality)

Figure 41 and Figure 42 show the basic difference between the perfect work quality case (Case #1) and the imperfect work quality case (Case #2). Case #1 never accumulates any hidden bugs, while Case #2 peaks at over 42,000 just during the development phase of the project. Again, accumulating hidden rework just exacerbates the already tough situation that projects within Case #2 face. The hidden rework will eventually be discovered, leading to more work in the Work to Do stock which will place projects even farther behind schedule. Projects slipping behind schedule will cause management to lean towards things like more overtime which leads to fatigue, decreased work quality, and increased attrition.

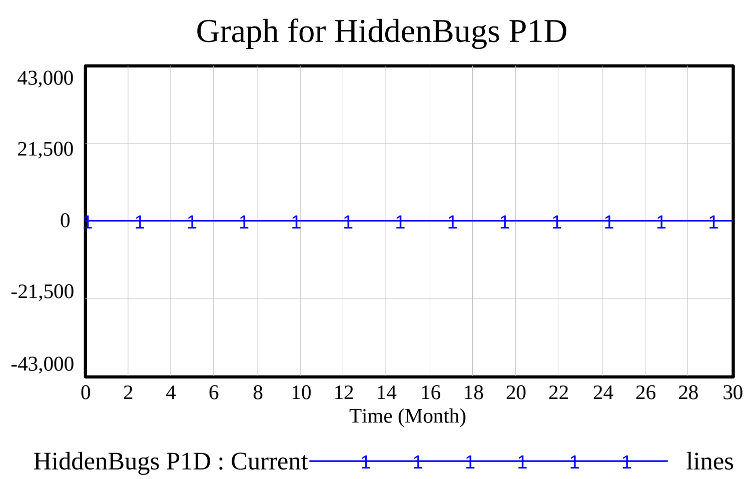


Figure 41 - Case #1 Hidden Bugs (Base - Perfect Quality)

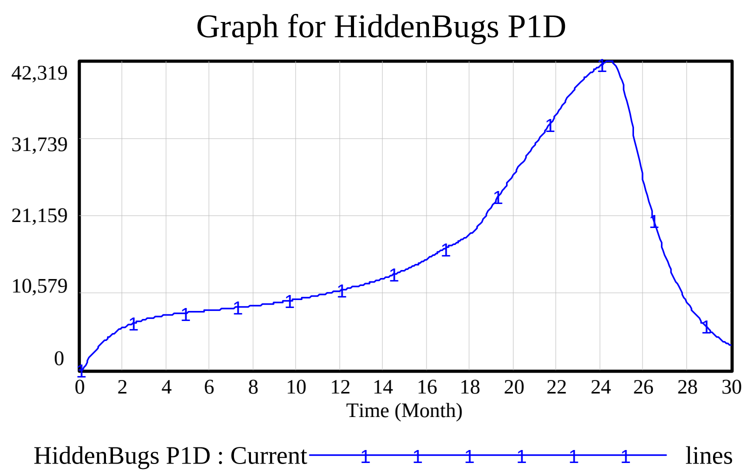


Figure 42 - Case #2 Hidden Bugs (Base - Imperfect Quality)

One of the significant portfolio differences between Case #1 and Case #2 is the fact that all projects within Case #1 can operate with substantially less employee fatigue. One of the root causes for fatigue is the long work hours that are brought on by lack of staff and/or unplanned rework. Figure 43 and Figure 44 illustrate the differing levels of fatigue between the development phases of Project 1.

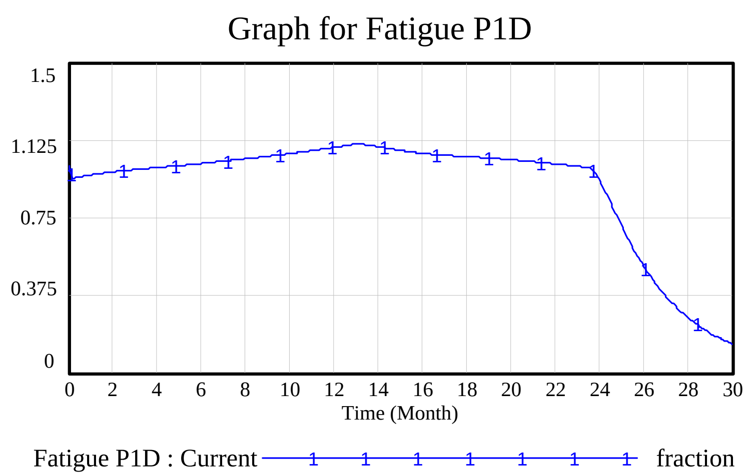


Figure 43 - Case #1 Fatigue (Base - Perfect Quality)

Case #1 exhibits minor fatigue during the middle of the development phase resulting from a slight under-staffing situation but not from rework. Case #4, on the other hand, experiences a growing level of fatigue throughout most of the project based partially upon the extra work required to redo work, possibly multiple times.

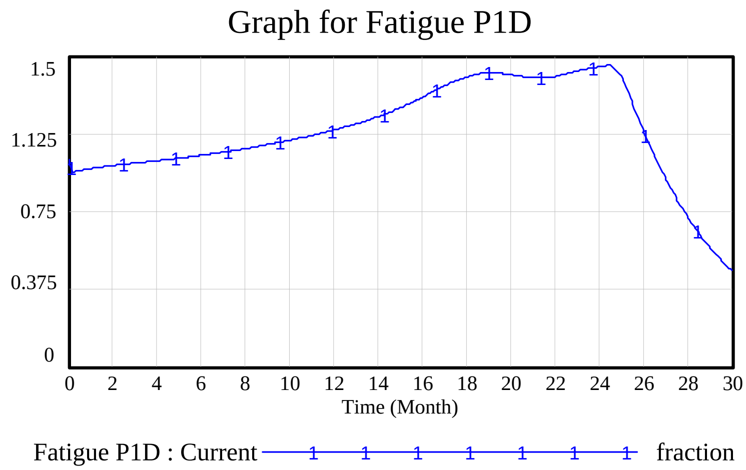


Figure 44 - Case #2 Fatigue (Base - Imperfect Quality)

Fatigue is a delayed reaction to assigned overtime work. Figure 45 and Figure 46 show the directly calculated overtime that would be required to complete the development phase of Project 1 on schedule. The fatigue graphs in Figure 43 and Figure 44 are really just dampened images of the overtime graphs since workers can normally respond to shorter durations of unplanned work without the project suffering.

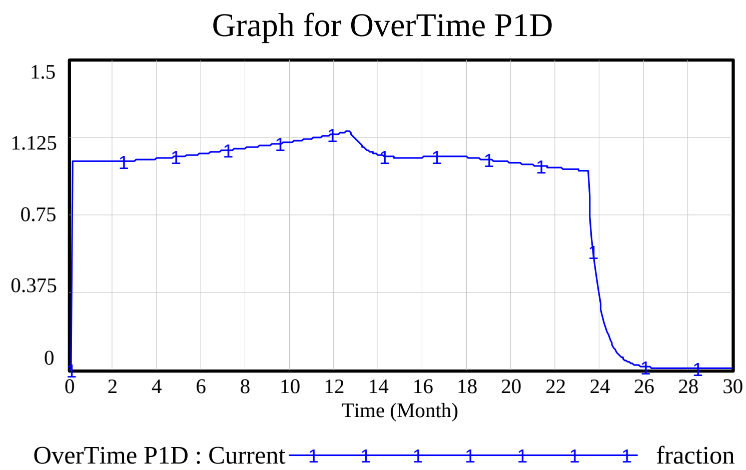


Figure 45 - Case #1 Overtime (Base - Perfect Quality)

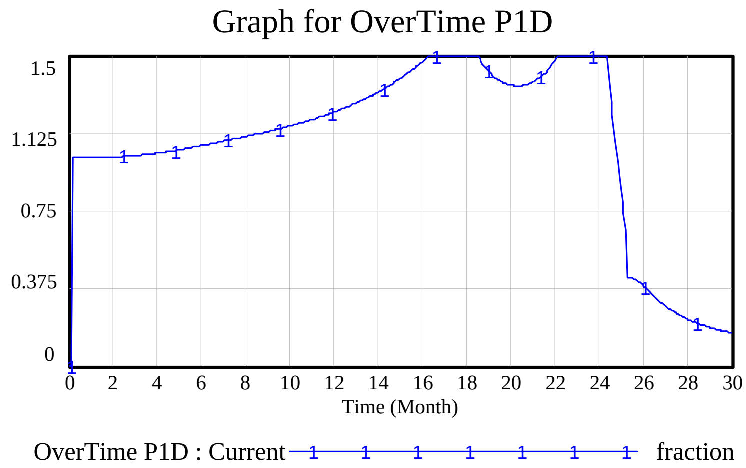


Figure 46 - Case #2 Overtime (Base - Imperfect Quality)

Figure 47 shows how the Work to Do for the development phase of Project 1 in Case #1 steadily grows as work is added to the queue based upon the amount of work completed in the previous phase. There are at least two things to note regarding the accumulation of Work to Do. First, Case #1 is able to initially add work to the development phase of Project #1 much faster than is Case #2. This is due to the fact that successful work is being accomplished more rapidly in the preceding phase (Requirements) in Case #1.

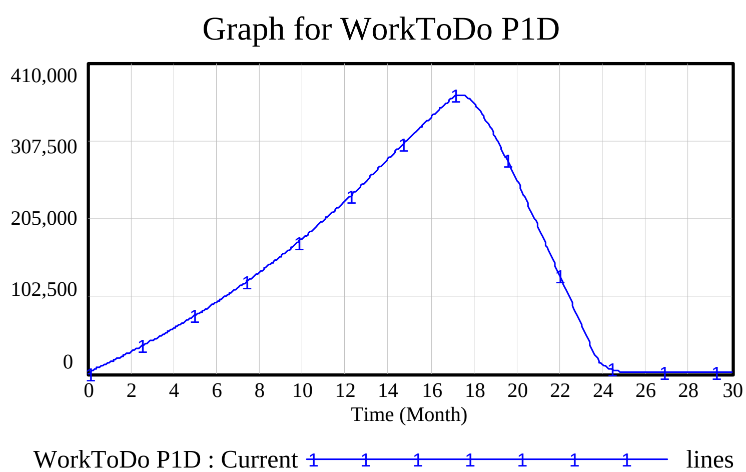


Figure 47 - Case #1 Work to Do (Base - Perfect Quality)

Second, Case #2 peaks at a higher value for Work to Do than does Case #1. Figure 48 illustrates how the development phase of Project 1 in Case #4 gains Work to Do at a faster rate between month 14 and month 18, again due largely to discovered rework. In addition, Case #4 peaks over 40,000 more work items than show in Case #1.

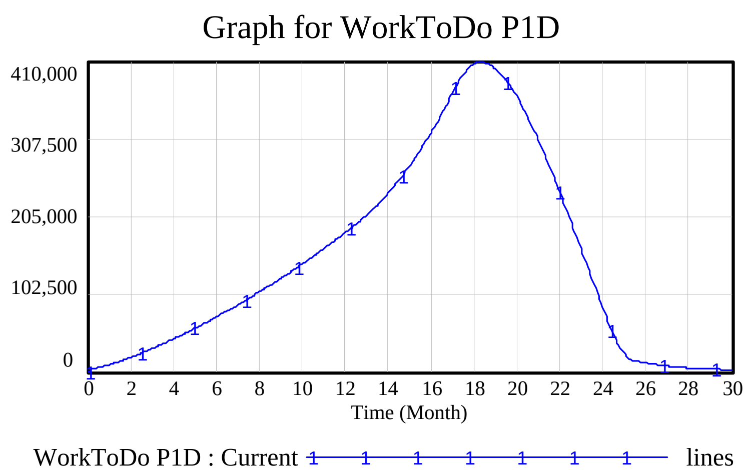


Figure 48 - Case #2 Work to Do (Base - Imperfect Quality)

Case #2 illustrates how the simply changing one assumption within the model can significantly impact the overall portfolio behavior. In this situation, reducing the overall work quality can lead to significantly degraded organizational performance. The fact that projects must compete for resources within a multi-project portfolio can lead to situations where one project is significantly disadvantaged in regards to resource allocation. Once one project gets behind schedule, it can start pulling resources from other projects that may have been in reasonably close to being on schedule. The end result can be that all projects within the portfolio are late if no overt action is taken to change the project interdependency dynamics.

Loss of Productivity

Case #2 and Case #4 illustrate the result that differing amounts of employee productivity can have on the overall project portfolio. While Case #2 shown in Figure 49 only requires a total of 26,000 people-months to complete all four projects, Case #4 shown in Figure 50 requires more than 29,000 people-months to complete all projects in the portfolio. Similar to the situation with decreased work quality, decreased productivity can lead to situations where projects that are scheduled to start later or projects that are of lower priority end up suffering.

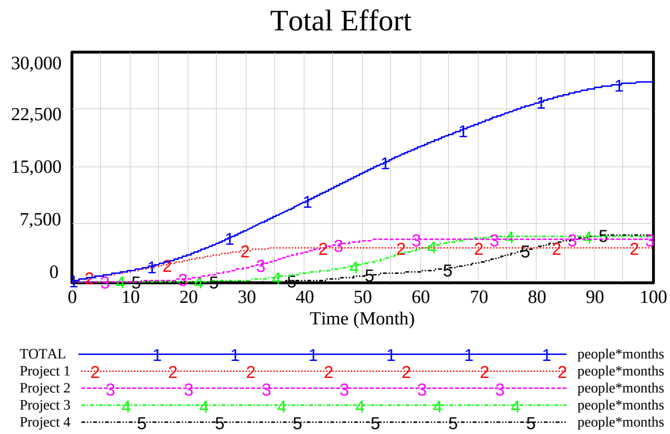


Figure 49 - Case #2 Total Effort (Base - Imperfect Quality)

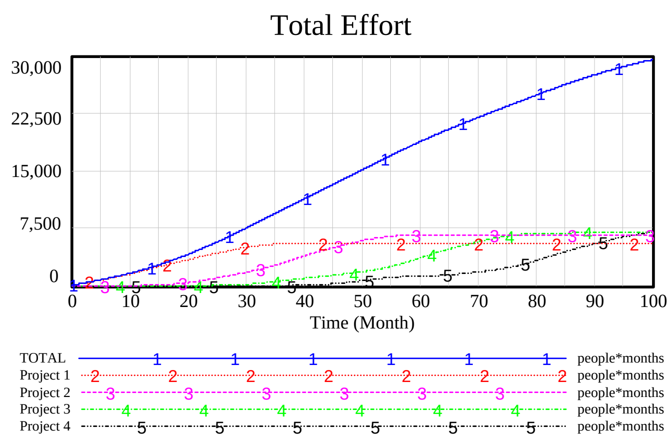


Figure 50 - Case #4 Total Effort (Complex Projects)

While Case #2 consumes more total effort than required by Case #1 (i.e. perfect work quality) it does not consume nearly as much as Case #4. The difference in the two cases is that all four projects within Case #4 are of very high complexity, while all the projects within Case #2 are of medium complexity. This additional complexity ends up driving more total work, rework, fatigue, and attrition, all of which contribute to driving down overall productivity. Figure 51 and

Figure 52 show the differing levels of perceived productivity during the development phases of Project #2 in the two cases.

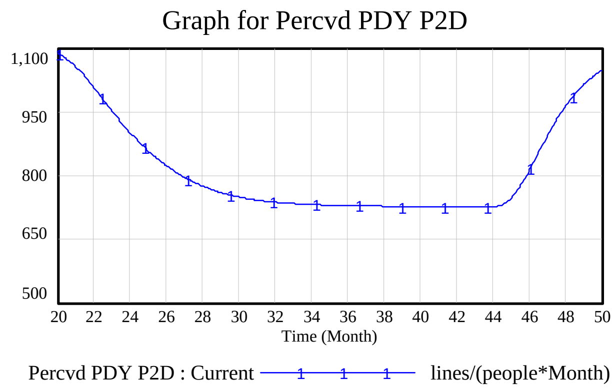


Figure 51 - Case #2 Productivity (Base - Imperfect Quality)

The increased complexity of the projects in Case #4 end up driving down productivity to almost half its standard level of around 1000 lines per person-month. Productivity at this low level requires some combination of extra work, extra people, or more project time, all of which exacerbate the situation.

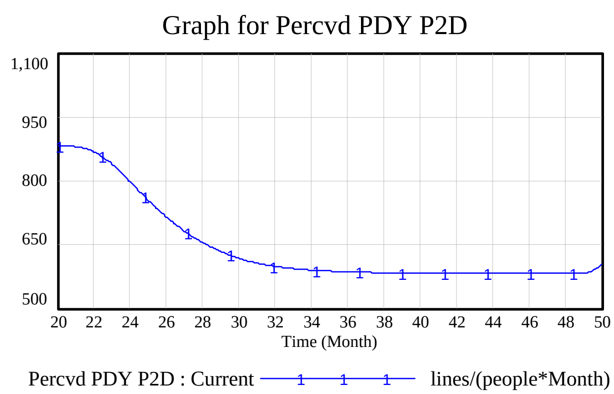


Figure 52 - Case #4 Productivity (Complex Projects)

The decreased productivity may not be initially perceptible, but it can add up over time. One of the things that helps make Case #4 not as bad as it could be in terms of overall effort is the fact that there are some balancing loops that aid in retaining experienced workers. The model assumes that complex projects are more desirable (leading to less attrition) and also that workers learn faster (leading to more experienced staff and higher overall intellectual capital[24]).

[24] Intellectual Capital in the model is defined as $\Sigma(\text{Novice}_i * \text{Novice Skill}) + \Sigma(\text{Intermediate}_i * \text{Intermediate Skill}) + \Sigma(\text{Expert}_i * \text{Expert Skill})$.

On-Time Completion of Projects

Comparing Case #2 and Case #6 shows how minor rearrangement of planned project starting times can result in less schedule slips. Both cases contain four projects of the same magnitude and duration. However, Case #6 defers the starting times of Project 2, Project 3, and Project 4 by 3, 6, and 9 months respectively. Figure 53 and Figure 54 show the Work to Do profiles as well as completions times for Project 4, the final project in each portfolio.

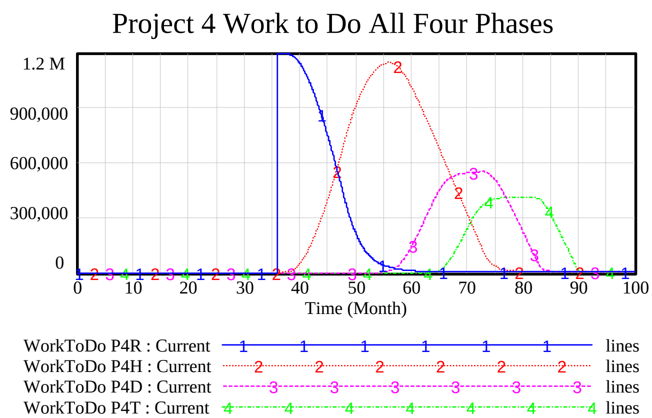


Figure 53 - Case #2 Work to Do (Base - Imperfect Quality)

Both cases exhibit as very similar Work to Do profile despite the fact the projects in Case #6 were planned to start later. However, the more important observation is that in Case #6 a latter start for Project #4 still results in completion around the 90 month mark. This results from less contention for resources with other projects. A small schedule shift allowed for Project #4 to actually be completed in less overall time and also to the planned schedule.

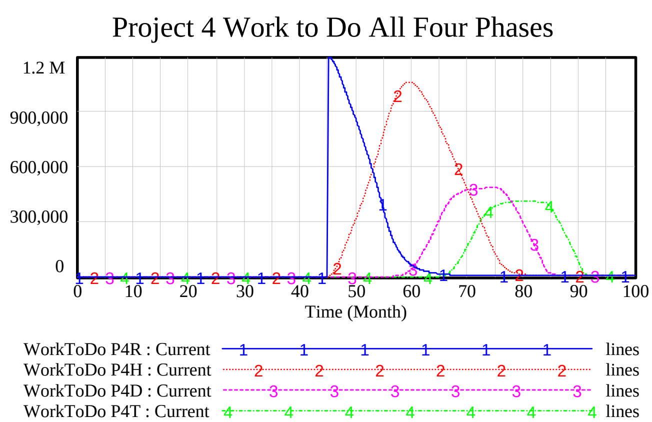


Figure 54 - Case #6 Work to Do (Inter-Project Gaps)

Figure 55 and Figure 56 show the duration time for all the projects within the portfolios of Case #2 and Case #6. Again, while the absolute calendar dates for Case #6 may not be any better, all the projects finish much closer to the planned completion times, something customers and employees are usually happy to see.

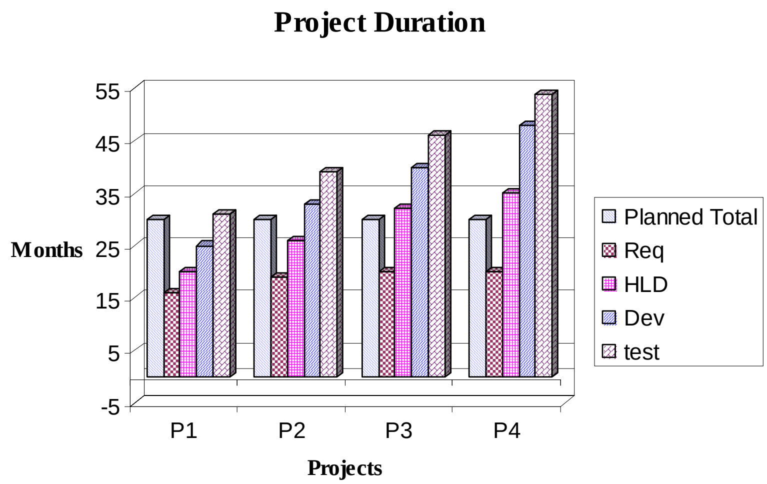


Figure 55 - Case #2 Project Duration Times (Base - Imperfect Quality)

Figure 55 shows how Project 2, Project 3, and Project 4 exceed the allotted time for completion by several more months than in Case #6. For instance, Project 4 takes almost 55 months to complete the Test phase even though it was originally scheduled to be done after 30 months. In contrast, Project 4 in Case #6 completes after only 45 months, 10 months earlier than in Case #2.

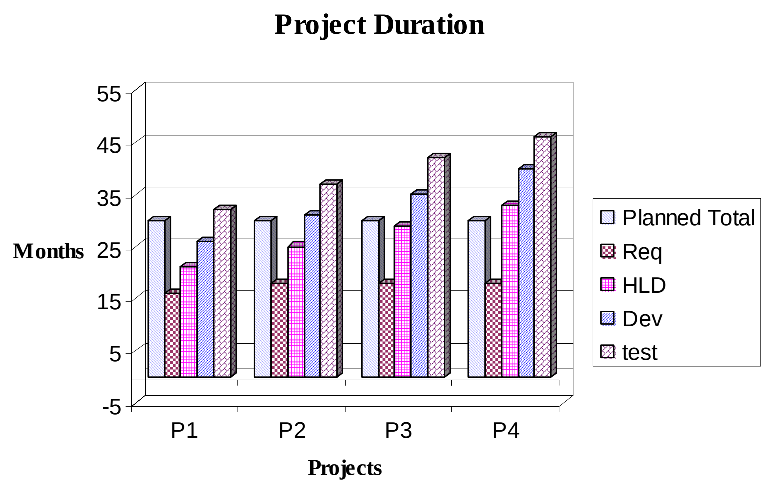


Figure 56 - Case #6 Project Duration Times (Inter-Project Gaps)

An added bonus of Case #6 is that the overall effort expended is also less than that in Case #2. Figure 57 and Figure 58 show how inserting a few inter-projects gaps in a schedule can relieve contention for resources and later reduce overall requirements for portfolio resources.

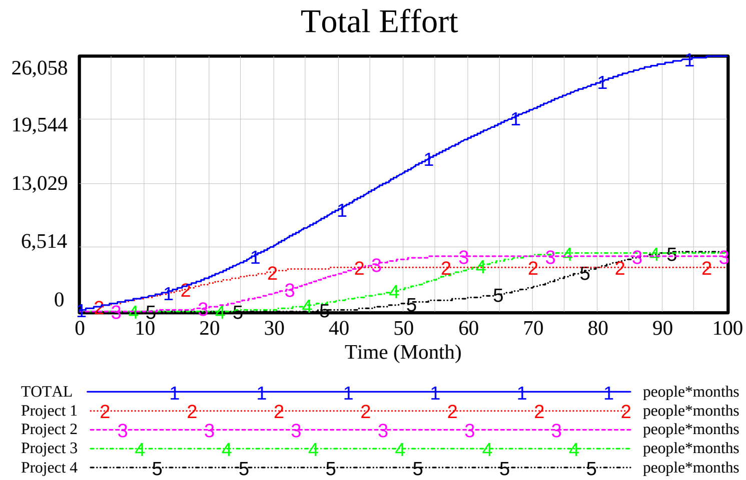


Figure 57 - Case #2 Total Effort (Base - Imperfect Quality)

While Case #2 was planned for earlier completion, all the projects end up finishing around the same time as in Case #6 but utilizing extra resources. This counter-intuitive result, scheduling later to finish sooner, is enabled by less resource contention between overlapping projects within the portfolio.

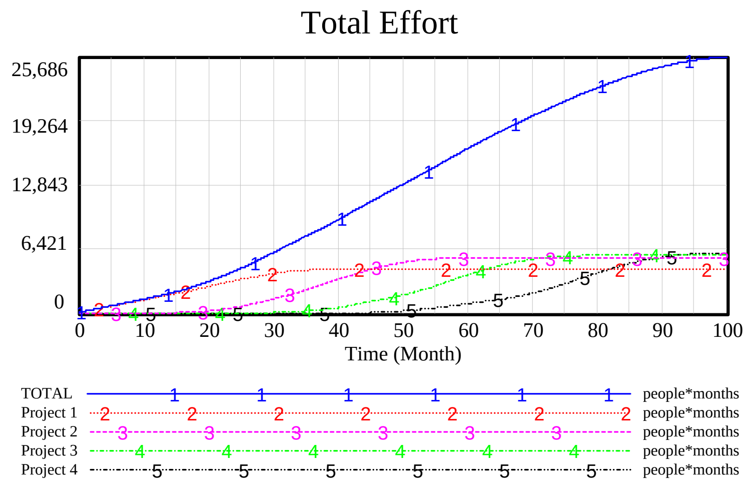


Figure 58 - Case #6 Total Effort (Inter-Project Gaps)

Attrition

Differences in internal portfolio dynamics result in driving varying rates of attrition within projects. Case #2 and Case #8 show substantially different attrition during the 25 month to 55 month time period. Both portfolios must deal with competition for resources. However, Case #8 exhibits much less dynamic competition for resources between projects due to the fact that its Project 3 is a much higher priority. In fact, Project 3 is at such a high priority that it pulls from other ongoing projects almost all the resources it requires to complete on time. This intense focus on Project 3 in Case #8 creates contributes to an environment with less overall fatigue and attrition. Thus, as shown in Figure 59 and Figure 60, during the time period corresponding to Project 3, Case #8 looses around 60 less Novice developers to attrition than does Case #2.

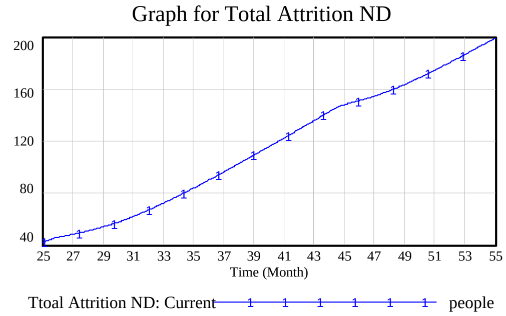


Figure 59 - Case #2 Attrition (Base - Imperfect Quality)

The starting attrition level is about the same at month 25. However, the attrition level in Case #2 grows much faster. Project 3 in Case #8 has high enough priority that it is able to pull resources away from the still incomplete Project 1 and Project 2. This results in slowing attrition since workers that end up on Project 3 have to work less overtime and thus experience less fatigue and rework. The unfortunate consequence is that while Project 3 finishes on schedule, Project 1 and Project 2 finish late. Both Project 1 and Project 2 get starved for resources and essentially sit in a stalled state until Project 3 finished and releases resources.

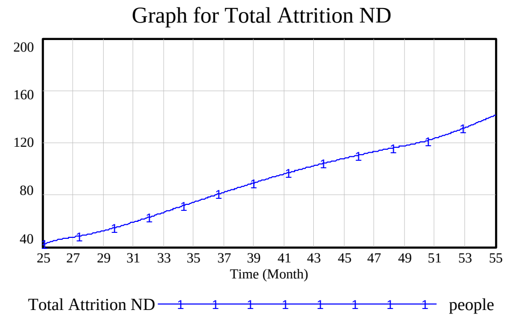


Figure 60 - Case #8 Attrition (Project 3 as High Priority)

Figure 61 illustrates the extent of how late Project 3 runs in Case #2. At the 55-month mark, Project 3 is just barely finishing the requirements phase. Both the development phase and test phase will take several additional months.

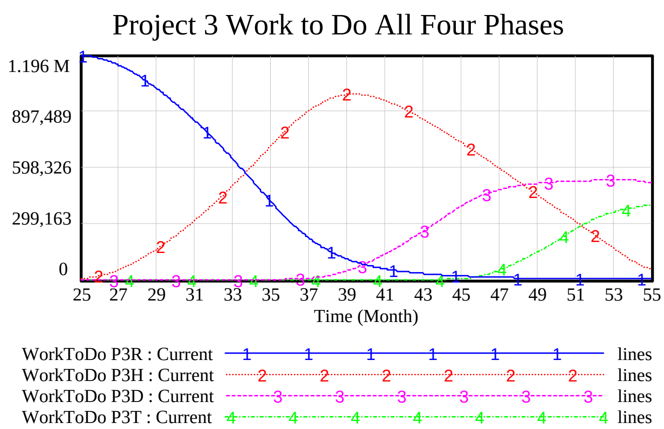


Figure 61 - Case #2 Work to Do (Base - Imperfect Quality)

When Project 3 is given a high relative priority, as in Figure 62, it is able to complete all its phases much sooner. Without needing to deal with resources going back and forth between competing projects, workers on Project 3 can be much more focused and productive. A priority decision that ends up being very advantageous for Project 3 results in potential disaster for Project 1 and Project 2. Before assigning such a high priority to Project 3, the portfolio manager must be consider the crippling effect the decision could have on Project 1 and Project 2. In some situations it could be better to take an action such as canceling Project 2 rather than have it be so late and possibly miss a market window of opportunity.

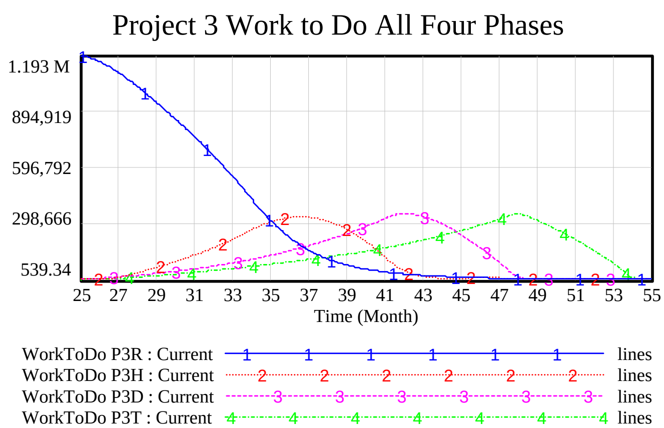


Figure 62 - Case #8 Work to Do (Project 3 as High Priority)

Progression of Workers between Skill Levels

The ability to focus on a project and not deal with issues such as fatigue and multi-tasking allows workers to more readily learn and progress to more advanced skill levels. During the 25 month to 50 month time period of Case #2, the total intellectual capital level of the project portfolio remains fairly constant as shown in Figure 63. In fact, while the total level remains about the same, the level of Intermediates and Experts slightly declines. The decline is only offset by the hiring of Novices.

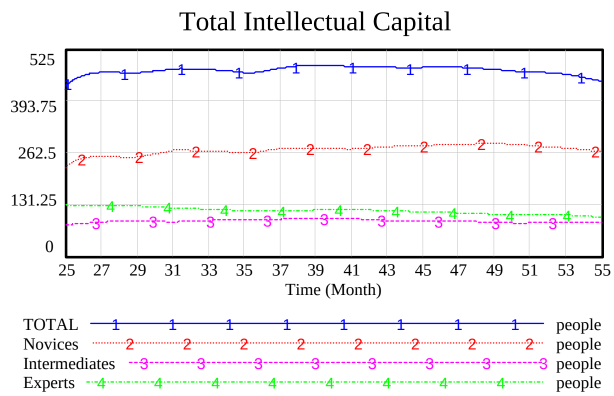


Figure 63 - Case #2 Intellectual Capital (Base - Imperfect Quality)

In contrast to Case #2, Case #8 experiences an increase in overall intellectual capital during the corresponding time period as illustrated in Figure 64. It does so by increasing the number of Experts and Intermediates even though the number of Novices actually declines. The primary driver for increasing intellectual capital during this time period is the increased focus and decreased fatigue that the workers experience. This results from Project 3 enjoying a higher relative priority. This ultra-high priority for a single project creates an working environment with intense focus.

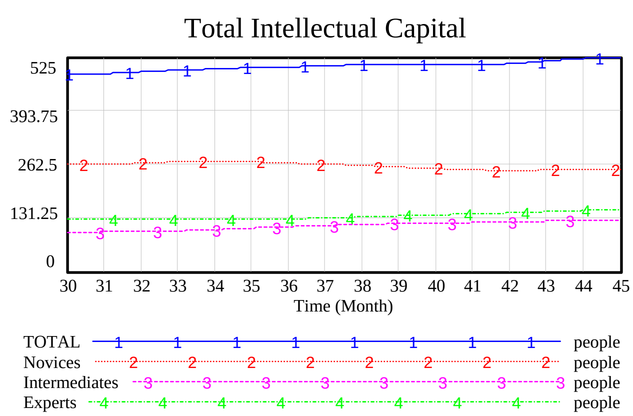


Figure 64 - Case #8 Intellectual Capital (Project 3 as High Priority)

Intellectual Capital Growth

While the overall intellectual capital level[25] of an organization can be increased by hiring additional employees that is often not an option. Frequently, projects are already operating at or near the maximum headcount. In addition, adding employees from outside the organization can end up decreasing productivity over the short-term due to the increase in mentoring from the existing worker base.

Another mechanism for adding to the overall intellectual capital base in the organization is to grow the skill level of the existing employees. However, this can be challenging in many environments where workers must multi-task and deal with being pulled between various assignments on different project. Figure 65 and Figure 66 show the growth of both intellectual capital and total workers in Case #8.

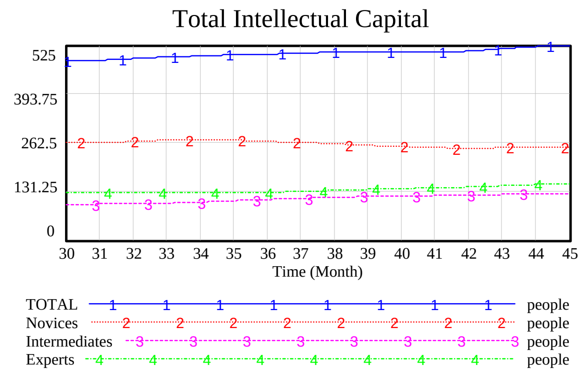


Figure 65 - Case #8 Intellectual Capital (Project 3 as High Priority)

Due to increased worker focus during the Project 3 time period, Case #8 is able to improve the intellectual capital base even though the number of total employees is relatively flat.

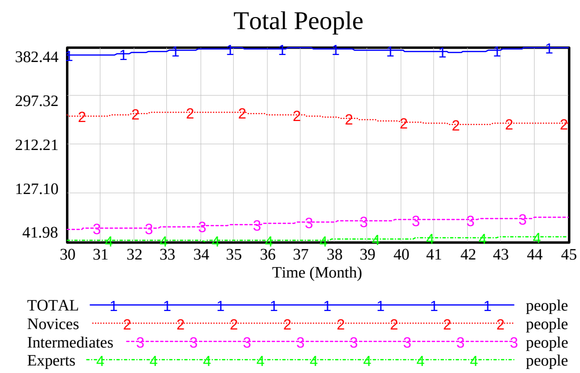


Figure 66 - Case #8 Total People (Project 3 as High Priority)

[25] Intellectual Capital in the model is defined as $\Sigma(\text{Novice}_i * \text{Novice Skill}) + \Sigma(\text{Intermediate}_i * \text{Intermediate Skill}) + \Sigma(\text{Expert}_i * \text{Expert Skill})$.

Summary

Differences in Perfect and Imperfect Quality

Rework within a portfolio can drive undesirable things such as overtime, fatigue, and attrition. This can be a vicious cycle. Projects that avoid rework will contribute positively to a better overall portfolio.

Loss of Productivity

A relatively small difference in individual productivity can add up to much more successfully completed work over the long run. Individual projects within the portfolio must be cautious about reducing individual worker productivity.

On-Time Completion of Projects

Although it is counter-intuitive, planning projects to start later may actually aid in allowing the overall project portfolio to finish sooner. In addition, changing the scheduled start dates may actually reduce the overall effort required to complete all projects within the portfolio.

Attrition

Reducing attrition may sometimes require sacrificing other portfolio parameters. Finishing a project later than expected may be the price of retaining workers.

Progression of Workers between Skill Levels and Intellectual Capital Growth

Intellectual capital can be increased in an organization without hiring additional workers. However, an environment must be maintained that fosters a progression in employee skill level.

The cases represented in Appendix A cover of range of possible project portfolio situations. Table 5 categorizes the cases such that they can be utilized as a basis for organizational policy formation.

Category	Case Number
Project Focus	1, 2, 4, 10, 11, 13, 14, 15, 16
Resource Allocation	6, 7, 9, 11, 12
Project Scheduling	3, 6, 9
Project Priority	3, 4, 8, 13, 14, 15, 16, 17
Portfolio Balance	4, 5

Table 5 - Policy Categories

(This page intentionally left blank for duplex printing.)

Chapter 7: Recommendations

Ever increasing competitive pressures within the aerospace and document imaging industries have led to attempts at organization renewal at the Sikorsky Aircraft and Xerox corporations. The intention of this renewal through organizational structure change is to facilitate workflow with knowledge sharing and knowledge retention while maintaining an acute focus on the customer. The product development processes that accommodate these structures are complex with interdependent elements. The allocation of scarce resources is a key source of this interdependence. A system dynamics multi-project, multi-phase model that reflects the current processes at both Sikorsky Aircraft and Xerox was developed and tested in order to provide insight into the dynamics of the development process and influence management mental models. Policy recommendations that lead to well-managed resource allocation processes are presented along with the implications for the effective management of knowledge. The policies are summarized in Table 6. Complete development of each policy follows.

Policy	Recommendation	Time Horizon
Overtime	Limit overtime to relatively short periods and only to recover from truly unforeseen circumstances.	Near-Term
Project Priorities	Keep project priorities relatively static. Adjust priorities only after impact assessment of entire portfolio.	Near-Term
Resource Allocation – The Beginning	Accurately account for project staffing ramp in formulating project completion date. Do not utilize simplified level-load assumptions for more than rough scoping. Update resource requirements as new information regarding all projects is obtained.	Intermediate
Resource Allocation – The End	Delay transition of workers from current project until some period after last milestone is accomplished.	Intermediate
Project Cancellations	Increase frequency of cancelled projects early in product lifecycle. Remove any future projects from the planning portfolio that have little chance of either starting on time or completing before marketplace opportunities have evaporated.	Intermediate
Portfolio Balance	Assess attractiveness of proposed projects based on a combination of contribution to intellectual capital growth as well as traditional monetary return on investment.	Long-Term
Inter-Project Gap	Schedule projects with buffer gap between the ending of one and the beginning of the next.	Long-Term

Table 6 - Recommended Policies

Overtime

One of the most easily implemented plans for recovery after things have not gone as planned is to instigate overtime for an organization. The assumption is normally that it can be implemented very rapidly and that the cost to the organization in terms of dollars is much less than what it would cost to acquire and train more workers. Unfortunately, this seemingly straightforward policy does not scale up very well. Unlike many machines, double the hours for a human worker does not necessarily mean double the productivity. In addition, long durations of overtime can lead to other undesirable affects such as increased attrition, decreased work quality, and degraded morale.

Typical Current Policy

Utilize overtime to greatest extent possible when behind schedule.

Proposed Policy

Limit overtime to relatively short periods and only to recover from truly unforeseen circumstances.

Traditional Logic

- The project will only require overtime for a short time – just until things are back on track.
- Fifty percent more work hours means fifty percent more work accomplished.
- The workers will understand that overtime is necessary.

Flaws in Traditional Logic

- Overtime situations are a slippery slope. Some overtime usually leads to more overtime as it becomes the status quo.
- Extended overtime leads to things like increased fatigue, increased attrition, and decreased work quality, all of which result in less effective work accomplished.
- In situations of paid overtime, many employees will welcome the increased income for a short while without thinking about whose fault it is. In environments of unpaid overtime, most employees will soon figure out that their personal time should not be spent making up for the poor planning or foresight of management.

Impact on Intellectual Capital

- No time allocated for learning via formal mechanisms like classes or training programs.
- Very little if any time allocated for mentoring from more experienced employees.
- Increased fatigues leads to higher attrition rates of experienced workforce.
- Little time spent reflecting on lessons learned from previous completed work.
- Rushing leads to Band-Aid type fixes rather than development of new concepts.

Discussion

Frequently projects that are behind lead to situations where overtime is instituted. Some environments allow for paid overtime while others do not. In addition, some companies only pay overtime to employees up to a certain grade level. After that it is assumed employees will work effectively due to a sense of professionalism and dedication to the company. Even worse, employees can be put in situations where management blames the schedule slippage on the workers themselves rather than on poor planning assumptions or procedures. In these situations, employees are often lead to believe that they “owe it to the company” to work nights and weekends away from their families in order to get back on schedule. While there are clearly examples of employees more freely working longer durations of overtime for little or no immediate compensation (dot-com’s being an example), these cases are more the exception than the rule. Also, it is not clear that dot-com employees are working overtime for free or simply for a chance at a huge stock option payoff that allows them to retire 20 years early.

Extended periods of overtime can rapidly become less effective than originally planned by management. There are numerous examples of employees taking long lunches or leaving early on Friday since they know they have to be in for mandatory overtime on Saturday anyway. Worse yet are the instances of an entire workgroup showing up to work overtime on a weekend even though ten percent of the people were actually required. The majority of the group ends up killing time by playing Microsoft Windows Solitaire at their desk while the “critical few” are actually working. The result is that all the employees loose valuable personal and family time while at the same time the “critical few” build up resentment for those that are not working as hard.

While there are situations where a few days of overtime make sense to recover from a point problem or unforeseen short-term setback, managers should beware of falling into an overtime trap that can drastically reduce the overall effectiveness of the organization.

One of the deceptive flaws with extended overtime is that no single impact may be immediately perceivable by either employees or management. However, the longer-term cumulative impact may be huge. Figure 67 and Figure 68 illustrate how only a relatively small change in some attributes can lead to significantly degraded overall performance.

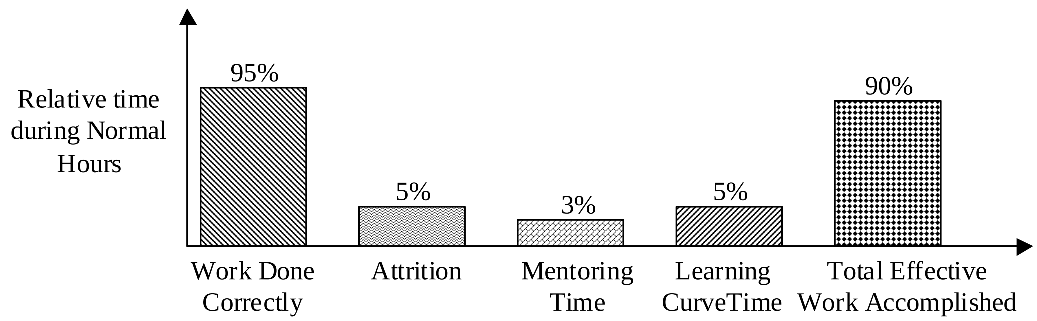


Figure 67 - Overtime Effects A

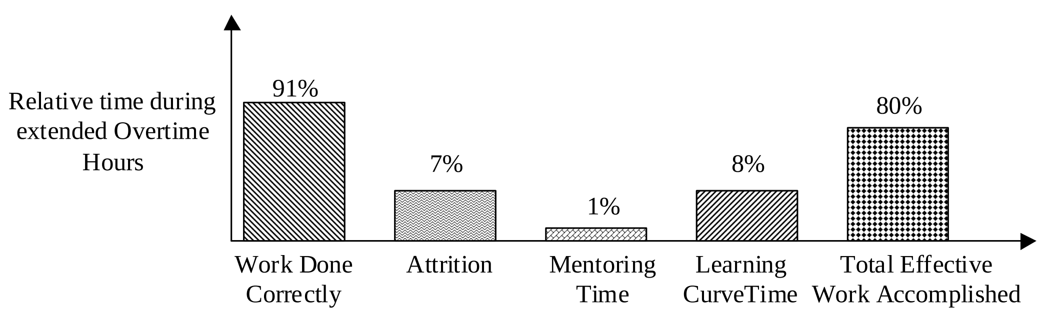


Figure 68 - Overtime Effects B

In an extended period of overtime, degraded work quality and effectiveness can actually lead to schedule slippage rather than schedule recovery. Figure 69 and Figure 70 illustrate the difference between the desired results and the typical actual results of overtime. While management's perception initially may be that the effect of overtime is having the desired impact, this does not hold true over the longer term. Fatigue, attrition, and less careful workmanship can all lead towards a significant decrease in overall effectiveness per hour of work. What seemed like a good recovery policy may actually have the reverse impact relative to what was intended.

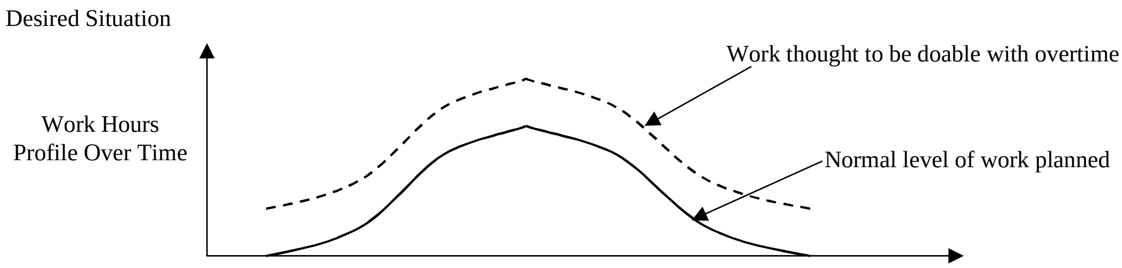


Figure 69 - Desired Overtime Profile

While the effect of overtime in the short-term is usually a near linear increase in productivity, the long-term result can be much more disappointing. The impact of fatigue, attrition, and decreased work quality can take their toll on the effectiveness of the overall organization.

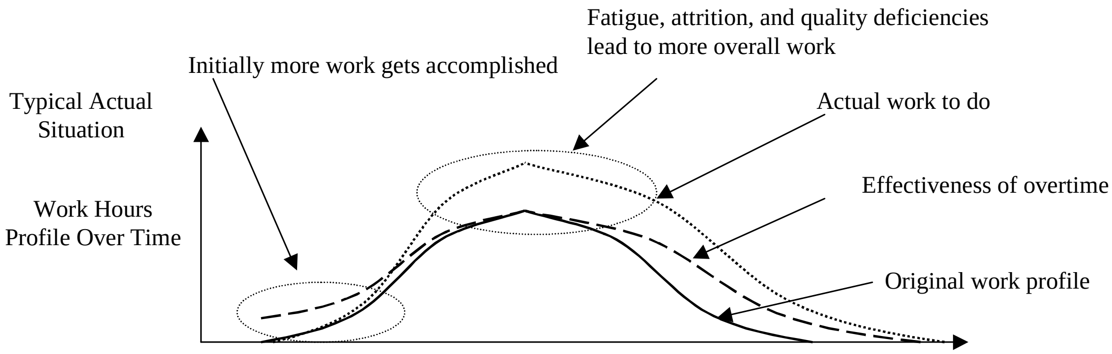


Figure 70 - Typical Overtime Profile

Barriers to Implementation

- Managers often see quick results when initially implementing overtime. It is easy to assume that these initial benefits can be extended indefinitely into the future without negatively impacting the organization.
- Even for managers that realize there may be a longer-term negative effect on the organization, it is easy to assume that their particular project will no longer need overtime work by the time any negative effect sets in.
- At least in pay-for-overtime environments, employees will generally respond positively to the policy initially. It can be viewed by workers as a quick route to a virtual pay increase.
- Managers may not be fully aware to what extent employees can "game" the system either by consciously or unconsciously decreasing work effectiveness. This includes the conscious phenomenon like workers playing Microsoft Windows Solitaire since they were told the

team had to work overtime on a Saturday, but individually they had nothing immediate to do. It also includes the possibly unconscious acts of employees ending their day a little early on Friday so that they can get a few personal things done before the mandatory overtime Saturday work begins.

Resource Allocation – The Beginning

In most development organizations, resources often seem to be too few, especially people resources. This lack of supply creates an environment where projects can compete over who works where. Ongoing projects usually have more clout than projects that are just getting started. Ongoing or current projects benefit from the mental impact of the immediacy of release. The fact that a current project will soon be in a customer's hands is normally a much more compelling argument for acquiring or retaining resources than that of a project that is barely off the drawing board. In addition, managers may often believe that projects that are just starting will somehow have time to catch up later. There seems to be eternal optimism that future projects will somehow have better luck than projects of the past. This optimism persists in spite of the fact that no portfolio management policies have explicitly been modified nor mental models changed since the last project had a difficult time acquiring initial resources to the planned level.

Typical Current Policy

Begin staffing cycle concurrent with project start.

Proposed Policy

- Accurately account for project staffing ramps in formulating project completion date. Do not utilize simplified level-load assumptions for more than rough scoping.
- Update resource requirements as new information regarding all projects is obtained.

Traditional Logic

- We'll magically get all the people we need on the exact day we need them since senior management told us that our project was a "priority".
- There shouldn't be much of a learning curve since people will be applying essentially the same skills as required in their previous assignment.
- It's alright if we start the project a few people short. Our processes are more refined since we started the last project. What we lack in people we'll make up for in process effectiveness.
- It will be perceived that we're wasting money if we staff a project too early.

Flaws in Traditional Logic

- Other projects that are in the pipeline now are likely to be more important as time goes on since they will be "close to finishing".
- It still takes time for employees to adjust to a new team, new project context, new procedures, new management expectations, etc.
- No amount of process will make up for unfulfilled assumptions. There is no substitute for getting people started early and avoiding crisis situations later in the project lifecycle where overtime, attrition, fatigue become much more significant factors.
- Look at the overall end-to-end project cost rather than the monthly burn rate. A project that cost slightly less per month but runs for twice as many months will be much more costly in terms of dollars as well as impact to follow-on projects in the portfolio.

Impact on Intellectual Capital

- No learning curve time planned to bring employees up to speed on latest technology or processes that will be utilized on new project. Workers are expected to "pick it up as they go".
- Starting people late on a project will increase overall fatigue and decrease time spent on activities such as mentoring.

Discussion

Figure 71 illustrates the typical simplified planning assumption that gets used. A manager takes the total estimated amount of work and then divides it by the amount of time allocated (i.e. the time between "planned start" and "promised delivery") in order to calculate a quick estimate of the number of workers required. One of the simplifying assumptions, all people being available day one, ends up significantly distorting the plan and ultimately impacting the probability of success for the project. Utilizing a Beta distribution of resources is also a distortion of reality. While the Beta curve accounts for the slower than immediate staffing of a project, senior management is normally reluctant to actually approve staffing beyond the level that was presupposed with the simplified level-staffing assumption.

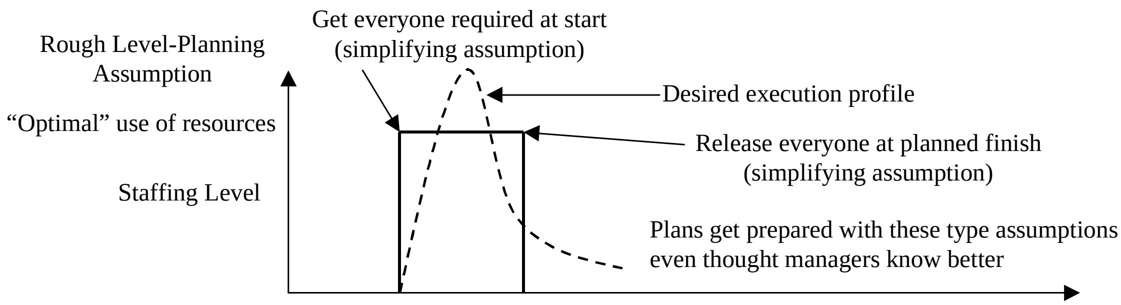


Figure 71 - Planned Use of Resources

The actual results are typically similar to those represented in Figure 72. Workers end up slowly rolling onto a project – even more slowly than the Beta assumption. Since the original planning assumptions were not met, the result is that the eventual outcome deviates significantly from that desired. The slower than planned staffing profile can lead to situations such as schedule slips that further result in things such as overtime. In addition, when the lifecycle for a project is longer there exists a greater chance for multiple change requests that result in rework conditions.

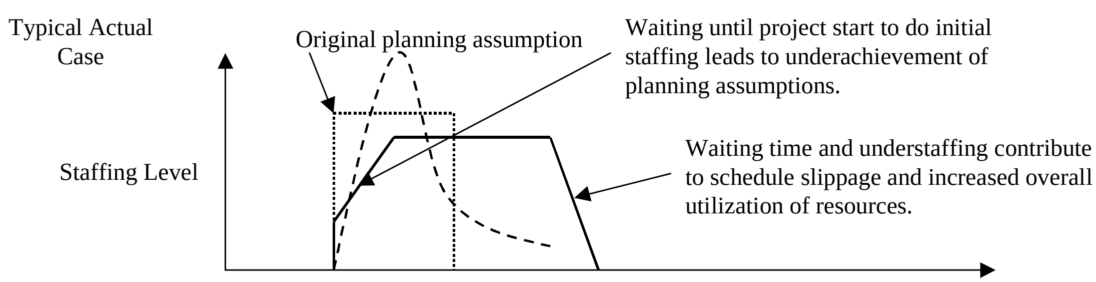


Figure 72 - Actual Use of Resources

Figure 73 shows the impact of utilizing the proposed policy accounting for the staffing ramps for calculating the duration of a project. While the cumulative staffing hours are greater than that of the "optimal" case depicted in Figure 71 the number of hours is significantly less than required in the typical case shown in Figure 72.

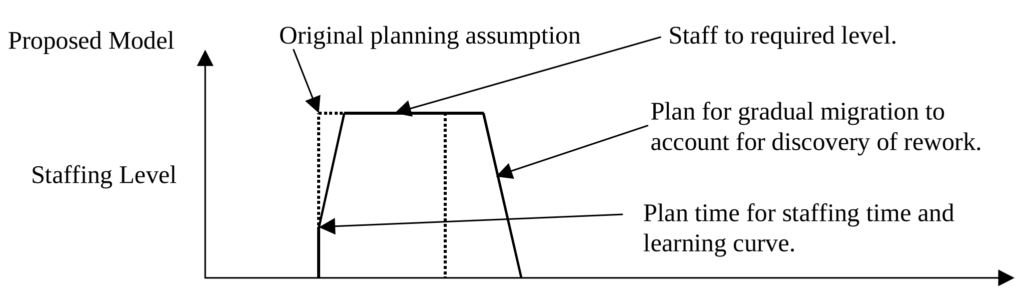


Figure 73 - Proposed Use of Resources

While the consequences of staffing a project insufficiently should be clear to most managers, the situations still frequently occurs. One example is that of a small “one year” project, called Project X, that lasted six years before finally being cancelled (Actually worse. No longer required in the marketplace). Project X was part of a much larger portfolio of projects. While the remainder of the projects in the organization’s portfolio were all of the fifty to one hundred person variety, this project would only require eight “good people”.

At the start of each year Project X would get staffed with only two people. The understanding was that as soon as some of the more important (i.e. larger projects) got done, Project X would get fully staffed. However, there was always some crisis on one of the more important projects that prohibited people transitioning to Project X. Worse, since Project X only had two people assigned, it could barely keep up with changing requirements. Project X simply did not have enough critical mass to move forward with any discernable progress. At the end of every year the internal customers were always disappointed with the progress that had been made. In addition to the engineering effort that was continually expended, various other supporting organizations ended up having to dedicate staff to the effort. The end result was that after six years, millions of dollars in funding was wasted. Even more significant than the sunk funding loss is the fact that numerous people were essentially wasted on a project that had close to zero probability of ever finishing successfully. This lost opportunity and damage to moral far exceeds the measure of tangible dollar loss.

Barriers to Implementation

- Managers are often rewarded for the success of a single project and not necessarily for the performance of the overall portfolio.
- Senior managers do not often remain in positions long enough so that the performance of the overall project portfolio can be measured and later attributed to their choice of institutionalized policies.
- There persists a common mental model that issues and/or work on current projects must take precedence over work related to future projects. The thinking seems to be that in order to get to a point where the future project matters, the organization must divert all attention to current issues.

Resource Allocation – The End

Similar to the situation in which resources are too few at project start, resources are frequently too few during the end of a project. Once a program "appears" to be done, there is normally an urgency to efficiently utilize the remaining workers somewhere else. The problem is that sometimes projects that "appear" to be complete, based on some common understanding of "done" or previously derived completion criteria, are not really at the point at which resources should be totally removed. There are cases where undiscovered rework may take from several weeks to even months or years until requiring development attention. However, it is in these cases that a manager, even knowing rework will eventually be required, has difficulty in holding resources that another project claims to be able to fully utilize. The demonstration of tangible "need" typically outweighs any argument for what "might" happen. Unfortunately this myopic assessment of resource allocation does not always contribute in a positive fashion to the goals of the overall project portfolio.

This policy should be distinguished from that of Inter-Project Gap in the following section. While on the surface both policies relate to managing the transition of resources between the end of one project and the start of the next, the Inter-Project Gap policy should be viewed from a more static long-term planning perspective. This policy, Resource Allocation – The End, should be viewed as a more real-time policy that deals with resource allocation once a project appears to be finished.

Typical Current Policy

Start transitioning workers off of current project as last significant milestone is met.

Proposed Policy

Delay transition of workers from current project until some period after last milestone is accomplished.

Traditional Logic

- Need to start transitioning people to next project as soon as current project shows signs of being "done".
- Maximum efficiency is obtained by keeping people "busy" and "working".
- The faster we start the next project the sooner it will be done.

- We can maintain momentum by getting people started on the next project right away.
- If the previous project requires any rework our people can always “multi-task”.

Flaws in Traditional Logic

- Busy is not equivalent to productive especially when considering overall product portfolio.
- The next project will only be finished sooner if it can be assigned dedicated resources. However, problems on any preceding projects usually become the highest priority since the product may be in customer hands.
- People will naturally loose momentum during transitions. A break is required for people to recharge after the normally intense period of product launch.
- Too much multi-tasking leads to tremendous inefficiency.

Impact on Intellectual Capital

- Firefighting situations lead to loss of time for learning and skill enhancement.
- People pulled back into previous project have more incentive to “Band-Aid” or kludge repairs rather than fix things properly. Workers tend to be under significant management pressure to fix things quickly.
- Employees end up with little or no time to reflect upon lessons learned from previous project so that they may be applied to future endeavors.
- Excessive multi-tasking can lead to poor moral.

Discussion

Figure 74 illustrates how the remaining work on a project, under perfect operating conditions, progresses to zero. Under this set of assumptions, the completion date for a Project A as well as the starting date for Project B can be perfectly synchronized. However, it is unusual for projects to proceed so smoothly.

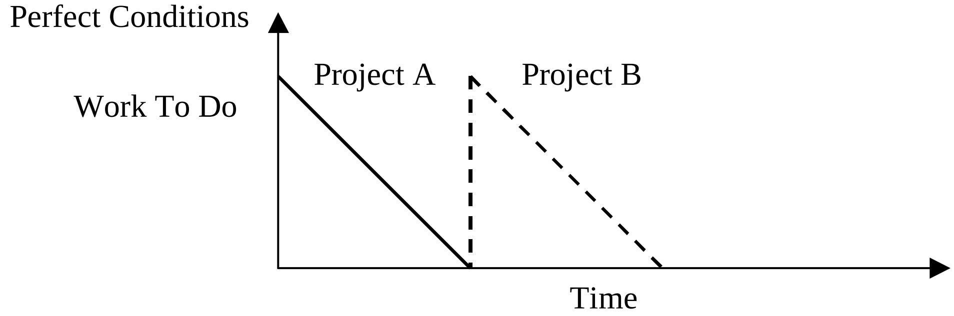


Figure 74 - Work To Do Under Perfect Conditions

A more typical scenario is depicted in Figure 75. Even though the perception is that Project A is finished, some remaining undiscovered rework may still remain. However, rather than postpone the roll-off of personnel from Project A until all potential rework is complete, people are immediately transitioned to Project B. Once issues are discovered for Project A, it takes some amount of time for the appropriate workers to be re-assigned from Project B. By the time this happens, Project A has a large backlog of issues and has gone into "crisis" mode. Since Project A has now gone into crisis mode, it is likely that people are working overtime and are fatigued. This combination of factors of Project A creates an environment in which problems may actually take longer to fix than under ideal conditions. Concurrent with the crisis on Project A, Project B has now potentially stalled due to lack of focus and/or critical mass. In addition, once people finish resolving issues with Project A, they now have to go through another transitions cycle and short re-learning curve for Project B.

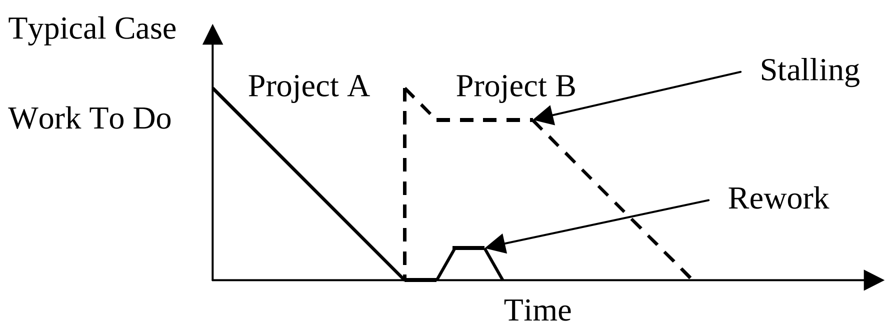


Figure 75 - Typical Case for Work To Do

The proposed model is represented in Figure 76. In this situation, the actual transition of workers from Project A to Project B is deferred in order to allow for potential rework. Since the people are still assigned to Project A, any rework issues should be able to be handled much more efficiently than could be done by pulling personnel from Project B and instigating firefighting. In the event that no rework issues occur on Project A, this time period can be utilized for training, rejuvenation, and reflecting on lessons learned from Project A. The benefits from keeping workers assigned to Project A are at least two-fold. First, providing adequate time to document lessons learned from the project can be very beneficial to future projects that are able to leverage improved processes and not repeat mistakes that may have been made in the past. Second, while workers on spending time capturing lessons learned, they are keep effectively "warm", ready to efficiently address any critical rework issues. Also, Project B and successive projects are completed in a more timely manner than with the typical case represented in the Figure 75. The fact that there is no firefighting between projects allows workers to focus on Project A followed by focusing on Project B. This increased focus can lead to more efficient execution of tasks required to complete Project B in a timely manner.

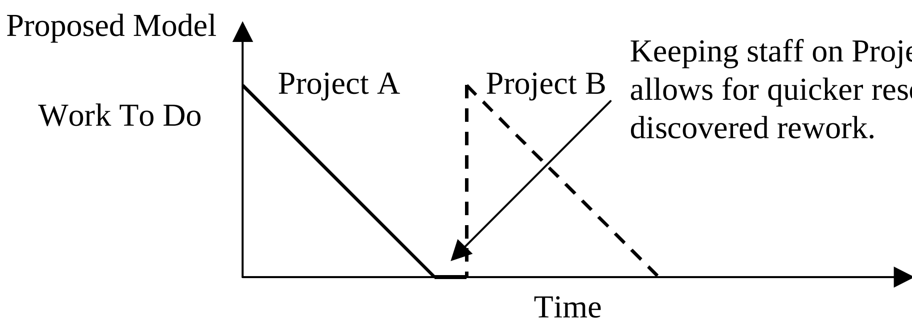


Figure 76 - Proposed Model for Work To Do

While management often considers multi-tasking a necessary skill for any competent worker in the development environment, the number of tasks that can be concurrently handled efficiently is often exaggerated. Managers would like to believe that their employees could effectively handle a dozen current tasks. However, the real number of tasks that can be efficiently handled is really only two to four before effectiveness falls rapidly[26]. Figure 77 illustrates how rapidly individual efficiency deteriorates as the number of assigned concurrent tasks increase. An infinite level of multi-tasking would allow workers to achieve maximum efficiency. Any time gaps spent waiting on an external dependency could be spent working on an additional assignment. However,

[26] Lucas, William A., et al. "The Wrong Kind of Lean: Over-Commitment and Under-represented Skills on Technology Teams", The LeanTEC Project, Sloan School of Management Massachusetts Institute of Technology, May 2000., p. 21.

context switching is much more difficult for humans than machines. Workers have a much more difficult time separating and compartmentalizing data than do computers. The price of this difficulty is paid in decreased worker efficiency. During scenarios such as firefighting between two projects, workers can be put into situations of this decreased efficiency. While it's easy for managers to only think of the two or three primary assignments that employees might have, there are usually numerous secondary assignments that need to be considered.

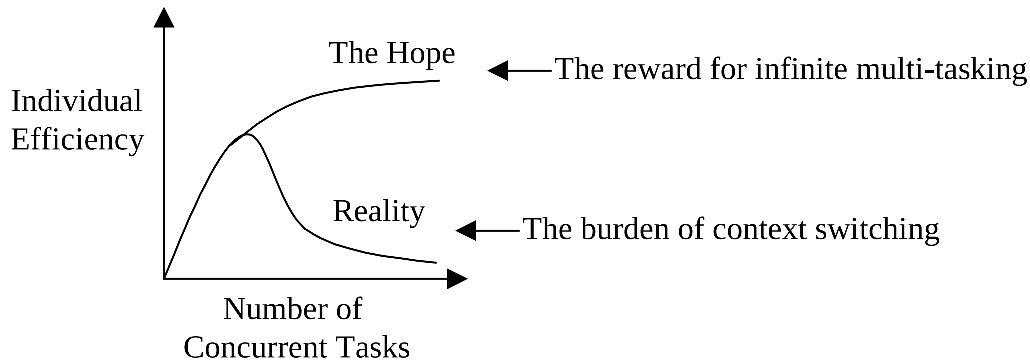


Figure 77 - Multi-Tasking Efficiency

An example of how detrimental the premature transition of workers can be to an organization is illustrated by the following story of Project X and Project Y. Project X consisted of 50 workers and was scheduled to complete before the start of Project Y which was planned to require 150 people. Due to typical project issues such as changes in scope of major design changes, Project X slipped its end date by a few months and thus the start of Project Y was deferred by the corresponding amount of time. However, now that Project Y slipped its start date, management was more eager than ever to immediately transition people from Project X when it reached its final implementation milestone. Not surprisingly, most believed that Project Y, with all the attention it was now receiving would quickly make up any lost time. However, the feelings of success for Project Y were short lived. Only a few months after Project Y started forward with its 150 people, issues with Project X became apparent. In fact, there were enough unforeseen issues with Project X that 25 people had to be taken off of Project Y in order to recycle through the High-Level Design and Development phases of Project X. While 25 people out of the 150 assigned to Project Y may not seem significant, these were 25 of the most critical people required to successfully begin Project Y. Since Project Y was based largely on work done previously on Project X, the plan had always been to rely on these 25 key individuals. Removing these few key individuals from Project Y caused the program to initially stall for several months, essentially incapacitating the other 125 workers on the project. The end result was that Project Y ended up slipping its schedule by over a year. The impact to the organization was millions of dollars worth of lost productivity along with immeasurable impact on the lost opportunities. In addition, at least 25 workers were placed in a no-win situation of high stress for several months.

Barriers to Implementation

- Managers feel comfort in seeing people occupied with a task that has directly observable results. The typical mental model is that we must be marching closer towards the organizational "goal" if everyone is "busy".
- People frequently forget about counter-productive or unsynchronized work that can both degrade organizational performance.
- Managers seem attracted by the pervasive "the early bird gets the worm" mental model, especially now in a fast-paced competitive market. There is a natural reluctance to trust in the words of Jay Forrester – "Don't just do something, stand there".[27]
- The typical managerial decision processes tend to value tangible and quantifiable things such as schedule slip and error rate over such "soft" factors such as rejuvenation time, lessons learned reflection period, and worker morale level.
- There is an under-appreciation for how multitasking and context switching can tremendously degrade the effectiveness of individuals.

[27] Keough, Mark, and Doman, Andrew, "The CEO as organization designer". The McKinsey Quarterly Number 2, 1992., p. 6.

Inter-Project Gap

In contrast with resource roll-on and roll-off policies that are more real-time in nature, this policy regarding the inter-project time gap should be viewed and exercised from the perspective of long-term planning. Organizations are keenly aware of how formalized processes and procedures, once institutionalized, can aid in overall development efficiency. However, it is too often assumed that an organization with institutionalized processes can operate almost perfectly with very little if any flaws in work. This leads management to believe there is marginal need in accounting for any rework during the planning process. Even if an organization has refined their internal planning procedures enough to accurately predict the amount of rework that will be necessary during a project, these procedures do not always account for the external dependencies such as subcontract work or deliverables from external suppliers. In either case, not including a buffer between projects for unforeseen issues can have a detrimental impact on future portfolio performance.

Typical Current Policy

Schedule projects essentially end to end with little to no buffer space in between.

Proposed Policy

Schedule projects with buffer gap between the ending of one and the beginning of the next.

Traditional Logic

- Place projects end to end without any buffer.
- Using a buffer is the sign of sloppy management.
- Planning for a project slip will only lead to a self-fulfilling prophecy.

Flaws in Traditional Logic

- Things will go wrong especially if this project did not start with all assumptions fulfilled.
- The longer the project, the more unforeseen obstacles. Normal projects initially get planned (from bottom up) with some buffer to account for uncertainty. However, as schedule is rolled up management usually likes to squeeze out buffers and assume more perfect operating conditions.

- Planning for a slip will only lead to self-fulfilling prophecy if work if left unchecked without any intermediate milestones that are rigorously monitored. It is important to have intermediate milestones that the projects teams drive towards and are measured upon.

Impact on Intellectual Capital

- No time to schedule things such as formal training or family vacation.
- Little or no time to reflect on lessons learned from previous projects.
- Longer-term impact of employee burnout.
- No time allocated for workers to research changes in technology since the beginning of their previously assigned project.

Discussion

Successive projects within a portfolio are often planned such that one project is assumed to ramp up at the same time another project is ramping down. There is infrequently any buffer space allocated between the planned projects. This is illustrated in Figure 78.

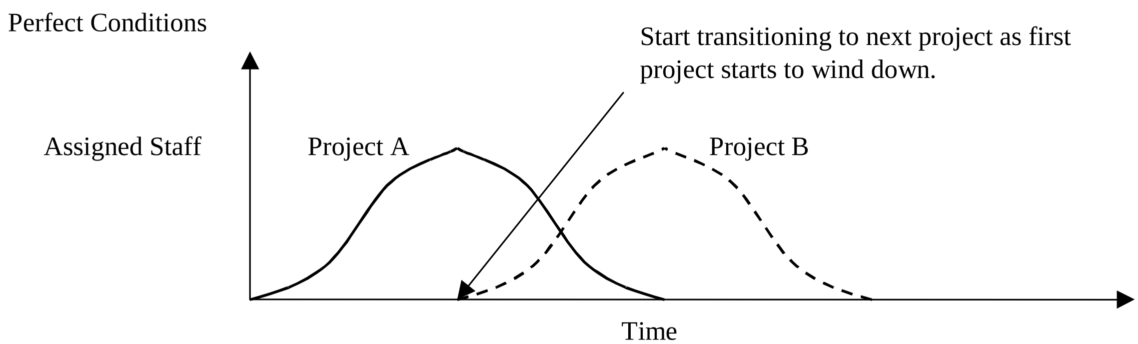


Figure 78 - Buffer Space Under Perfect Conditions

Unfortunately, the typical situation is that the previous program, Project A in Figure 79, slips its initial finish date. In this situation, Project A starts eroding into the planned time for Project B. Rather than simply slip Project B day for day contingent on Project A, there are normally internal political pressures as well as external marketplace pressures to at least partially staff Project B in hopes of accomplishing at least some advanced work. This creates an environment where not only is Project B far short of its assumed staffing level but also where a very real firefighting situation exists between the two projects.

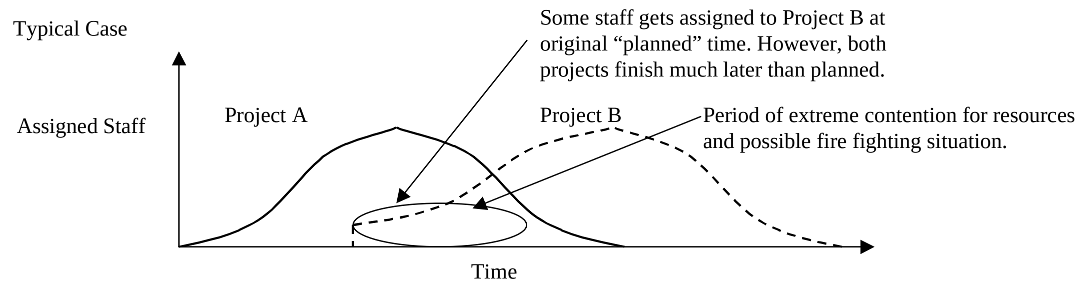


Figure 79 - Buffer Space Under Typical Conditions

In order to avoid situations such as firefighting and inadequate staffing, the proposed policy is to plan for some inter-project buffer. This is illustrated in Figure 80. While at first glance this may seem like planned under-utilization of resources, it is actually a mechanism to ensure smoother operation of the product portfolio when viewed as a whole.

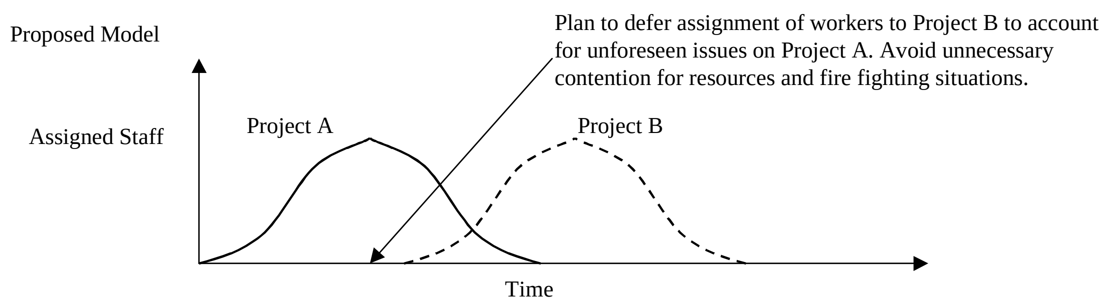


Figure 80 - Buffer Space Under Proposed Model

One of the difficult barriers to overcome with this proposed policy is the current mental model of many managers. As depicted in Figure 81, most managers would like to plan for a situation in which all onboard resources are always assigned to some project (assuming no organizational growth). This level staffing can be achieved when all projects are mated together such that a transition off one project results in the correspond transition onto another project.

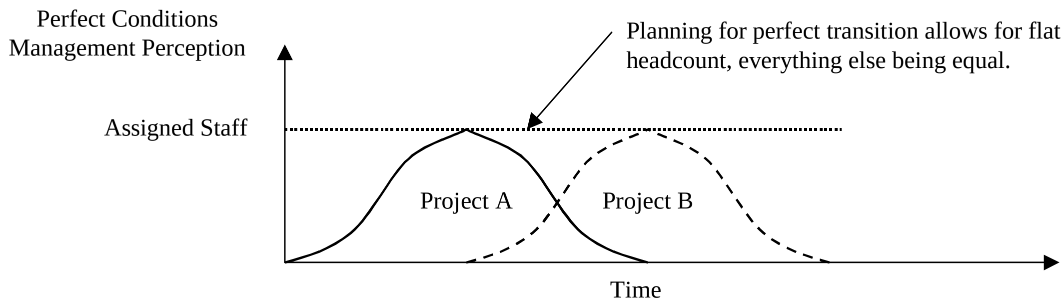


Figure 81 - Buffer Space Perceptions A

The proposed policy results in what may be perceived by management as an unwanted staffing gap as shown in Figure 82. The challenge is to change management's mental model such that they are not afraid of this apparent gap between onboard staff and planned staff. This situation is comparable to the challenge of continuous machine utilization and how it is not always the best strategy for an organization to follow.[28] If Project A finishes on time then the worst that can happen is that Project B either starts a little earlier and/or there is enough inter-project time to be utilized for training, reflection, and rejuvenation. If Project A finished later than planned a firefighting situation with Project B is avoided.

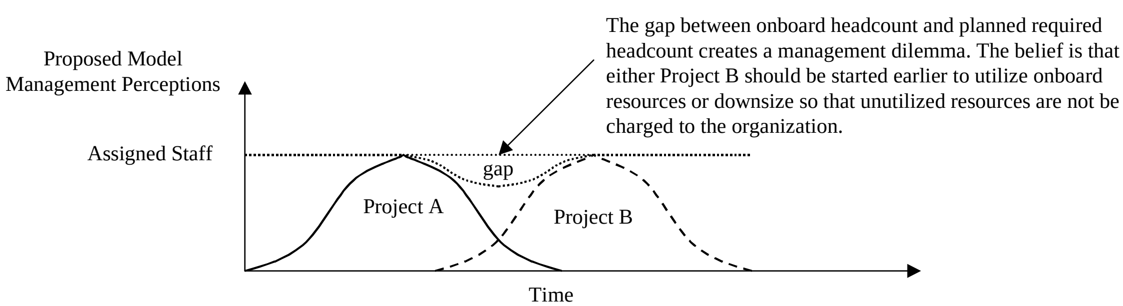


Figure 82 - Buffer Space Perceptions B

Allowing for no inter-project gap results it little or no time that can be solely dedicated to other causes. No down time results in the critical resources having to call work while they should be spending much needed time with their families. Too numerous are the stories of people trying to get a good dialup connection with their laptop computer while vacationing at a cabin in the mountains.

---

[28] Goldratt, Eliyahu M., The Goal. New York: North River Press, 1992., p. 32.

It's not enough for organizations to simply have policies regarding the reimbursement of educational or training expenses. Workers cannot benefit and advance their skills without also being allocated the time to pursue such endeavors.

Barriers to Implementation

- Managers can be trapped by the pervasive mental model that equates busyness with productivity. Having a planned buffer means that workers may theoretically be idle for some period of time between projects. In practice, this is unlikely the case since the majority of large development projects extend past the originally planned completion date. Nonetheless, managers are uncomfortable taking a chance. They overlook the fact that this unlikely "idle time" could still be utilized for less tangible but still beneficial things such as training.
- During long-term planning the organization may be overly optimistic relative to the amount of rework that may occur on projects. The tendency is for managers to set aggressive targets that challenge the organization. The typical belief is that planning for buffer in the schedule will lead to a self-fulfilling prophecy.
- Typical incentive systems are set up for rewarding managers that minimize expenditure of funds for a given set of work. This philosophy extends to the portfolio planning community as well. A planner that can squeeze five projects into five years may be considered more effective than a planner than can only squeeze four projects into the same five years.

Project Cancellations

While development organizations often have established procedures that call for the cancellation of projects that do not meet certain criteria (i.e. schedule milestones, profitability threshold, probability of success), these policies are not always enacted. While managers understand that in a theoretical sense, it may be healthier for an organization to cancel a project, nobody wants it to be their project. In addition, workers or managers not even directly involved with the cancellation of the project in question may have to live the negative stigma associated with a "failed" project. The mere association with a cancelled project can limit career growth. Unfortunately these canceled projects are not always somebody's "fault". They sometimes can be the result of a well-thought-out risk that simply did not pay off based on factors such as technology readiness or external market conditions. Nonetheless, canceling a project can be a positive event when viewed from the portfolio perspective. It can provide valuable lessons to be incorporated into other successive projects. It also frees up funds that can be utilized in hopefully more profitable ways.

Typical Current Policy

Projects are infrequently cancelled early in the product lifecycle.

Proposed Policy

Increase frequency of cancelled projects early in product lifecycle. Remove any future projects from the planning portfolio that have little chance of either starting on time or completing before marketplace opportunities have evaporated.

Traditional Logic

- We'll get the people we need soon.
- We'll fix the quality issues later. Basic functionality is what we need now.
- Just get a few demonstrateable features to work now so we can buy the time we need to get it right.
- We only need to get rid of a few of the features in order to make schedule.
- Our policy is to kill projects that have little chance of success, but this one is special and we need to keep it alive for the good of the company.
- Canceling a project is a sign of management and worker failure.

Flaws in Traditional Logic

- Hoping to get people soon usually does not account for the learning curve time required to bring new people up to speed. Also, it often does not account for the organizational complexity of putting people on an already late project.
- While 99% functionality or reliability may seem like a great accomplishment to workers on a project, customers may not feel the same. Customers often focus on the 1% that is not complete or that the product does not work reliably. Customers normally want a reliable product at low cost with the requested features, no more, no less. It is unlikely that they will give credit to the engineering organization for the years of work they may have put into a product, but rather blame them for the 1% that doesn't work correctly. This churn with the customer can lead to internal battles as well as loss of product momentum in the marketplace.
- Despite the fact that development organizations like to have the admiration of company executives, running a successful "demo" in front of senior management can often be one of the worst things to do. The dreaded words at the completion of a successful demo to senior management can be the words "ship it". While the engineering organization might know that the demonstration was a facade put together with Band-Aids and duct tape, management might now believe that the project is in good shape.
- Getting rid of 10% of the planned features typically does not eliminate 10% of the remaining work to do. By the time a decision is made to eliminate features, both the Requirements, and High-Level Design phases of a project may be complete. Eliminating features during the Development phase may mean re-staffing both the Requirements, and High-Level Design phases. This entire re-staffing and rework effort may far exceed the amount of effort to simply include the features in the product as originally planned. In essence, eliminating work in the later stages of development is often counter-productive.
- Despite the fact that workers on a project feel that this particular project is special in some way, they are likely not assessing the more holistic picture. Killing this particular project might seemingly wipe out thousands of hours of personal sweat. However, not all is lost since freeing the people to work on a project with more chance of success may be much better for the people and the company in the longer run.
- While canceling projects often carries with it the stigma of failure, the mental model should not rule out possibly much stronger positive attributes that can be associated with a canceled project. Freeing resources to work on a more promising project should lead to more success

for all involved. Also, carrying forward the lessons learned, both technical and otherwise, from a canceled project benefits all the successive projects.

Impact on Intellectual Capital

- One of the types of projects that should be canceled is one in which the market requirements have change sufficiently to not require the product any longer. In this case there is a risk that workers are spending time learning about outdated technology. Any time spent on this outdated technology is an opportunity for learning that is wasted and can not be recovered.
- Another type of project that should be canceled is one in which there are enough unforeseen issues that the program is running sufficiently late as to miss the market opportunity. In this case, it is likely that policies such as extended overtime have been put into place. In these situations, employee fatigue leads to increased attrition and decreased mentoring. Effects such as this impact overall intellectual capital in a negative fashion.
- The later a project is cancelled the more personal effort has been expended by individual workers. Employees can feel a great sense of personal loss at the cancellation of a project after it may have gone on for several years.

Discussion

Under perfect conditions, projects can flow smoothly from one to the next. As illustrated in Figure 83, the trailing edge of one project mates nicely with the leading edge of the next.

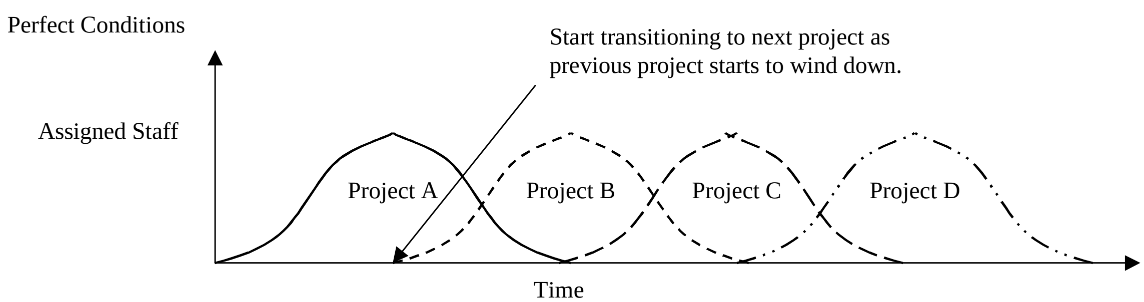


Figure 83 - Project to Project Flow Under Perfect Conditions

Unfortunately, it is infrequent that a project portfolio is able to operate over the longer term under such idealized conditions. A much more typical case is depicted in Figure 84 where Project A runs beyond its original planned completion date. In this situation, Project B begins with fewer than the planned number of resources and is likely to continually fight for resources with Project A. This in turn causes Project B to also slip beyond its assumed completion date. This domino effect can continue to the point where Project C and Project D have little if any chance for success.

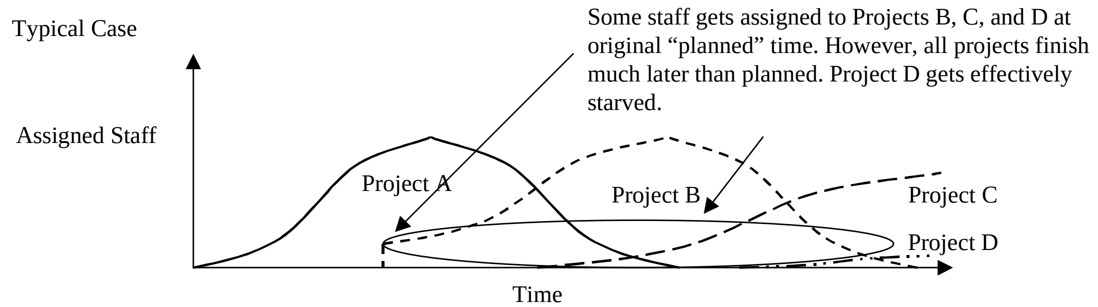


Figure 84 - Project Flow Under Typical Conditions

The proposed policy ensures successful completion of more projects in the portfolio when viewed as a whole since a chain of cascading late projects is broken. In the case shown in Figure 85, this is effectively done by canceling Project C before it has even begun. It should be apparent sometime during Project A or B that Project C has little or no chance of success.

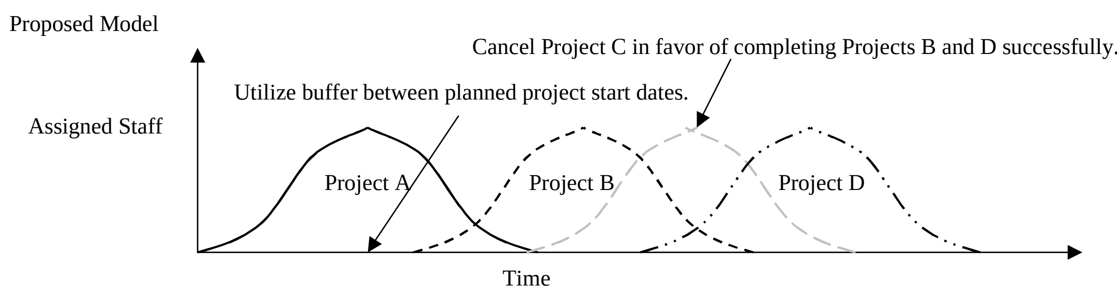


Figure 85 - Project Flow with Canceled Project

While managers admit that it makes sense to cancel projects that have little chance of success, they seldom want the canceled project to be theirs. No matter how dismal the outlook, human nature seems to take over as illustrated by the following story of Project X.

Project X was originally planned to take two years to complete at around thirty million dollars per year. It was intended to be the next generation product that would replace an existing legacy product that had been in the field for over 20 years. The intention was for Project X to last at least 20 years once it was placed into operation. As such, the development organization planned on ensuring this 20-year lifetime by using nothing less than the latest and greatest in available technology.

Due to the zest in utilizing the very latest in technology, Project X suffered from several early setbacks. It became apparent that it would unlikely be completed within the original plan of two years. There was tremendous pressure from outside the project to cancel and start again with less aggressive assumptions. Nonetheless, managers on the project refused to throw in the towel and admit defeat. They pleaded for just one more year. They even had the development organization piece together several demonstrations for senior management to show how the original concept could possibly work, if only given the chance. After six years and 200 million dollars Project X was finally cancelled. Despite having a great team, there were just too many dependencies on underdeveloped technology. The cancellation resulted in the merger with another existing project, called Project Y. All of the people on project X and Project Y were combined to form a single larger team. Their new charter was to develop future projects off of the older and less sophisticated Project Y. However, this caused significant disappointment for those associated with Project X. They had spent six years hard at work. They had been trained in the latest technology. Throwing away six years of work and reverting to the older technology of Project Y was just too much to bear for many. The result was that the organization eventually lost over half of the Project X staff.

Barriers to Implementation

- Even though most managers have been taught the fallacy of sunk cost, there is mental barrier regarding the "throwing away" of investments in time and money. A common mode of thinking is "we've already spent twenty million, another five million will allow us to complete the project". However, the remaining risk may not be worth the extra five million dollars that could be put to better use elsewhere in the portfolio.
- Employees may have to continue their careers living with the stigma of being associated with a "failed" project, even though they may not personally be to blame. This can end up leading to attrition of very qualified individuals.
- Projects that are in trouble create opportunities for people to be heroes. While being associated with a single failed project may ruin one's career, saving a project that has low probability of success can be a huge boost for a manager's or worker's career path. The

risk/reward trade-off may tempt some individuals to favor the continuation of a questionable project despite its potential negative impact on the overall project portfolio.

- Managers and workers alike fear the “mourning cycle” that follows a cancelled project. Rather than view the period as a time of rejoice and reflection, employees may end up using the time to hunt down those that are to blame for the apparent failure. There is a very strong tendency (i.e. very much established cultural norms) to not have a project cancellation party to celebrate the extra ten million dollars or fifty work-years of effort that was not wasted and is now bestowed upon the organization to put to better use.

Project Priorities

In addition to other policy actions such as overtime, adjusting the priority of projects is a mechanism for management to effectively divert both people and funding resources between projects without changing any organizational structure but only a few simple decision parameters. However, modifying project priorities in real time to adjust for shortfalls on one project may inadvertently damage other projects within the portfolio. Managers are really just trying to stabilize the system, not realizing that their actions may be met with resistance from those that are not beneficiaries of their prioritization decisions. For instance, a project that goes from a higher priority to a lower priority may institute an overtime policy to make up for the people resources the project may have lost during the reprioritization decision. Jay Forrester calls such phenomenon the "counterintuitive behavior of social systems".[29] A portfolio that is not perfect, but fairly stable, can be thrown totally out of balance and destabilized by rash decisions.

Typical Current Policy

Project priorities are adjusted frequently based upon current "crisis du jour" or "hot" issue.

Proposed Policy

Keep project priorities relatively static. Adjust priorities only after impact assessment of entire portfolio.

Traditional Logic

- We'll only need to reassign people to the "crisis" for a few days.
- Any decent manager should be able to recoup a few days of lost time if he loans people to the "crisis".
- The entire project depends upon fixing this one "hot" issue.
- Impact to overall product portfolio is not significant.

Flaws in Traditional Logic

- Over a multi-year project, loaning people to several of these "crisis" issues can add up to substantial loss in productivity and momentum.

[29] Forrester, J.W., "Counterintuitive behavior of social systems", Technology Review, 73(3), 1971., pp. 52 – 68.

- Managers are only as good as the people that work for them. A single incident is probably recoverable. However, there are finite limits as to how much a project plan can be perturbed without suffering.
- It's human nature to always classify temporally near events as more important than temporally distant events.
- Seemingly insignificant events on a single project can ripple through an entire series of successive projects.

Impact on Intellectual Capital

- While some individuals thrive on being at the center of attention, most workers get fatigued after more than short spurt of firefighting.
- Over time, crisis management can become the status quo an result in less time dedicated to employee skill development.

Discussion

Figure 86 illustrates the scenario in which two projects operate under somewhat ideal conditions. Neither project undergoes any crisis period that prompts management to temporarily elevate priorities.

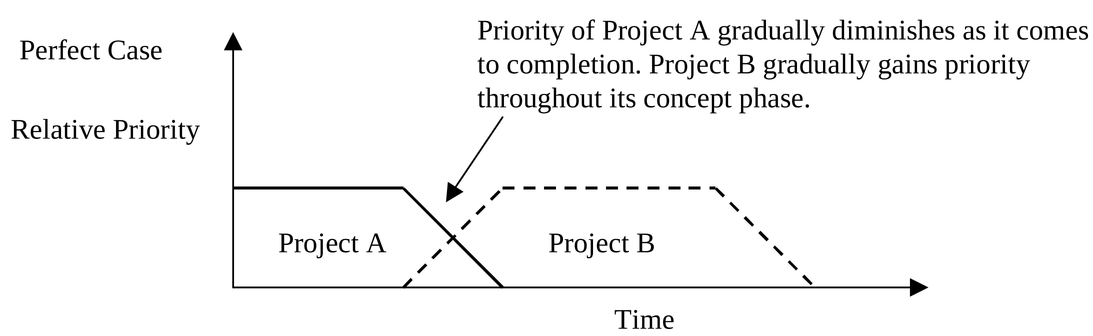


Figure 86 - Project Priorities Under Perfect Conditions

Not surprisingly, few project portfolios can exist for long without some type of unforeseen issues developing. It is during these times of crisis that projects can undergo periods of priority escalation as depicted in Figure 87. The challenge for management is to not view any one project in isolation but rather assess the impact of their prioritization decisions on the overall portfolio. Otherwise, seemingly endless priority escalation duels can lead to wasteful firefighting.

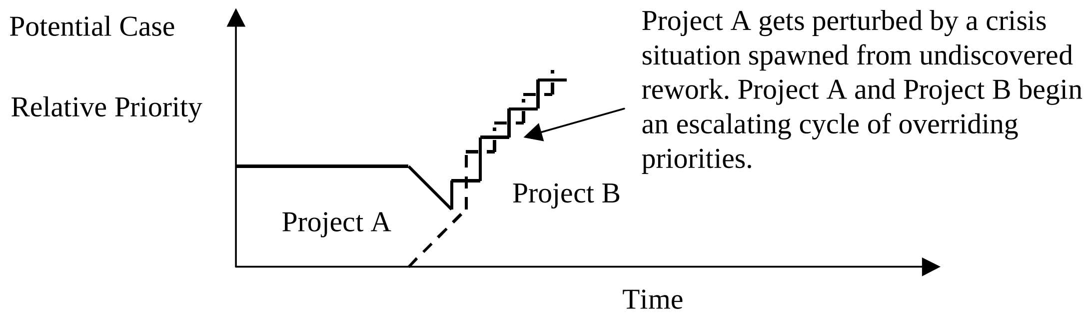


Figure 87 - Project Priorities Under Imperfect Conditions

Project priority escalation cycles such as that show in Figure 87 can go on for some period of time before they are short-circuited by some external event. One unfortunate case can be one or both of the projects getting cancelled due to failure to meet delivery expectations. Another case can be that of senior management stepping in and declaring the "loosing" project to always be at a lesser priority level than the other project.

Barriers to Implementation

- There is a strong tendency for managers to place more value on the temporally near-term events over events that may or may not occur in the future. The typical incentive system does not place as much weight on the success of the longer-term portfolio as it does on the success of single projects.
- Not all managers are well versed in "systems thinking". If they were, they would realize that seldom can decisions be made in isolation without somehow effecting other things within the system – sometimes negatively. Learning about the dynamic nature of project portfolio management may be outside the scope of the expertise that is typically applied in reprioritization decisions.
- Common management principles and corporate dogma lead one to believe that organizations should be able to adjust to dynamic conditions. Unfortunately, just as individuals have difficulty multi-tasking and context switching, so do organizations. Things like prioritization escalation between projects may not be recognized until after the damage has already been done to the portfolio.

Portfolio Balance

Successful organizations must be able to attract, develop, and retain qualified individuals. There are numerous methods for accomplishing all three goals. Among these are things such as signing bonuses, education programs, and stock option plans. However, another method that can aid in accomplishing the three goals is to offer challenging work that allows for immediate job satisfaction as well as longer-term individual intellectual growth. While an organization must meet the needs of its customers in order to make a profit, companies must also be able to develop superior intellectual capital if they want to succeed as market leaders. "Simple" projects are more often the projects that have a higher probability of being completed close to schedule and budget, while "complex" projects normally have a higher probability of being late and over budget. However, "complex" projects allow employees to learn many skills at an accelerated rate, helping to increase the on-board intellectual capital. In addition, it is these coveted projects that also help retain the best employees inside and attract the best workers from outside the company.

Typical Current Policy

Projects are evaluated primarily based on direct monetary return of investment.

Proposed Policy

Assess attractiveness of proposed projects based on a combination of contribution to intellectual capital growth as well as traditional monetary return on investment.

Traditional Logic

- The more money a project makes the better for the company.
- It's better to bleed a cash cow dry before investing in unproven projects.
- We'll use our profits from our cash cows to fund "knowledge development" as we need it.
- It's no use in training people before they need it. If we don't do "just in time" training, people will just forget.
- We'll just outsource the project if our people don't know how to do it.

Flaws in Traditional Logic

- In the short term, projects that generate more cash flow may be better for the company. However, in the long term it may be better to invest in more diversified projects that aid in intellectual capital development.
- Depending upon the field, developing substantial intellectual capital may take decades. By the time an organization realizes it is lacking, it may be too late to recover.
- Funding typically gets directed to things other than pure intellectual capital development. One of the first things to get slashed during budget crunches is training and technologically risky development projects.
- "Just in time" doesn't always apply to complex skill development. Many skills literally take years to master despite the fact an employee might have been granted a certificate in a one-week seminar.
- Outsourcing core technology work can lead to significant erosion of internal skills base.

Impact on Intellectual Capital

- Neglecting the impact of project mix on intellectual capital development can lead to imbalances in staff skill levels. There are different optimal resource skill mixes depending upon the industry and type of work involved. However, the key is to maintain an appropriate mix of skill levels within each phase of a project.
- All Novices means there is nobody left over to mentor. It also signals that for some reasons not many people are left from the previous project. Otherwise, there would be some Experts around.
- All Experts means there is nobody left to do some of basic work. It could also signal that the organization is stagnating and not bringing in enough fresh ideas.
- Having a disproportionate number of Expert means that the organization may be at risk of a retirement loss or even salary compression issues.
- The balance in intellectual capital can be very delicate. Once the mix of people gets skewed one way it may be difficult to re-achieve equilibrium.

Discussion

Figure 88 illustrates some type of equilibrium between skill levels. While the number of Novices, Intermediates, and Experts need not be the same, management should seek some sort of balance depending upon the type of project portfolio.

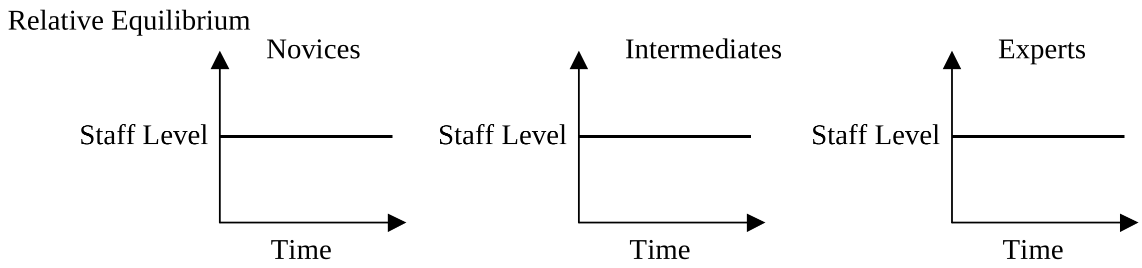


Figure 88 - Relative Staffing Levels A

Figure 89 depicts a scenario where the relative number of Experts is increasing and the relative number of Novices is declining. Situations such as this can be a signal that there are not enough new hires being attracted to the company.

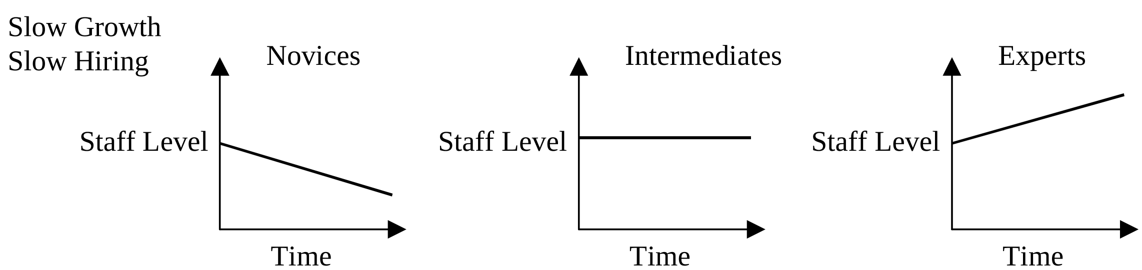


Figure 89 - Relative Staffing Levels B

In contrast to Figure 89, Figure 90 illustrates the opposite situation. Situation such as this could signal that there are not enough "high complexity" projects in the portfolio or that too many of the experienced workers are being driven out resulting in higher than anticipated attrition.

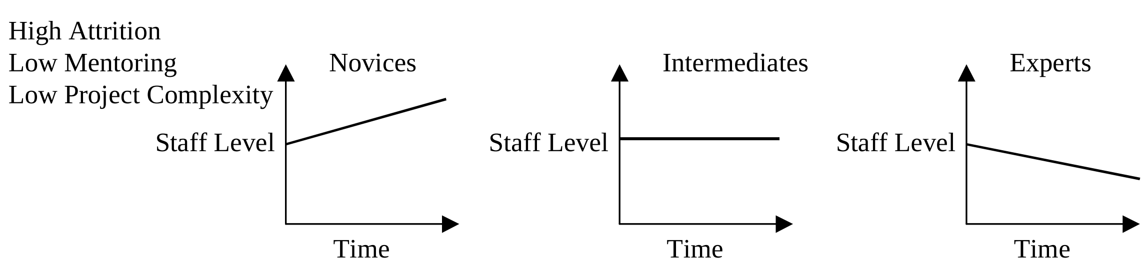


Figure 90 - Relative Staffing Levels C

While the most complex projects are often the most risky in terms of schedule and successful completion they can also be the same projects that contribute to intellectual capital at the greatest rate. The most complex projects allow internal skills growth as well as attract the best people from outside the organization.

Barriers to Implementation

- Project complexity is somewhat subjective. One manager may consider a project to be complex enough to further develop skills of on-board workers while another manager may consider the same project to be of little value at all.
- Intellectual capital can be difficult to quantify. Management may consider their current project staff to be well versed in the required technology that is utilized at the company. If this is the case, management may see little need in expending extra time or funds in working on complex projects.
- Managers are all trained in skills such as “listening to the customer”. There are environments in which current customers do not necessarily ask for “complex” products. In these situations, management may opt for just giving the customers exactly what they ask for over the short-term and neglect the longer-term implication of degraded intellectual capital.

Chapter 8: Model Limitations & Future Work

While the existing Vensim model serves in gaining insight into the management of multi-project portfolios, there are obviously limitations. Experienced system dynamics modelers are often quoted as saying “All models are wrong, but some are useful”. The trick is to find a balance between the one extreme of modeling a problem down to excruciating detail and the other extreme of trivializing the problem at hand. Developing a Vensim simulation for either extreme can lead to a very elegant but useless system dynamics model.

With the current level of detail, our model has enabled us to gain further insights into the issues surrounding and managing a multi-project portfolio. Nonetheless, there are areas where the model is not as detailed as it could be. It is possible that the addition of such detail could lead to yet further insights into the organizational dynamics surrounding a multi-project enterprise.

Seven items of possible exploration are presented in the discussion that follows.

People and Late Projects

As projects exceed planned completion dates due to excessive levels of work-to-do, the ratio of actual workforce to desired workforce impacts attractiveness such that workers are reassigned from projects that may currently be on track. Additional model structure would be beneficial in order to address other strategies for mitigating the crisis. One example is for the project deadline to be replanned. This strategy could reduce the sudden jump in project attractiveness due to resource gap. Other mitigating strategies that may translate to model structure include reducing the scope of work or outsourcing work that has yet to be started.

Model Disaggregation into Sub-Models

One model represents all formulations in an aggregate form. This virtually eliminated the ability to develop model structure in parallel. Model debugging was also difficult in that the effects of many variables were visible in the simulation results. Isolation of variables into sub-models would be beneficial.

Scale Model to N Projects

The model is constructed so that up to four concurrent projects can be simulated at a time. This enabled sufficient inter-project dynamics to provide useful insights. However, the Vensim structures could be appended so that more concurrent projects could be simulated. This would be useful to organizations with a larger number of projects in the active portfolio.

Best/Worst Case Planning Conditions

The existing user interface allows for specific input regarding project parameters. Often this specific input is unknown, especially when in the early project planning stages. However, a range of project data input, that is based on recent or past experience or rules of thumb, is usually readily available. The model could possibly be extended to account for things like a range of acceptable project completion dates rather than only a specific parameter.

Comparison Metrics

The model simulations are compared using aggregate metrics that have representative meaning to a project manager. The current comparison metrics comprise total intellectual capital measured in terms of people, total project effort measured in people-months, total people within the organization and project total or phase duration. Expanding the list of metrics to include those useful in other organizations would expand the application of the current model.

User Interface

The user interface that exists now is in the Microsoft Excel spreadsheet format. This interface presents the user with all the project portfolio configuration parameters that may be modified for model simulation. This interface is static in nature as it requires setting parameters then running the model. Dynamic controls within the user interface would permit a "real-time" ability to explore decision scenarios.

People Centric Model

The model developed in this research focuses on a project view rather than on a people perspective. Additional insight may be derived from an examination where individuals and the tasks they perform are tracked.

Appendix A
Simulation Test Data

Case 1

1. Case Description: Four Sequential Projects, Same Duration, Perfect Quality

2. Rational for Use: Base Case for Sequential Projects with Perfect Quality Representative of an Ideal World Where No Mistakes are Made by Employees.

3. Model Constants: Reference Vensim Input Sheet

   The Following Table Functions are Adjusted Such That There is No Effect from Function Output, Function Output Set to 1;

   Effect of Fatigue on Productivity

   Effect of Fatigue on Quality

   Complexity Effect on Productivity

   Complexity Effect on Quality

   Projects That Begin Earlier Have Higher Priority as Follows;

Project	Priority
1	4
2	3
3	2
4	1

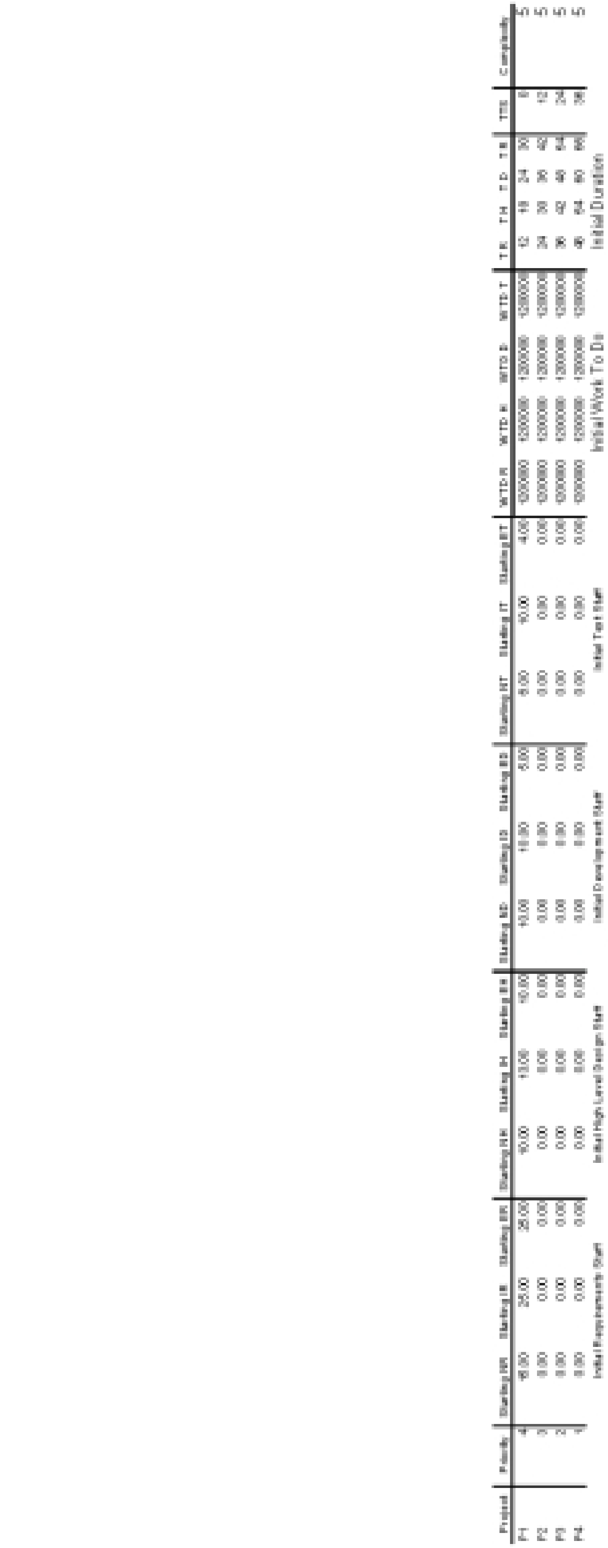


Figure 91 - Case #1 Initial Simulation Parameters

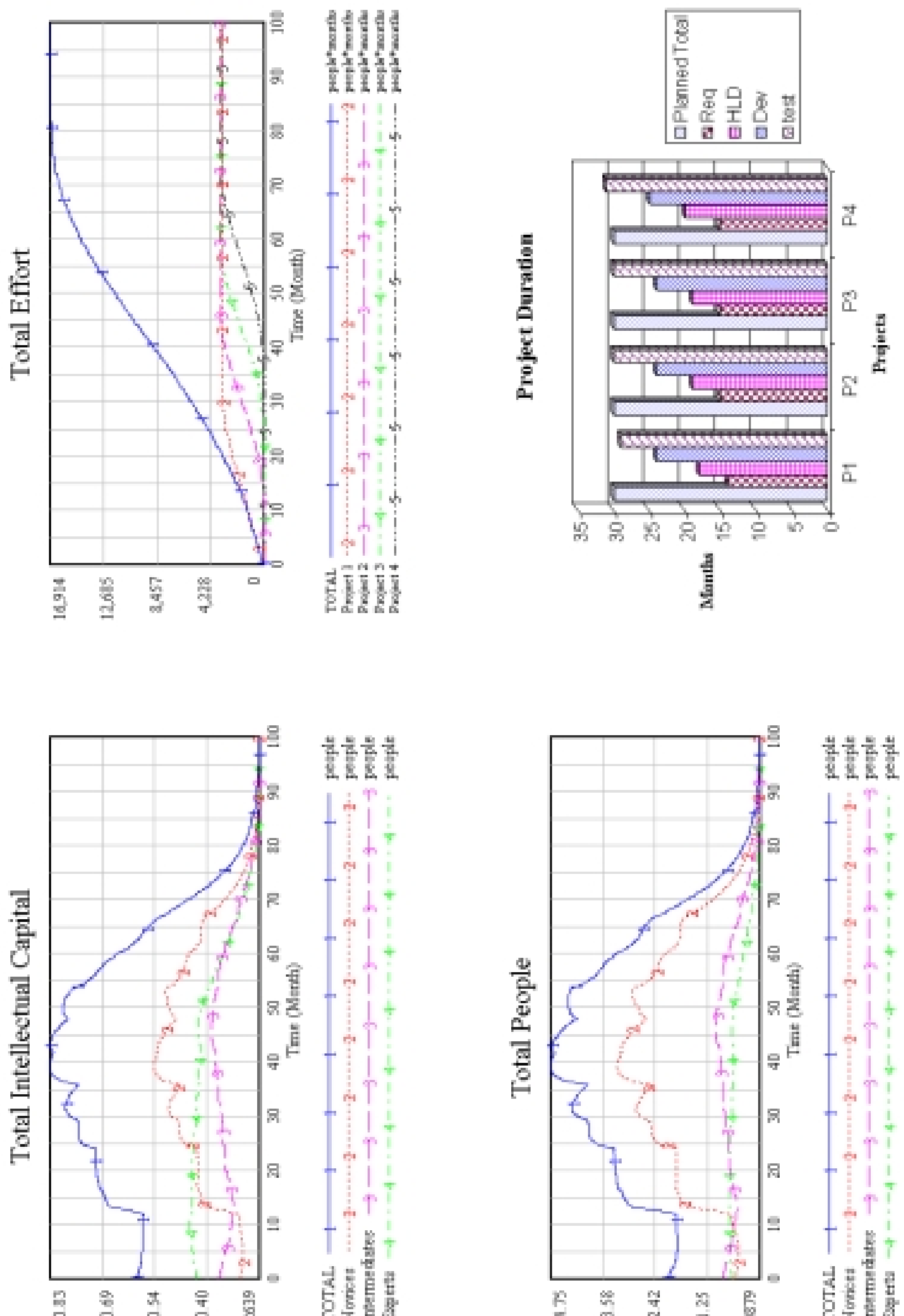


Figure 92 - Case #1 Simulation Results

Case 2

1. Case Description: Four Sequential Projects, Same Duration, Imperfect Quality

2. Rational for Use: Serves as Base Case for Comparison to Other Cases and Demonstrates Existing Mental Model that Current Projects are More Important than Future Projects. This Effect is Executed Through the Assignment of Higher Priority to Projects that Begin Sooner. As an Example;

Project	Priority
4	4
5	3
6	2
7	1

3. Model Constants: Reference Vensim Input Sheet

All Projects Have the Same Intermediate Level of Complexity.

Project	Complexity
1	5
2	5
3	5
4	5

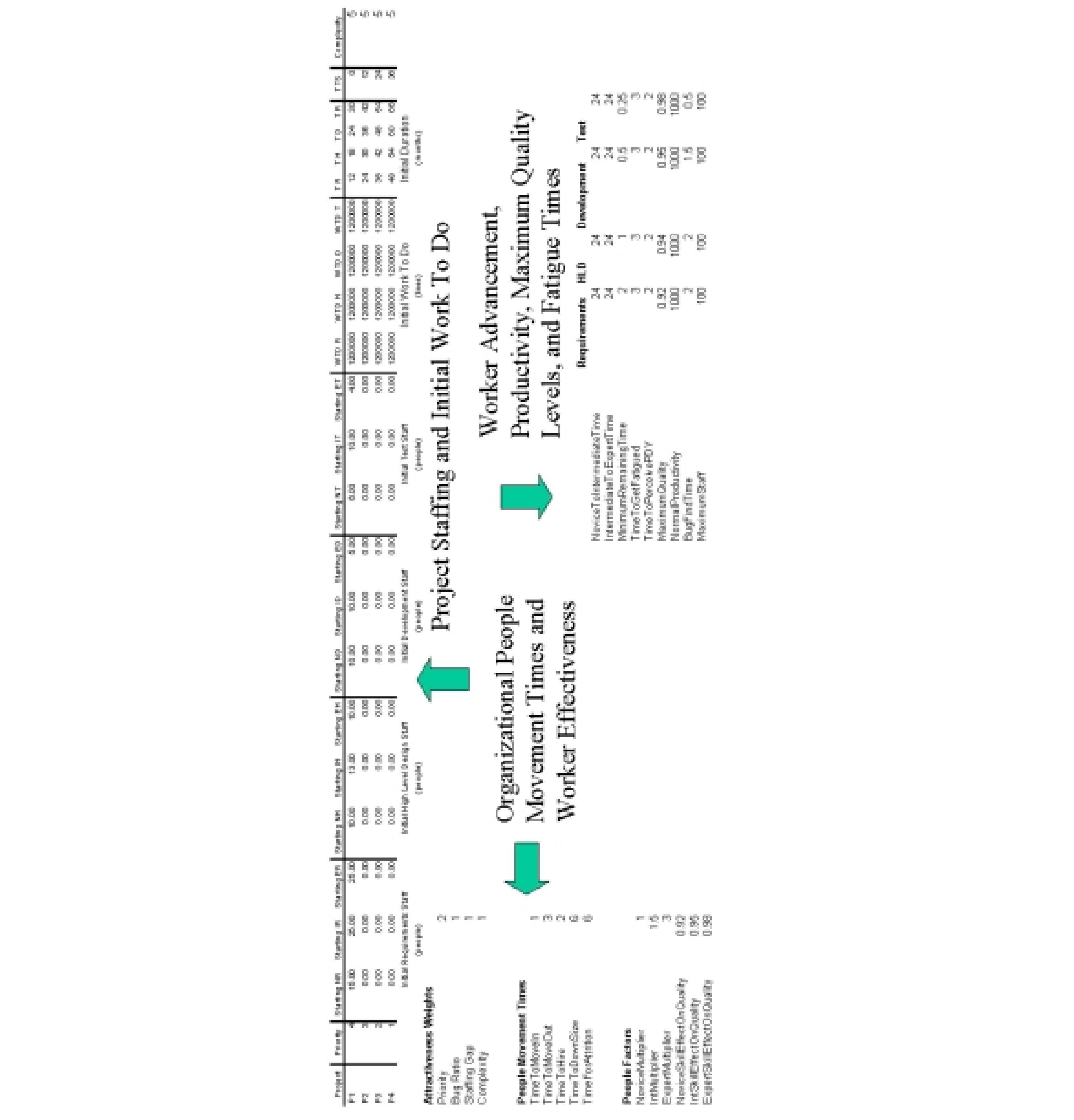


Figure 93 - Case #2 Initial Simulation Parameters

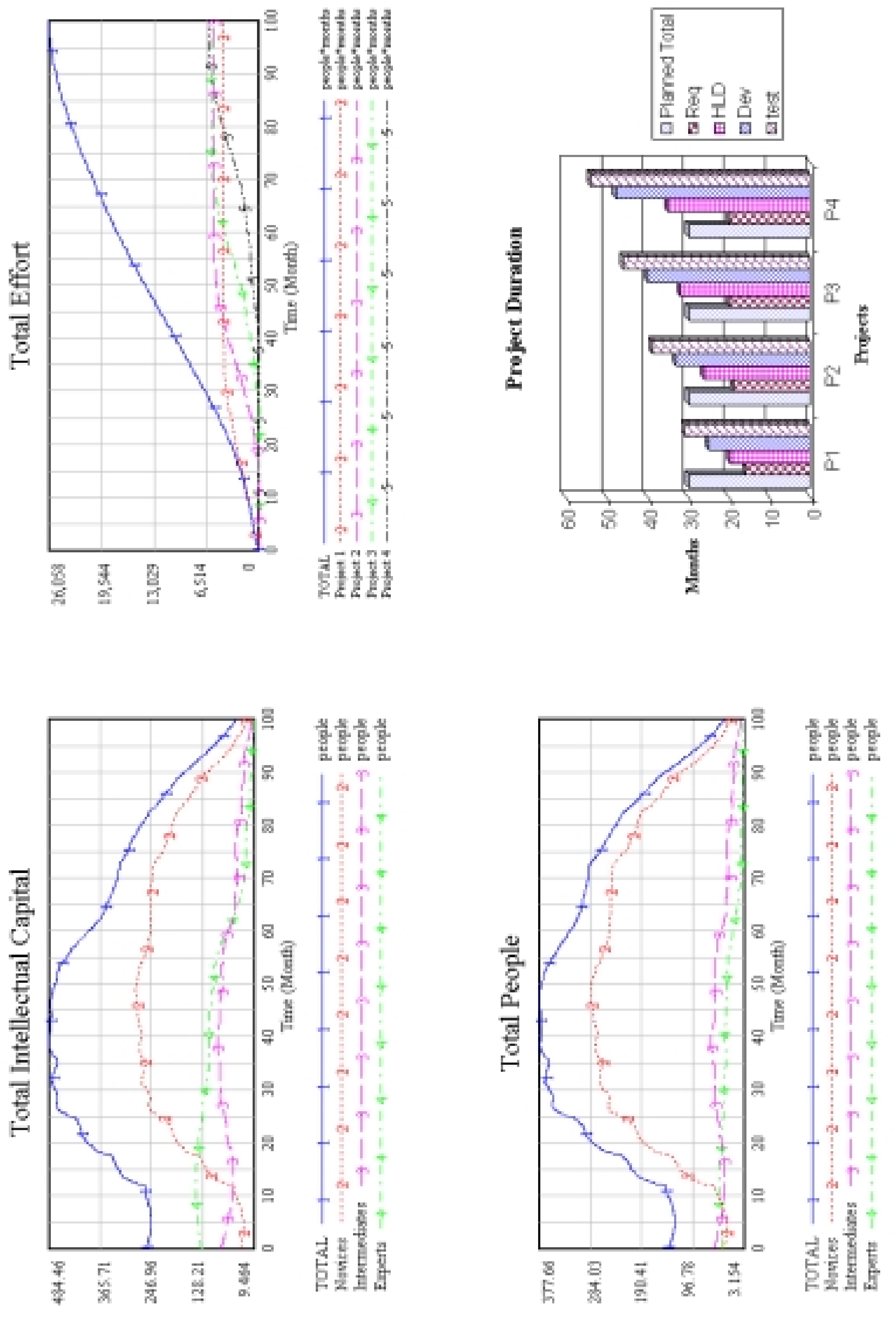


Figure 94 - Case #2 Simulation Results

Case 3

1. Case Description: Four Sequential Projects, Same Duration, Remove Project 3

2. Rational for Use: Test Impact of Removing the Overburden.

3. Model Constants: Reference Vensim Input Sheet for Case 2.
   Dropped Project 3 by Setting Priority to Zero.

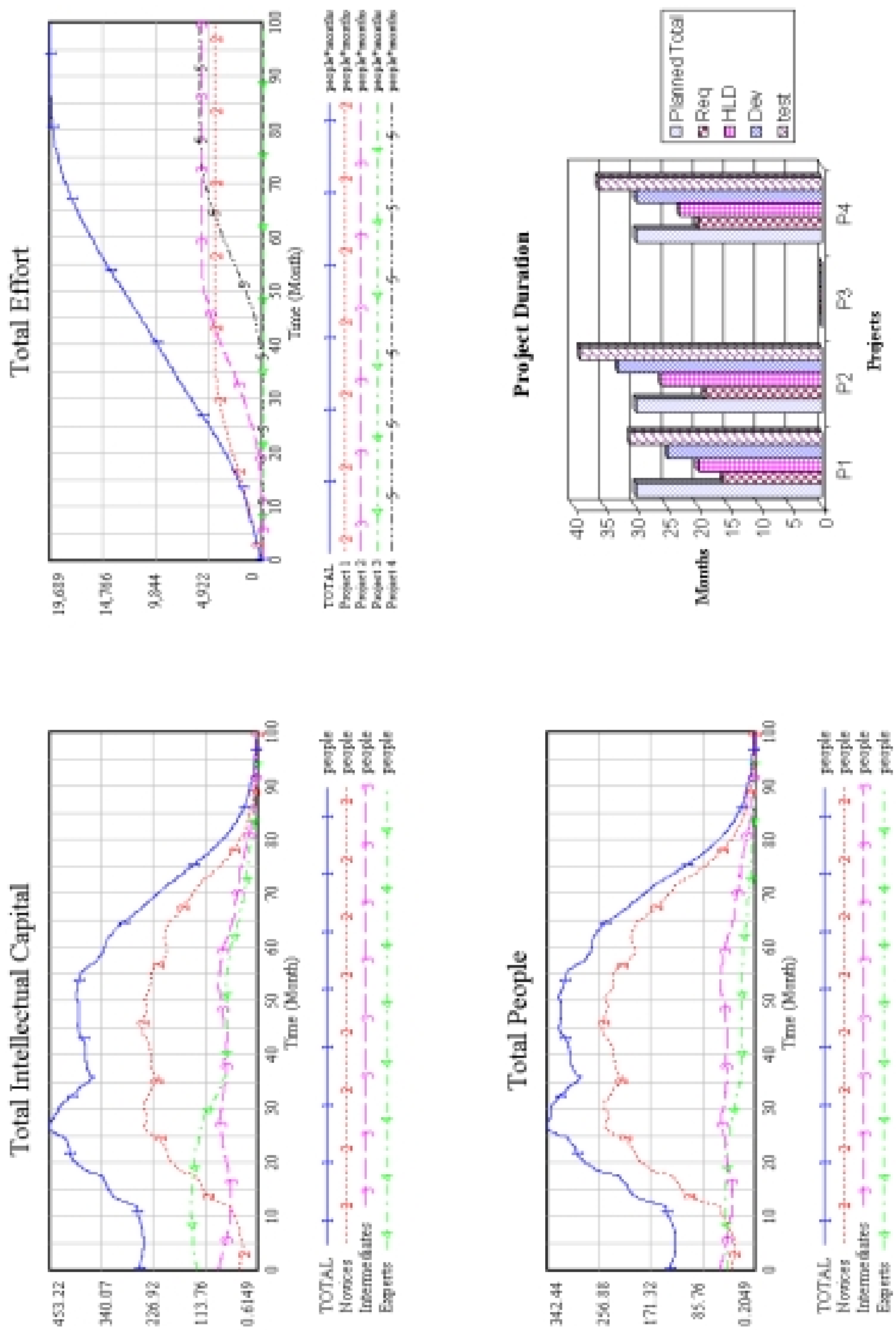


Figure 95 - Case #3 Simulation Results

Case 4

1. Case Description: Four Sequential Projects, Same Duration, All Complex

2. Rational for Use: Look at Extreme Case for Retaining & Developing Intellectual Capital.

3. Model Constants: Reference Vensim Input Sheet for Case 2.

All Projects Have the Same High Level of Complexity.

Project	Complexity
1	10
2	10
3	10
4	10

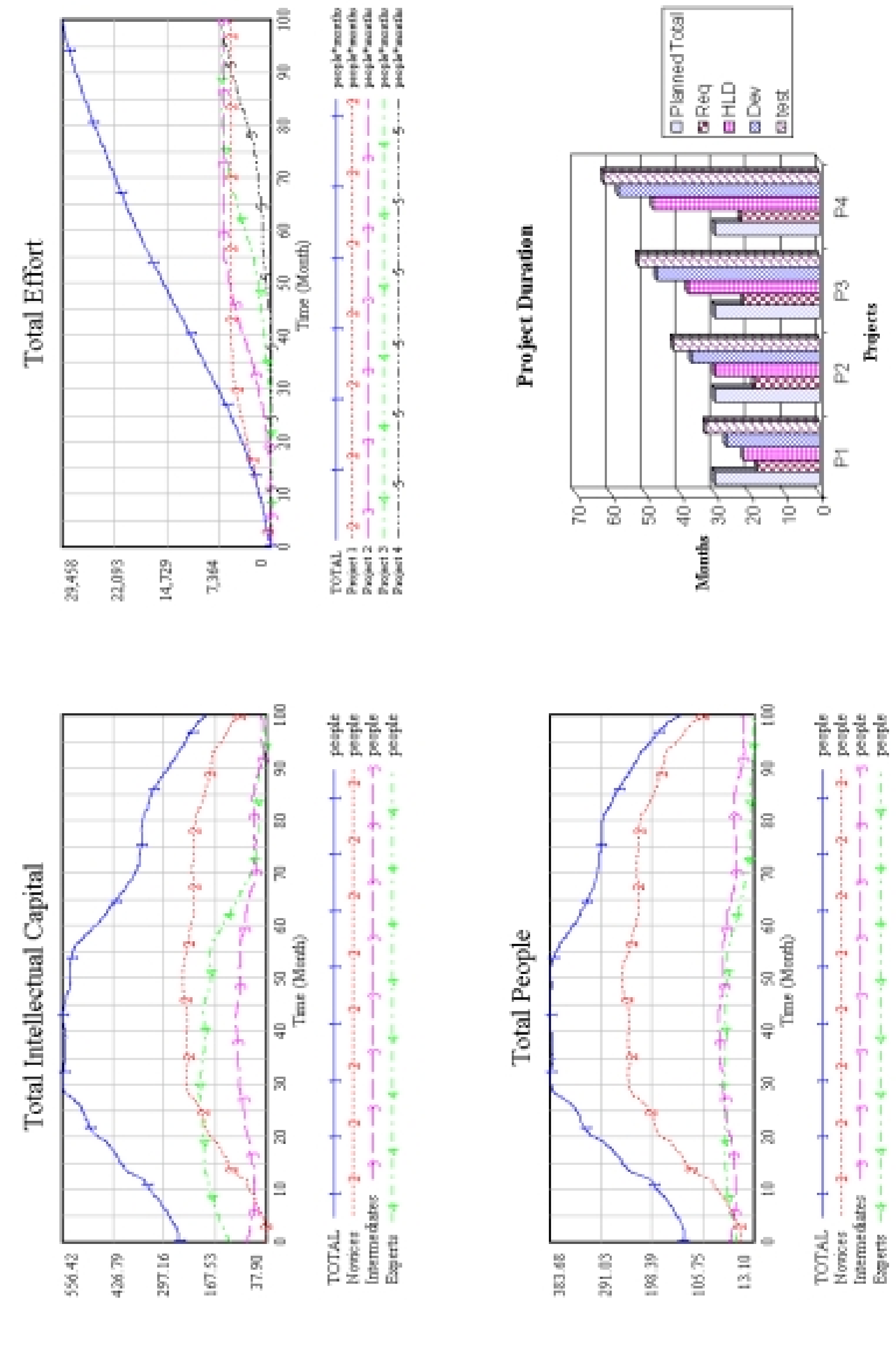


Figure 96 - Case #4 Simulation Results

Case 5

1. Case Description: Four Sequential Projects, Same Duration, Complexity; Simple, Complex, Simple, Complex

2. Rational for Use: Intermediate Complexity Case

3. Model Constants: Reference Vensim Input Sheet for Case 2.
Projects Alternate Between Low Complexity and High Complexity.

Project	Complexity
1	1
2	10
3	1
4	10

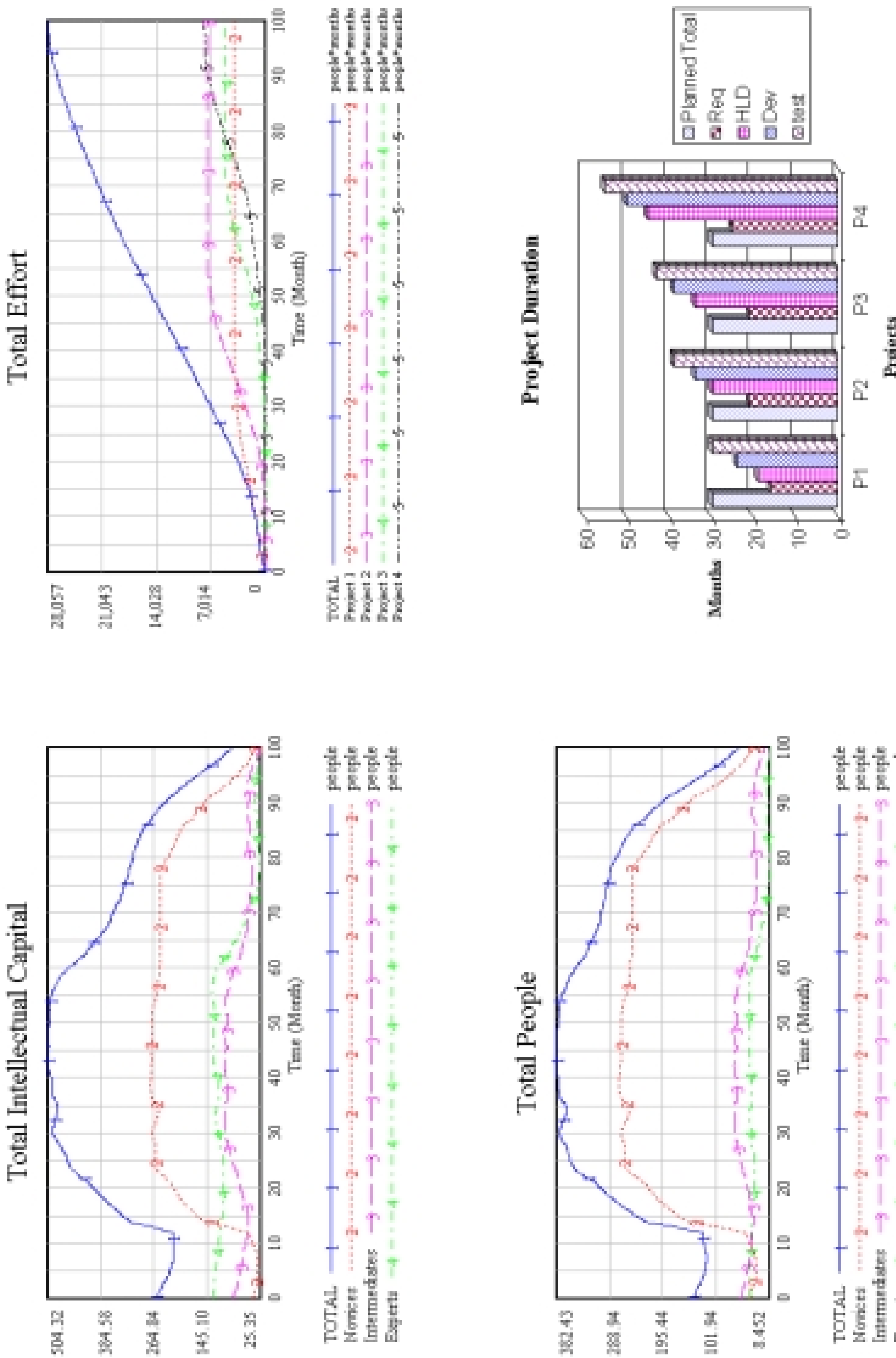


Figure 97 - Case #5 Simulation Results

Case 6

1. Case Description: Four Sequential Projects, Same Duration, Equal Small Gaps Between Projects

2. Rational for Use: Test effect of Allowing Space for Rework Discovery & Other Unforeseen Circumstances.

3. Model Constants: Reference Vensim Input Sheet for Case 2.
Additional Time for Rework is Provided by Deferring the Start of Project 2 by Three Months, Project 3 by Three Months and Project Four by Three Months. Project Duration Remains the Same.

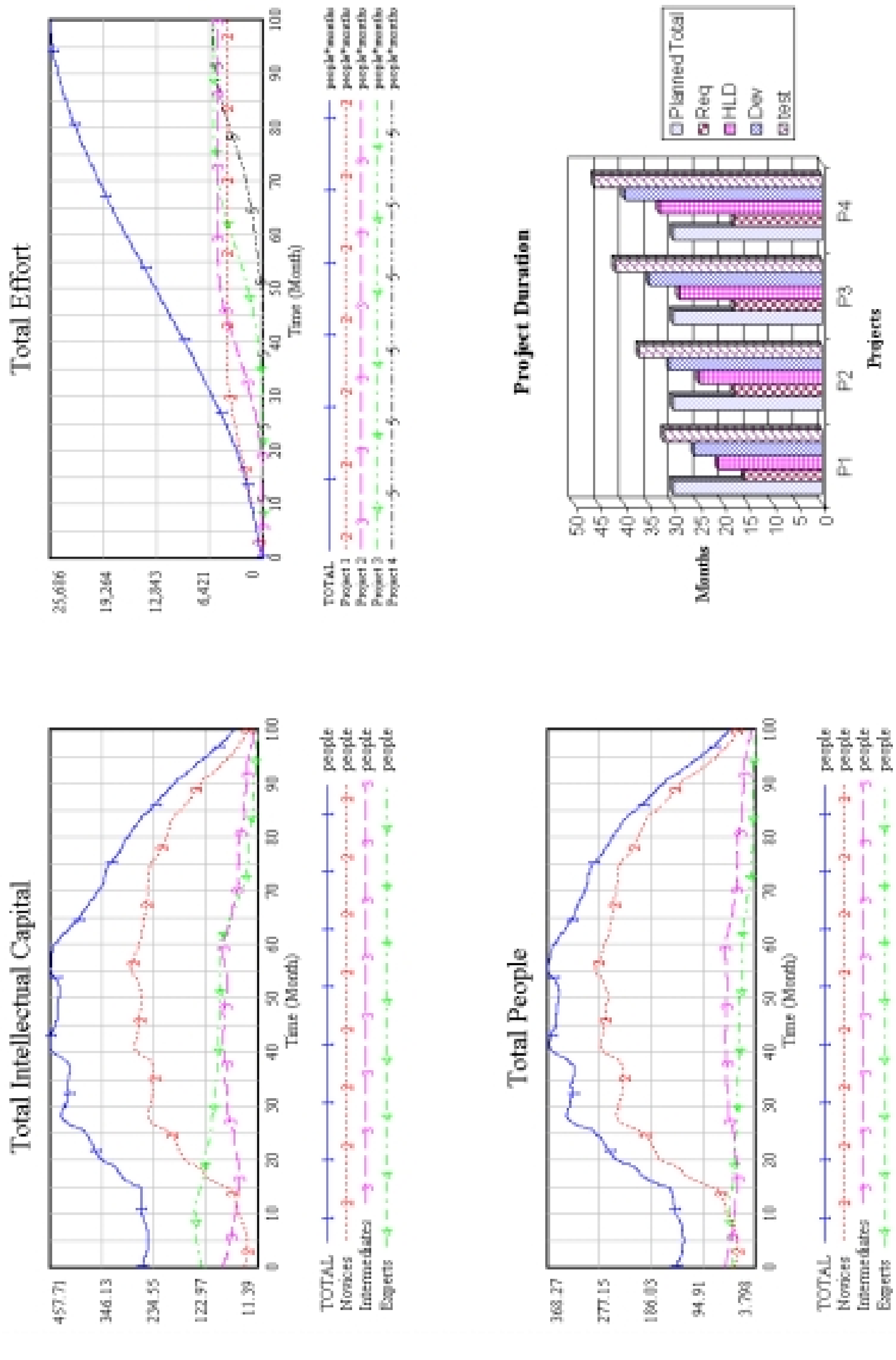


Figure 98 - Case #6 Simulation Results

Case 7

1. Case Description: Four Sequential Projects, Same Duration, Vary Bug Find Time

2. Rational for Use: Explore Effect of Projects with Delayed Rework Discovery on Other Projects.

3. Model Constants: Reference Vensim Input Sheet for Case 2.
   The Bug Find Time is Three Times Longer Than the Base Case, Case 2.

	Requirements	HLD	Development	Test
Case 2: BugFindTime	2	2	1.5	0.5
Case 7: BugFindTime	6	6	4.5	1.5

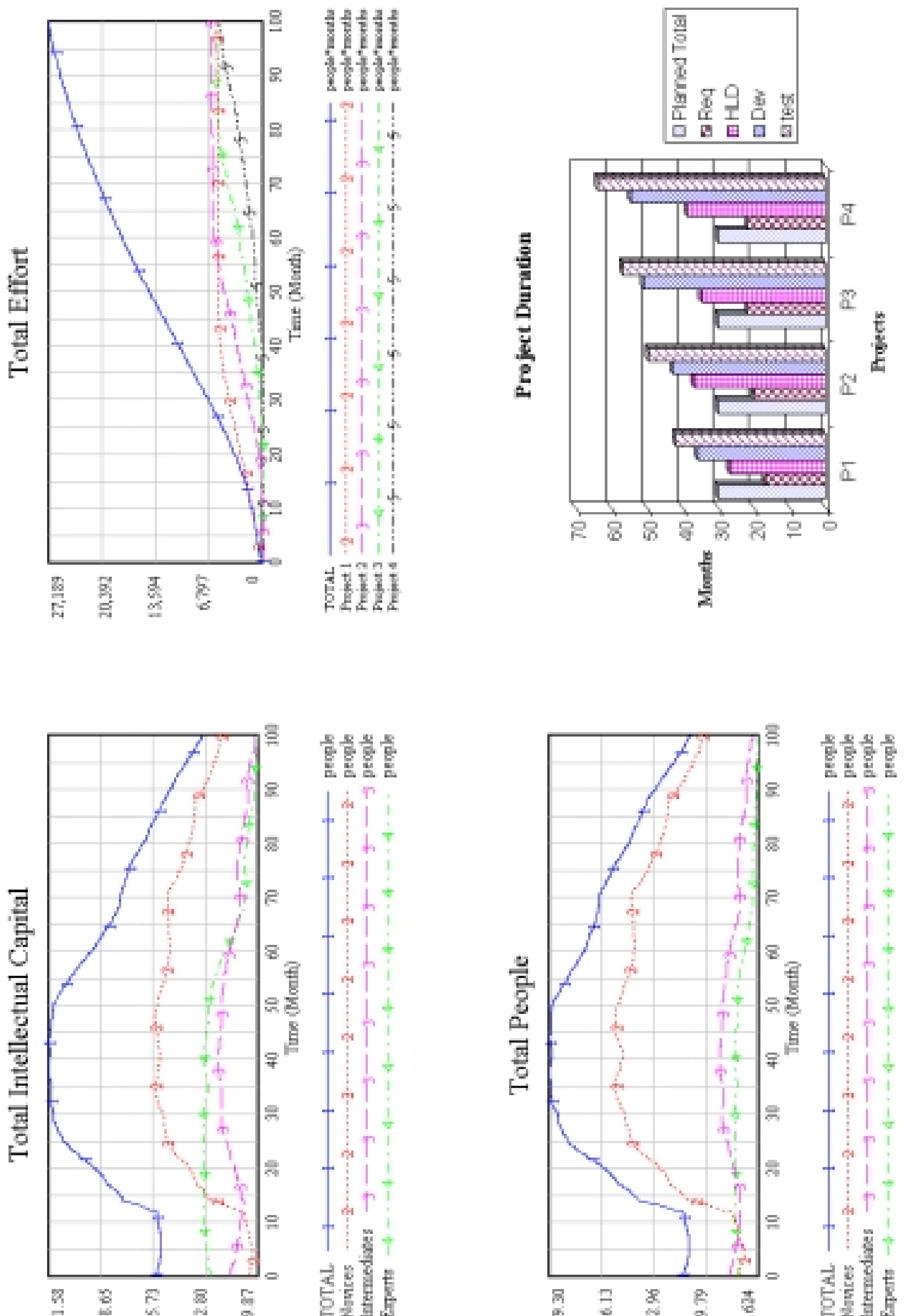


Figure 99 - Case #7 Simulation Results

Case 8

1. Case Description: Four Sequential Projects, Same Duration, Project 3 High Priority

2. Rational for Use: Demonstrate Effect of Unexpected Priority Shifts.
Priority Shifts May be Due to Changes in Market Conditions.

3. Model Constants: Reference Vensim Input Sheet for Case 2.
Project Priority Adjusted as Follows;

Project	Priority
1	4
2	3
3	10
4	1

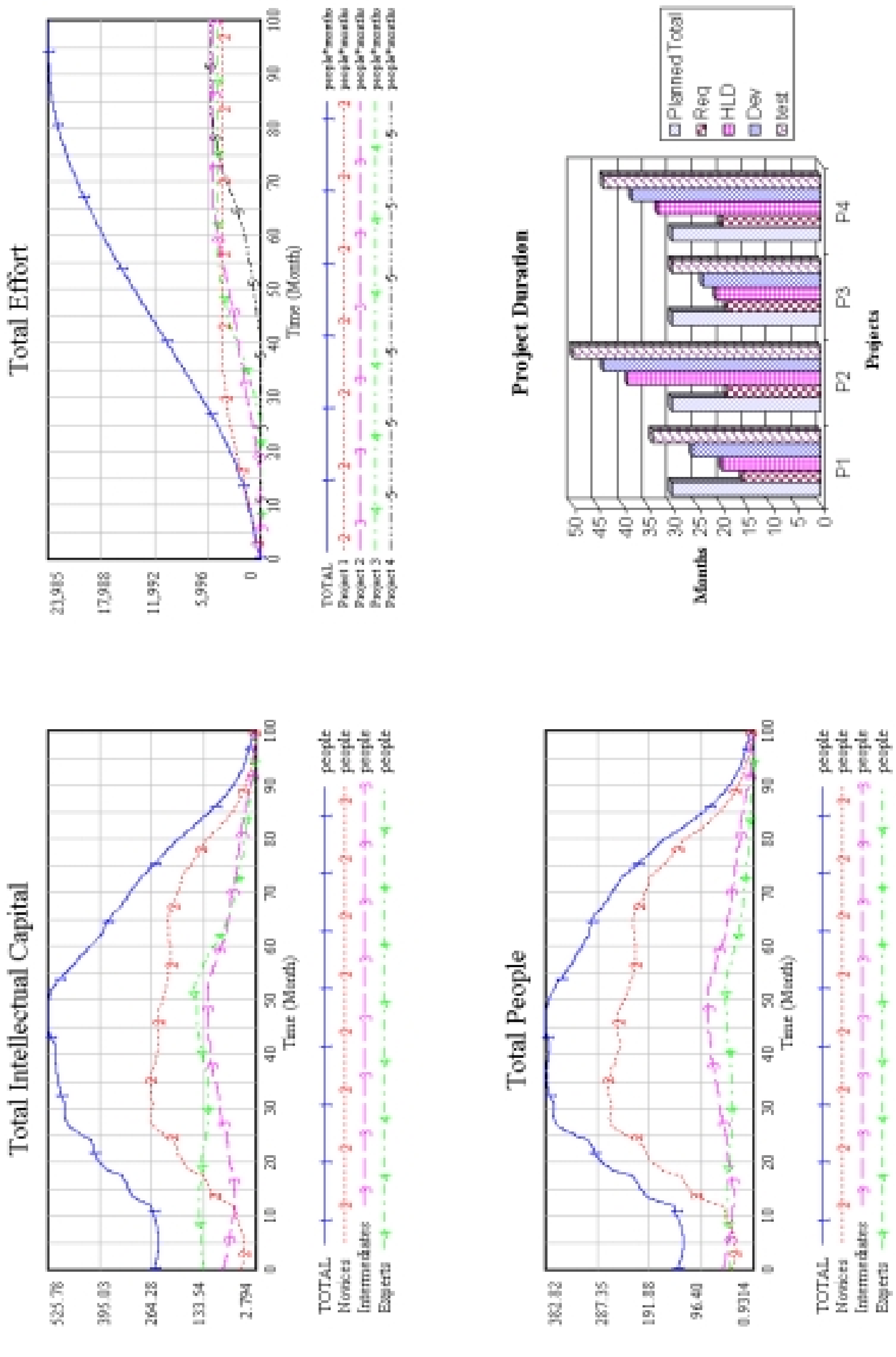


Figure 100 - Case #8 Simulation Results

Case 9

1. Case Description:	Four Sequential Projects, Same Duration, Adjust Time for Resources to Move Off a Project Two Cases - Fast, Slow
2. Rational for Use:	Test Effect of Manpower Availability for Discovery of Rework
3. Model Constants:	Reference Vensim Input Sheet for Case 2. Time to Move Out of A Project is Set to Three Months in the Base Case for Imperfect Quality, Case 2. Adjust for Case 9A to One Month. Adjust to Six Months for Case 9B.

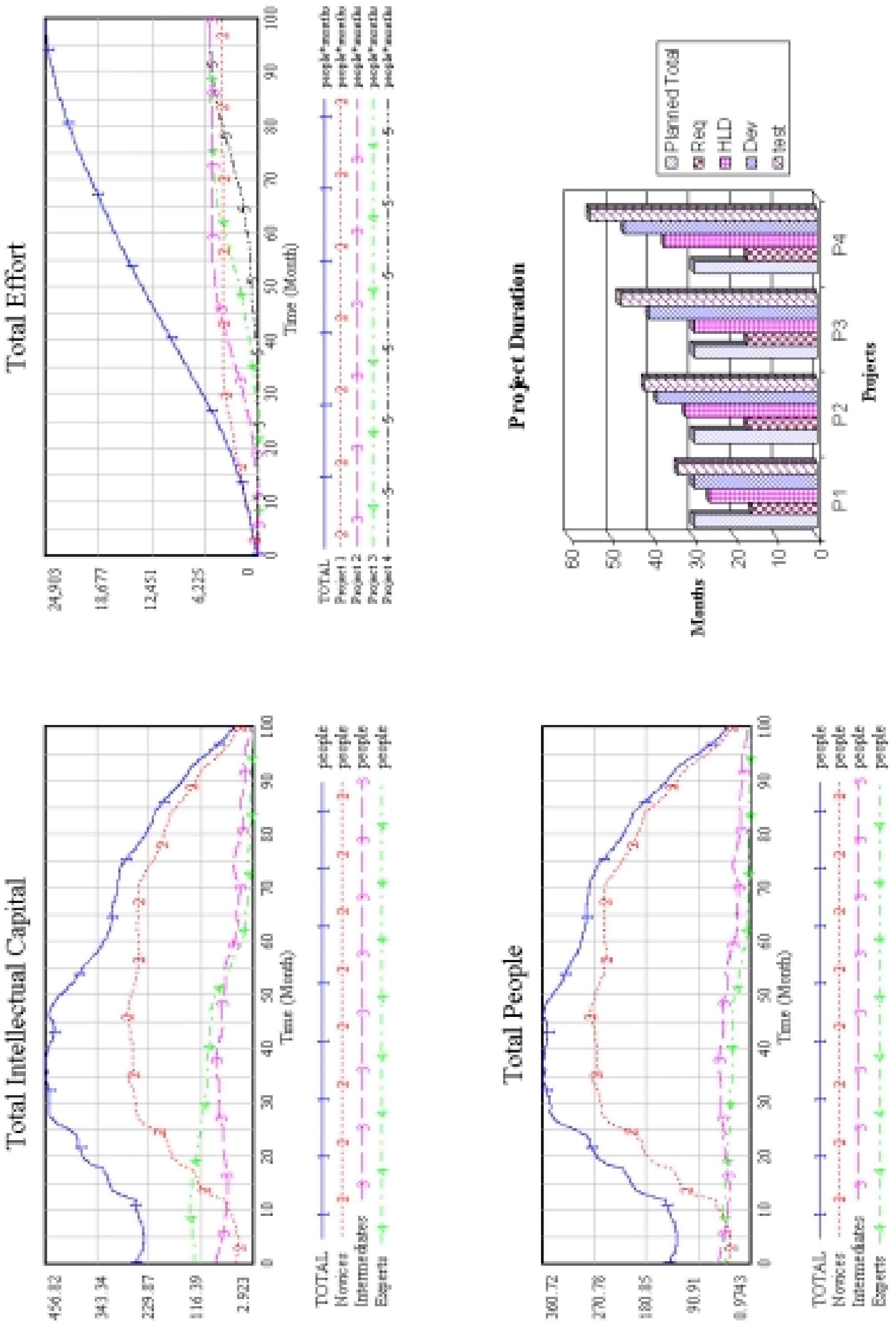


Figure 101 - Case #9a Simulation Results

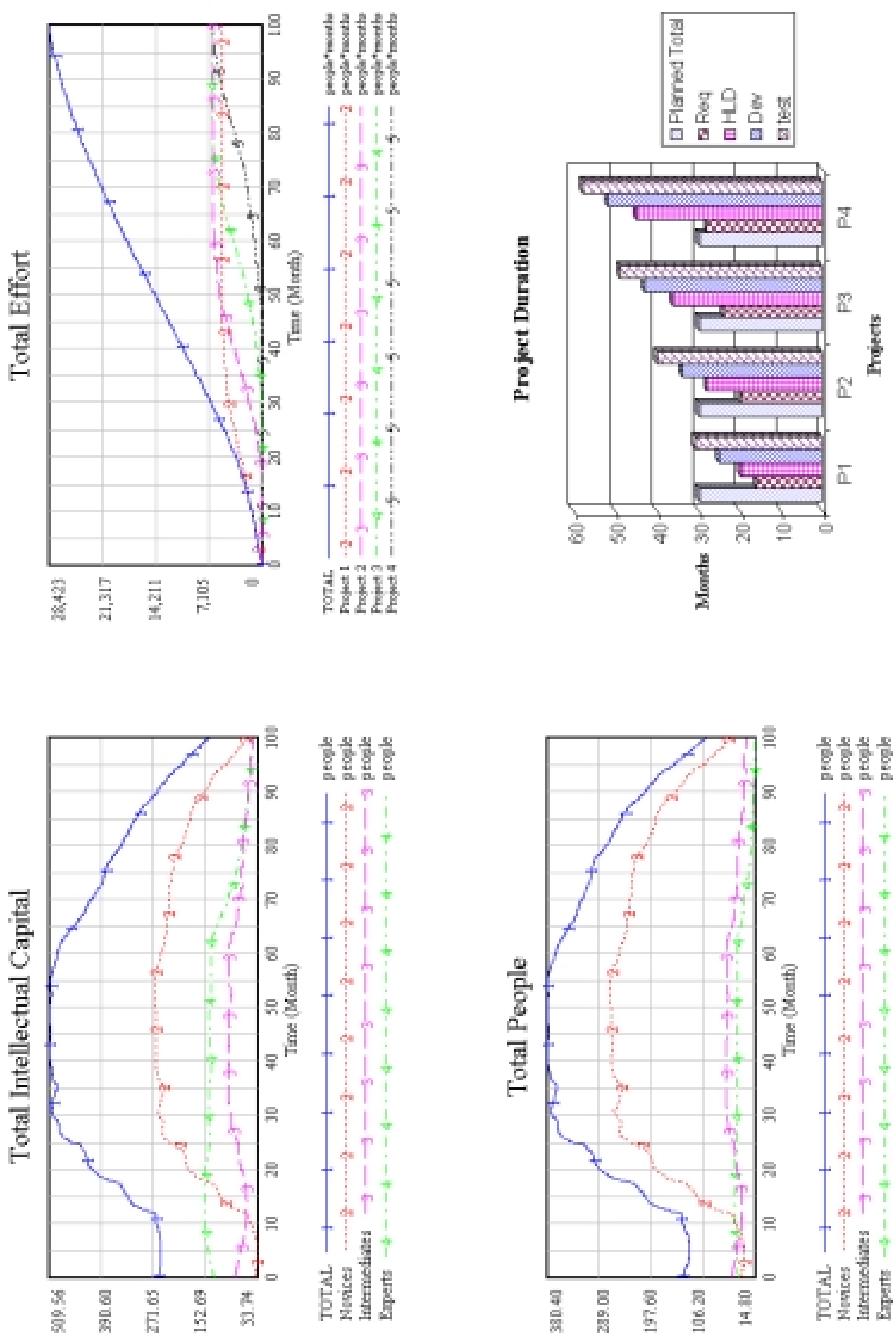


Figure 102 - Case #9b Simulation Results

Case 10

1. Case Description: Four Sequential Projects, Same Duration, Set Maximum Staff per Phase to Represent Constrained & Unconstrained Levels.

2. Rational for Use: Explore the Effect of a Constrained and Unconstrained Number of People on Project Completion Time.

3. Model Constants: Reference Vensim Input Sheet for Case 2.
Per Phase Staffing is Set at a Maximum of 100 in the Base Case Imperfect Quality Example, Case 2. Case 10A Constrains Staffing to, One-Half the Base Case 2 Amount, 50 Workers Per Phase. Case 10B Provides for Unconstrained Workforce by Setting the Maximum Staff to, Two Times that of Base Case 2, 200 Workers Per Phase.

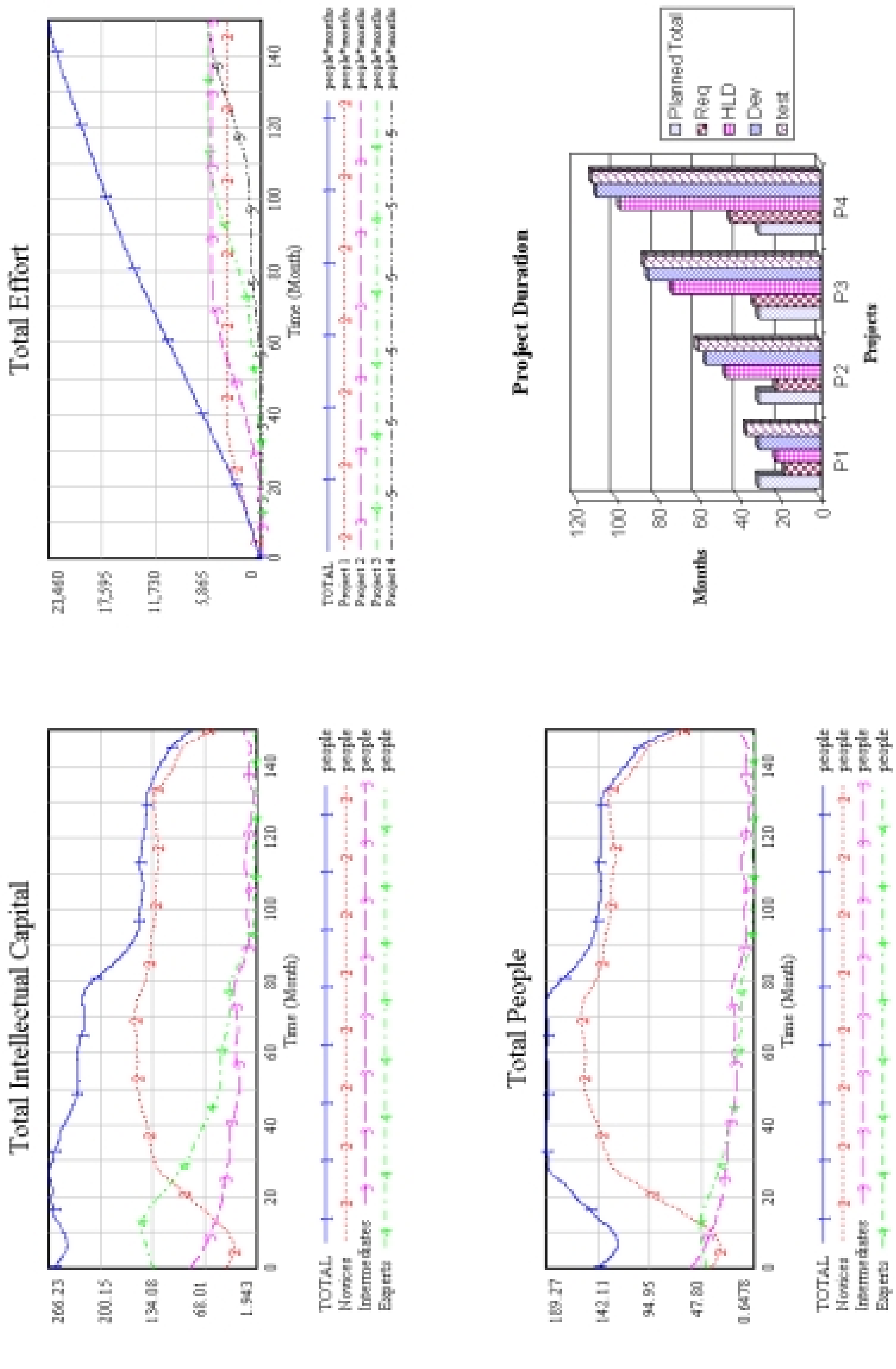


Figure 103 - Case #10a Simulation Results

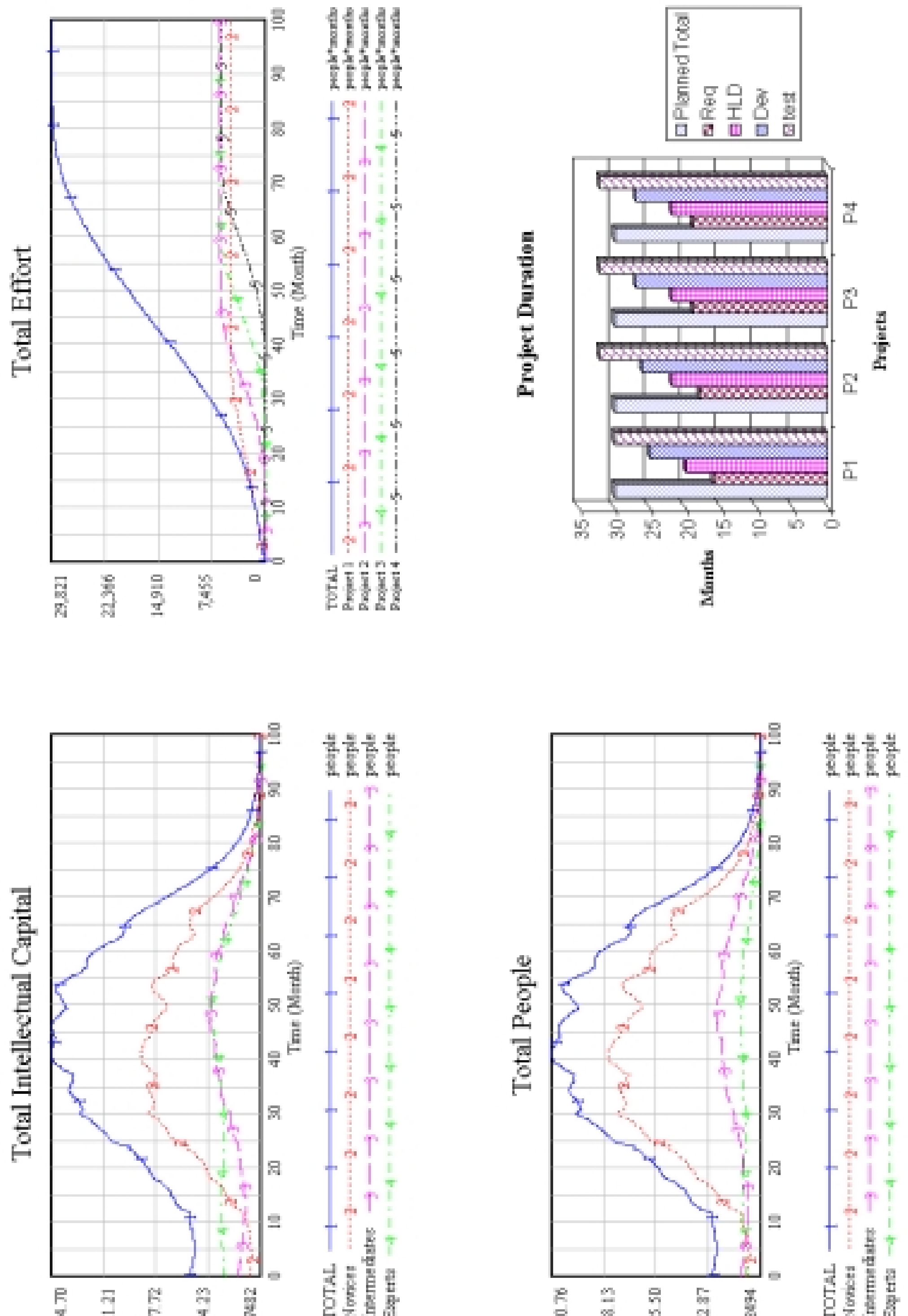


Figure 104 - Case #10b Simulation Results

Case 11

1. Case Description:	Four Sequential Projects, Same Duration, Adjust Time to Hire Two Cases - High and Low
2. Rational for Use:	Explore Ability of Organization to Meet Resource Demand
3. Model Constants:	Reference Vensim Input Sheet for Case 2. Base Case, Case 2 - Time to Hire: 2 Months Case 11A, Organization Very Responsive - Time to Hire: .5 Month Case 11B, Organization Un-Responsive - Time to Hire: 6 Months

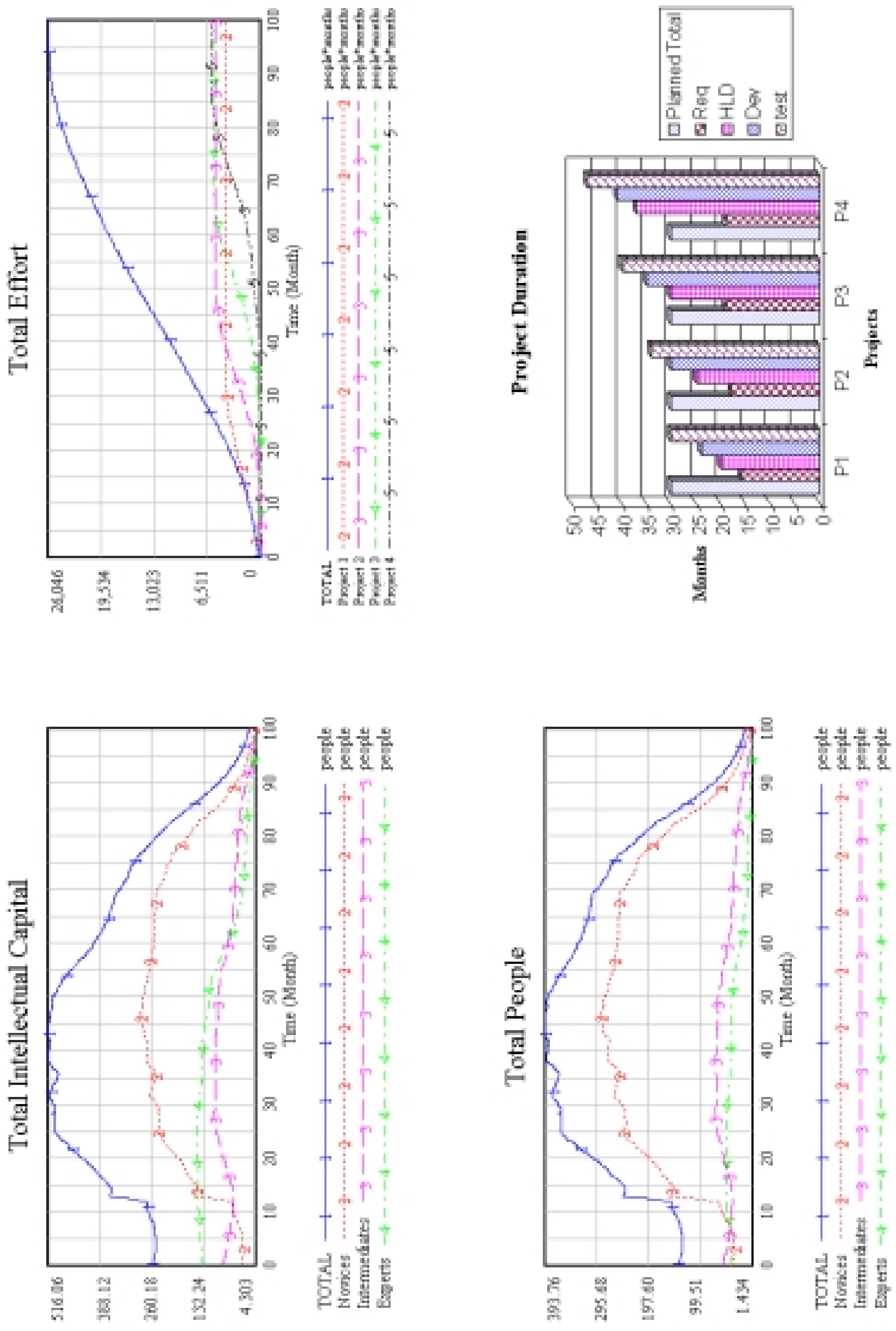


Figure 105 - Case 11a Simulation Results

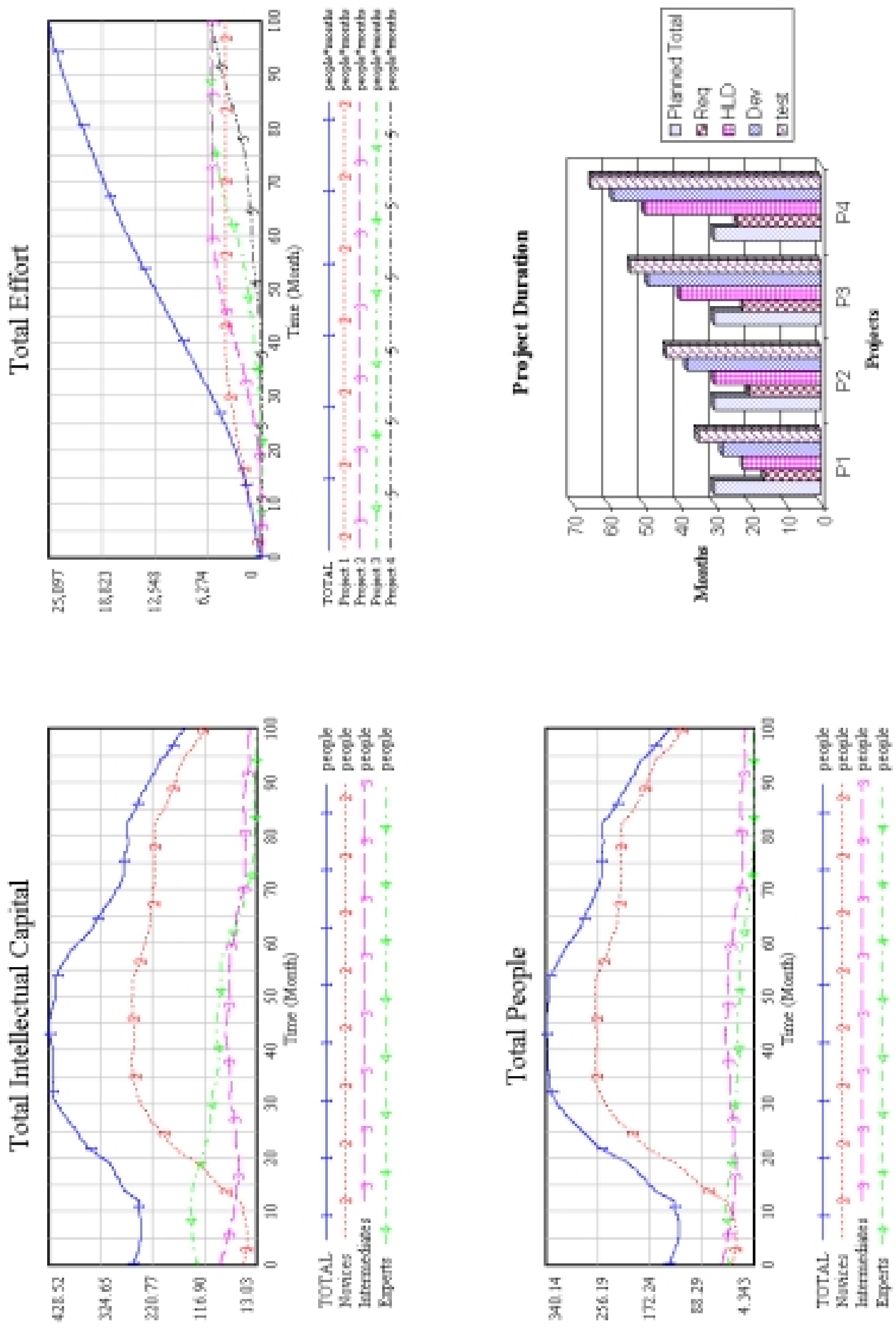


Figure 106 - Case #11b Simulation Results

Case 12

1. Case Description: Four Sequential Projects, Same Duration, Adjust Time to Downsize.
   Two Cases - High and Low

2. Rational for Use: Explore Organizational Tolerance to Unassigned Staff
   and Impact to Learning.
   Also, Explore Impact on Completing Rework as it is Discovered.

3. Model Constants: Reference Vensim Input Sheet for Case 2.
   Base Case, Case 2 - Time to Downsize: 6 Months
   Case 12A, Organization Very Intolerant,
   - Time to Downsize: 2 Months
   Case 12B, Organization Tolerant - Time to Downsize: 12 Months

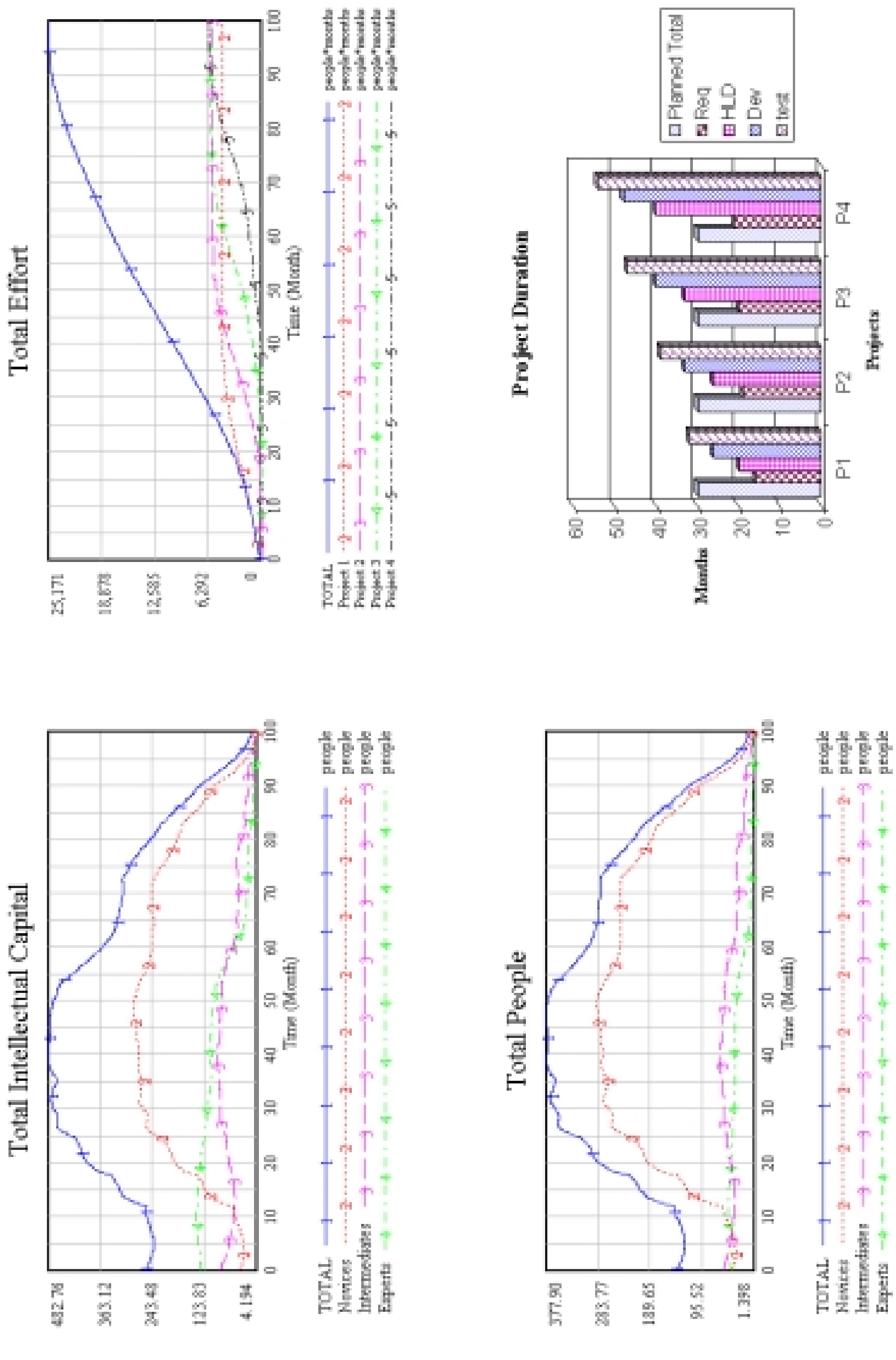


Figure 107 - Case #12a Simulation Results

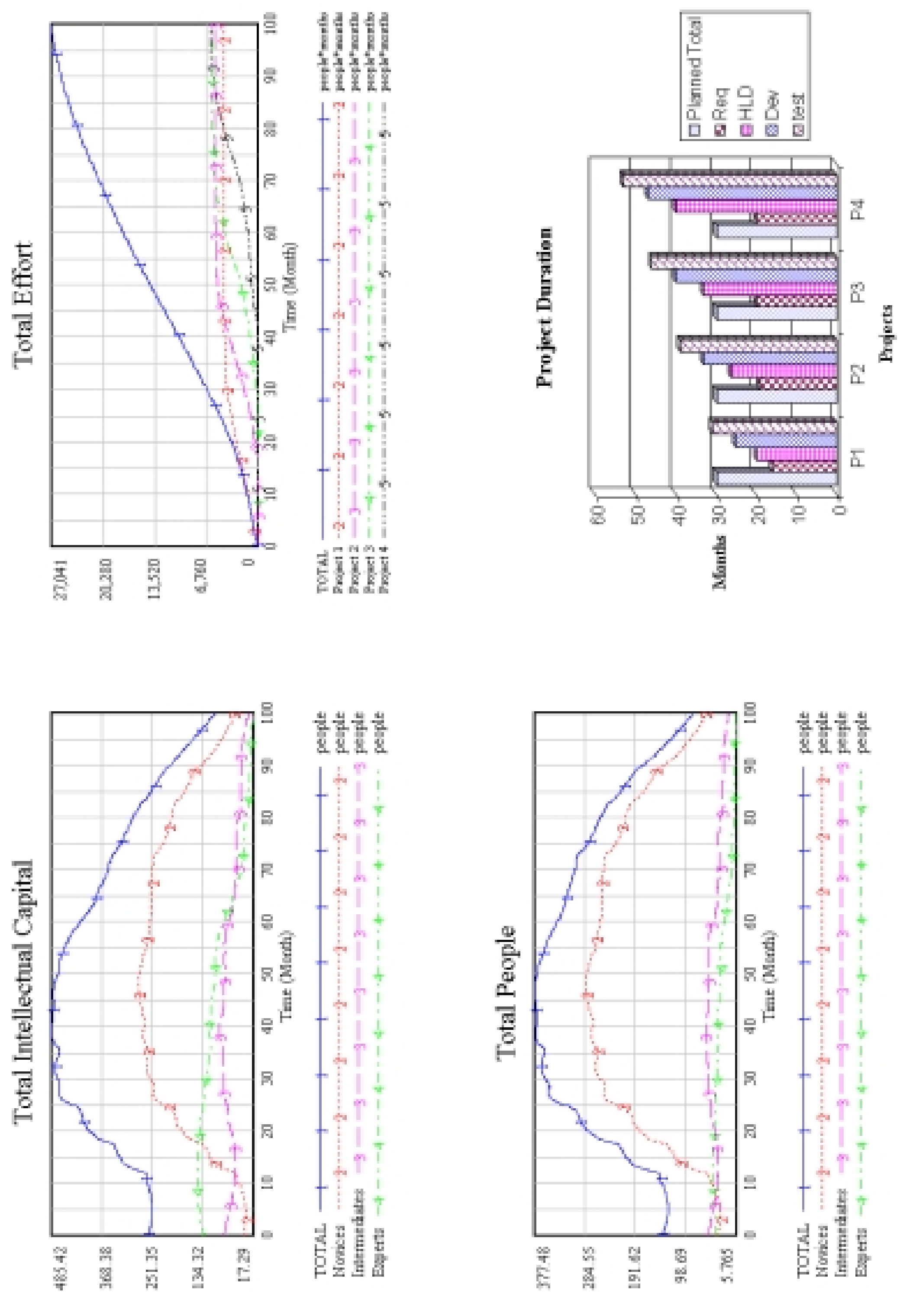


Figure 108 - Case #12b Simulation Results

Case 13

1. Case Description: Four Parallel Projects, Same Duration, Perfect Quality

2. Rational for Use: Base Case for Parallel Projects with Perfect Quality Representative of an Ideal World Where No Mistakes are Made by Employees.
Also, Explore the Effect of Project Concurrency on Firefighting.

3. Model Constants: Reference Vensim Input Sheet.
The Following Table Functions are Adjusted Such That There is No Effect from Function Output, Function Output Set to 1;

    Effect of Fatigue on Productivity
    Effect of Fatigue on Quality
    Complexity Effect on Productivity
    Complexity Effect on Quality

All Projects Have the Same Duration at the Start

All Projects Start with the Same Number of Staff Contrary to Sequential Project Staffing Where Only Project 1 was staffed at the Beginning.
Total Staff per Project Phase Does Not Exceed Maximum Staff Level Now Adjusted to a Level of 200 per Phase.

Projects Have an Intermediate Level of Complexity;

Project	Complexity
1	5
2	5
3	5
4	5

Figure 109 - Case #13 Initial Simulation Parameters

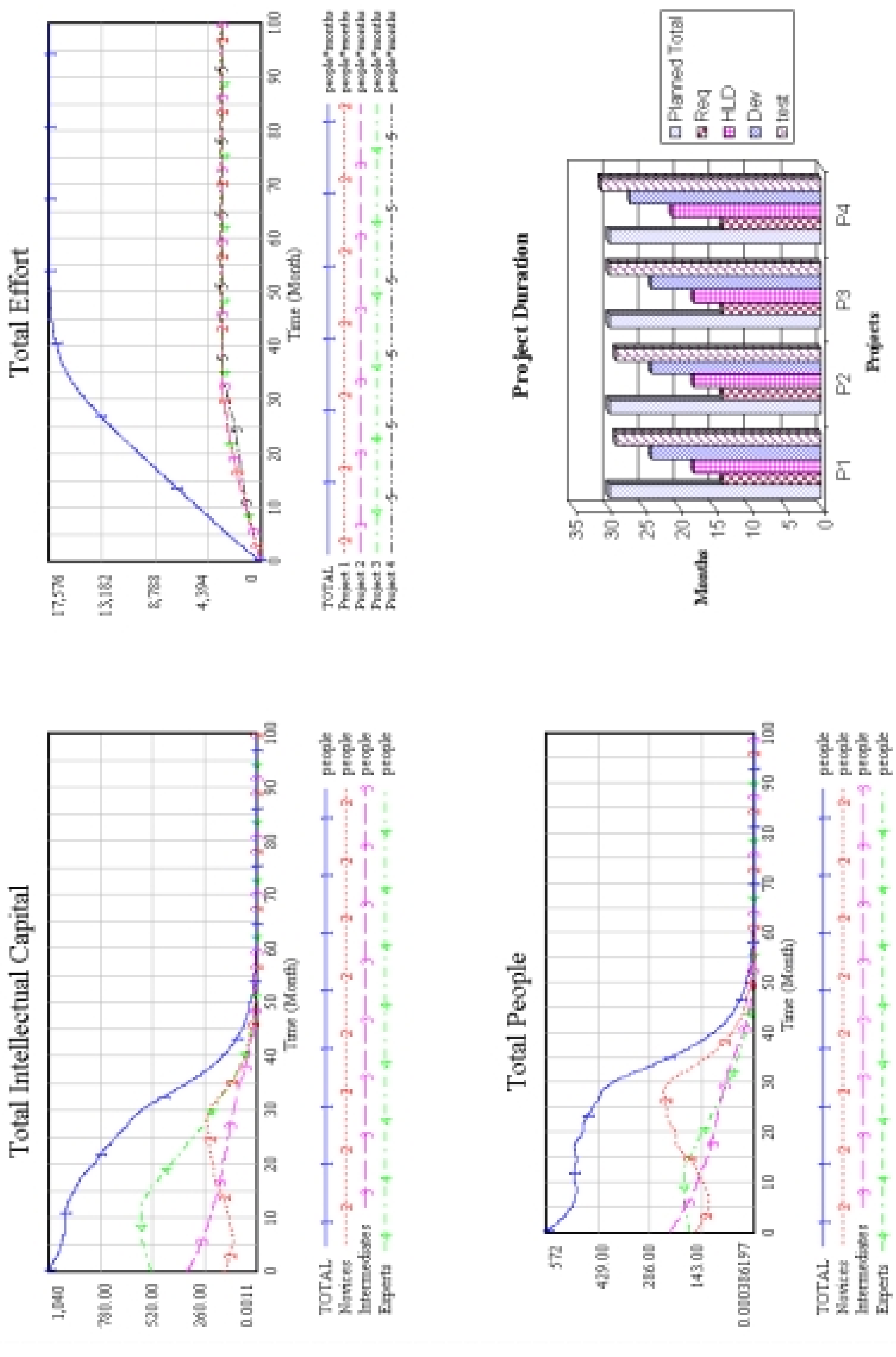


Figure 110 - Case #13 Simulation Results

Case 14

1. Case Description: Four Parallel Projects, Same Duration, Imperfect Quality

2. Rational for Use: Base Case for Parallel Projects with Imperfect Quality Representative of Real World Where Mistakes are Made by Employees.
Also, Explore the Effect of Project Concurrency on Firefighting.

3. Model Constants: Reference Vensim Input Sheet.
All Projects Have the Same Duration at the Start

All Projects Start with the Same Number of Staff Contrary to Sequential Project Staffing Where Only Project 1 was staffed at the Beginning.
Total Staff per Project Phase Does Not Exceed Maximum Staff Level.

Projects Have an Intermediate Level of Complexity;

Project	Complexity
1	5
2	5
3	5
4	5

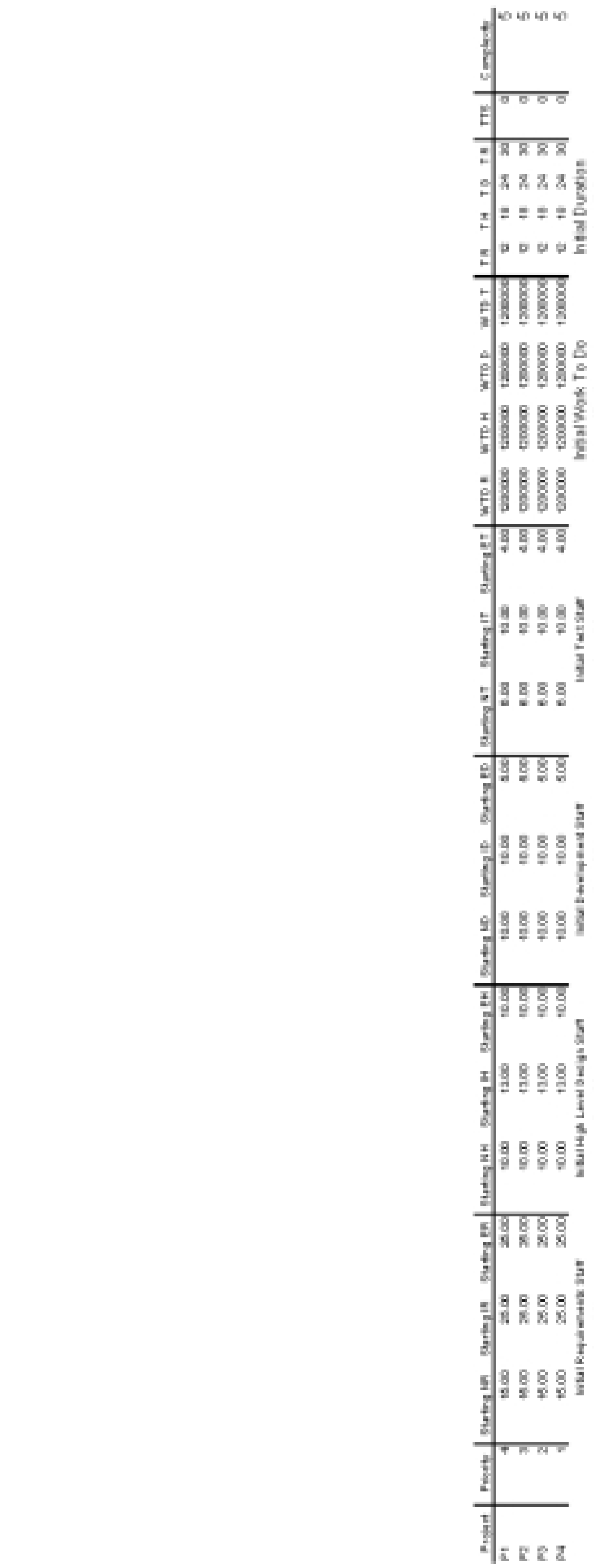


Figure 111 - Case #14 Initial Simulation Parameters

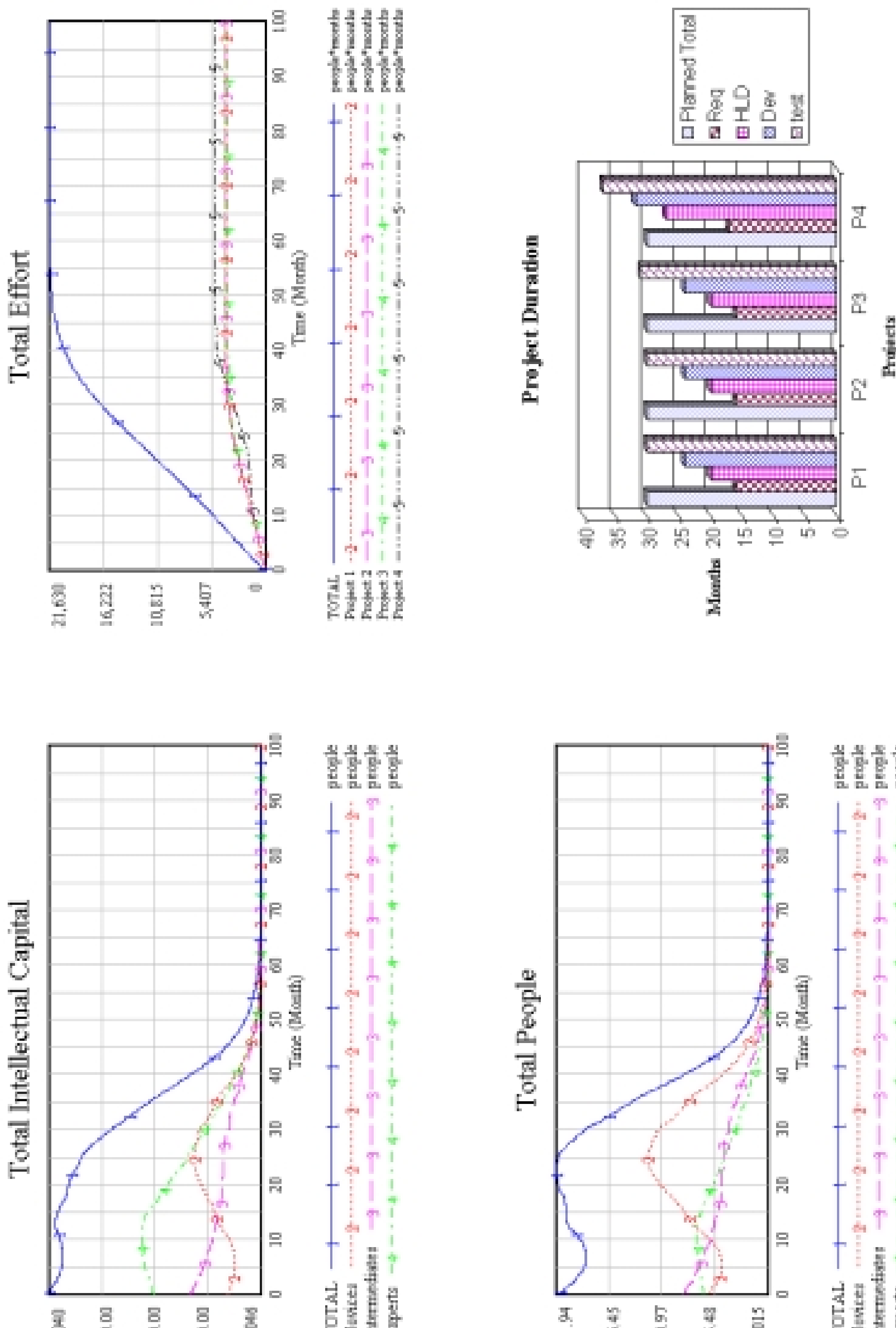


Figure 112 - Case #14 Simulation Results

Case 15

1. Case Description: Four Parallel Projects, Same Duration, All Complex

2. Rational for Use: Look at Extreme Case for Retaining & Developing Intellectual Capital.

3. Model Constants: Reference Vensim Input Sheet for Case 14.

All Projects Have the Same High Level of Complexity.

Project	Complexity
1	10
2	10
3	10
4	10

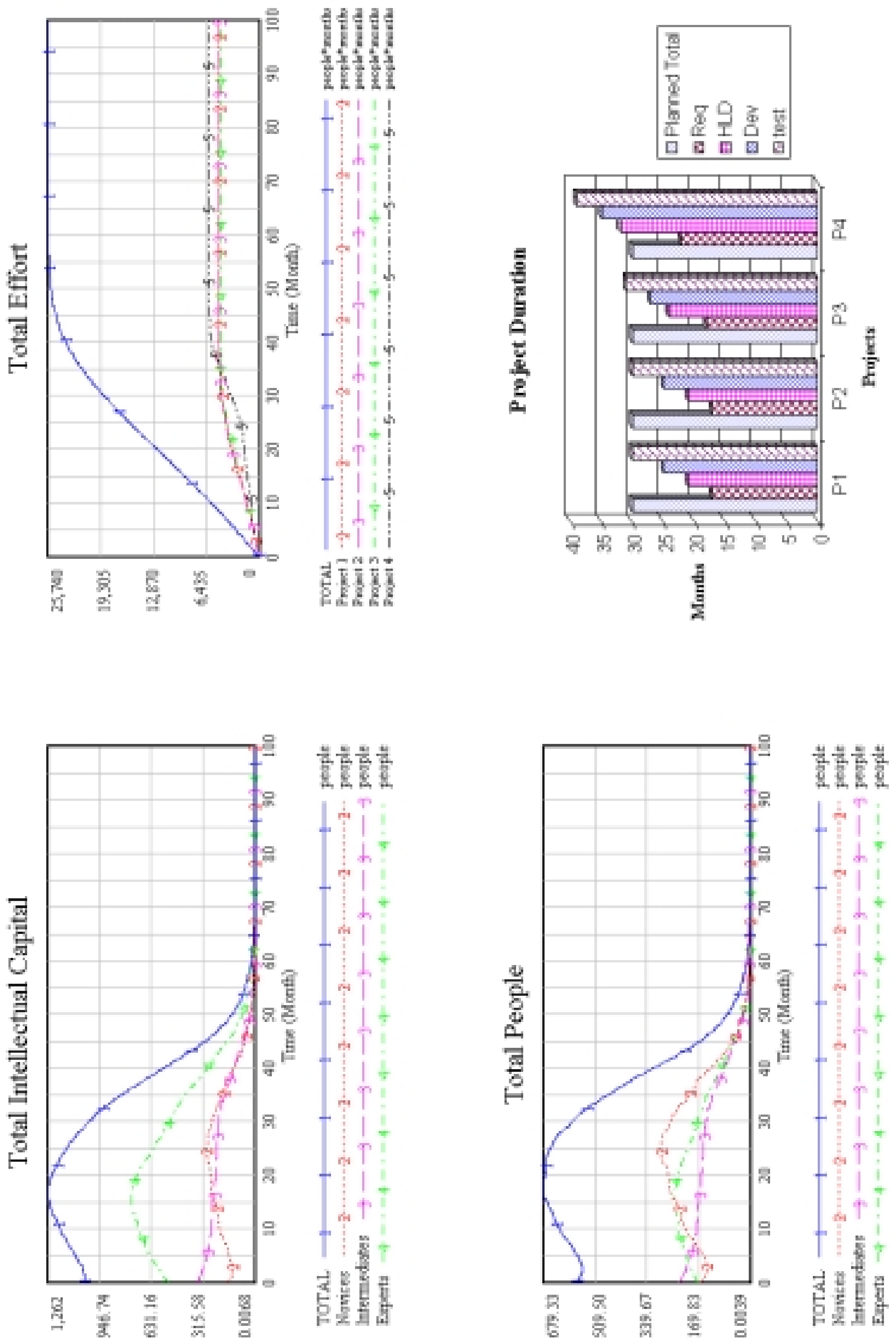


Figure 113 - Case #15 Simulation Results

Case 16

1. Case Description: Four Parallel Projects, Same Duration, Complexity; Simple, Complex, Simple, Complex

2. Rational for Use: Intermediate Complexity Case

3. Model Constants: Reference Vensim Input Sheet for Case 14. Projects Alternate Between Low Complexity and High Complexity.

Project	Complexity
1	1
2	10
3	1
4	10

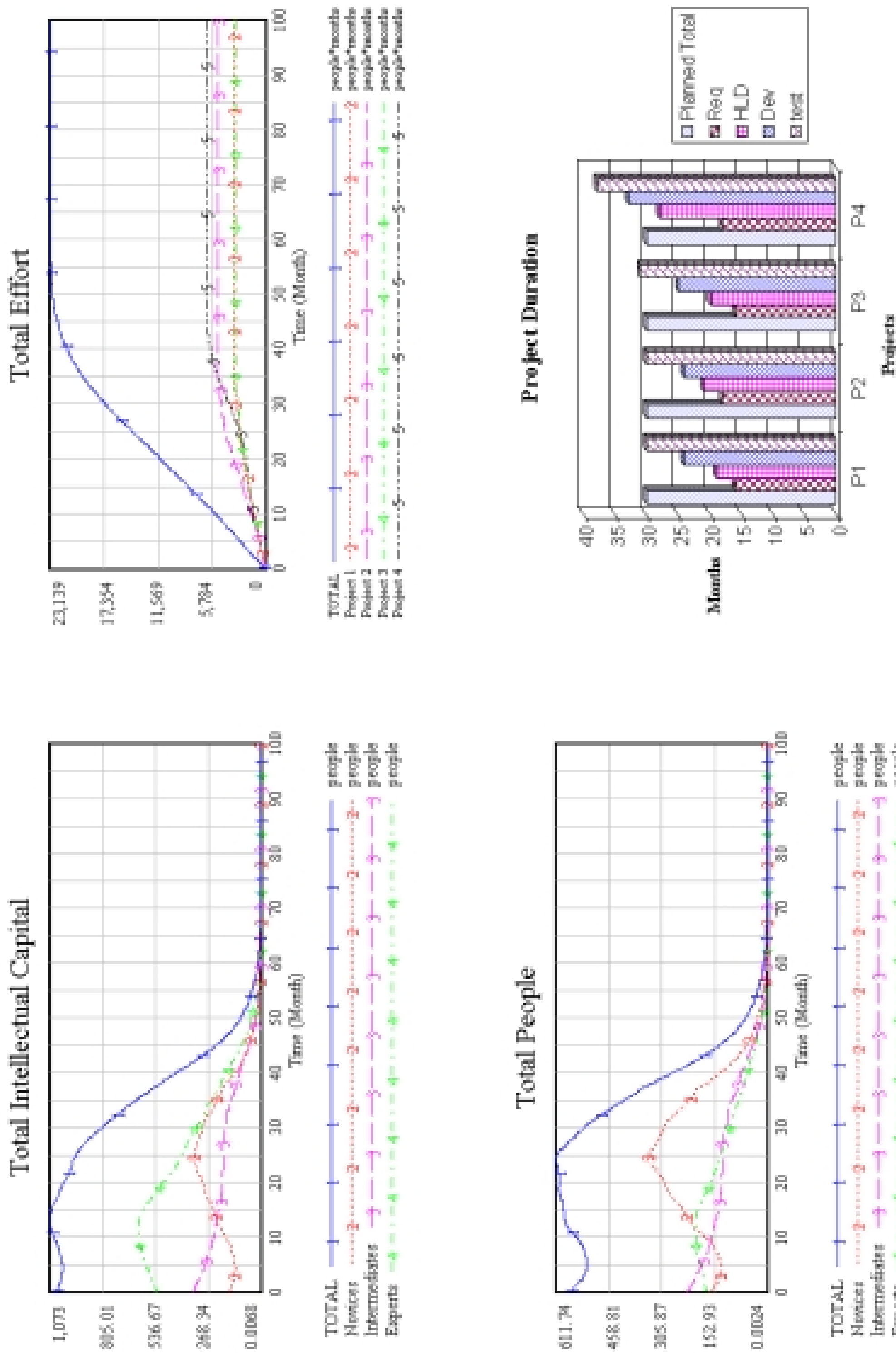


Figure 114 - Case #16 Simulation Results

Case 17

1. Case Description: Four Parallel Projects, Same Duration, Cancel Lowest Priority Project 3, Re-allocate Workers to Highest Priority Project 1.

2. Rational for Use: Test Impact of Reducing Competition for Resources by Eliminating a Low Priority Project and Re-allocating that Staff to the High Priority Project. This Creates an Overstaff Situation for the High Priority Project.

3. Model Constants: Reference Vensim Input Sheet for Case 14.
Projects are All Intermediate Level of Complexity.

Project	Complexity
1	5
2	5
3	5
4	5

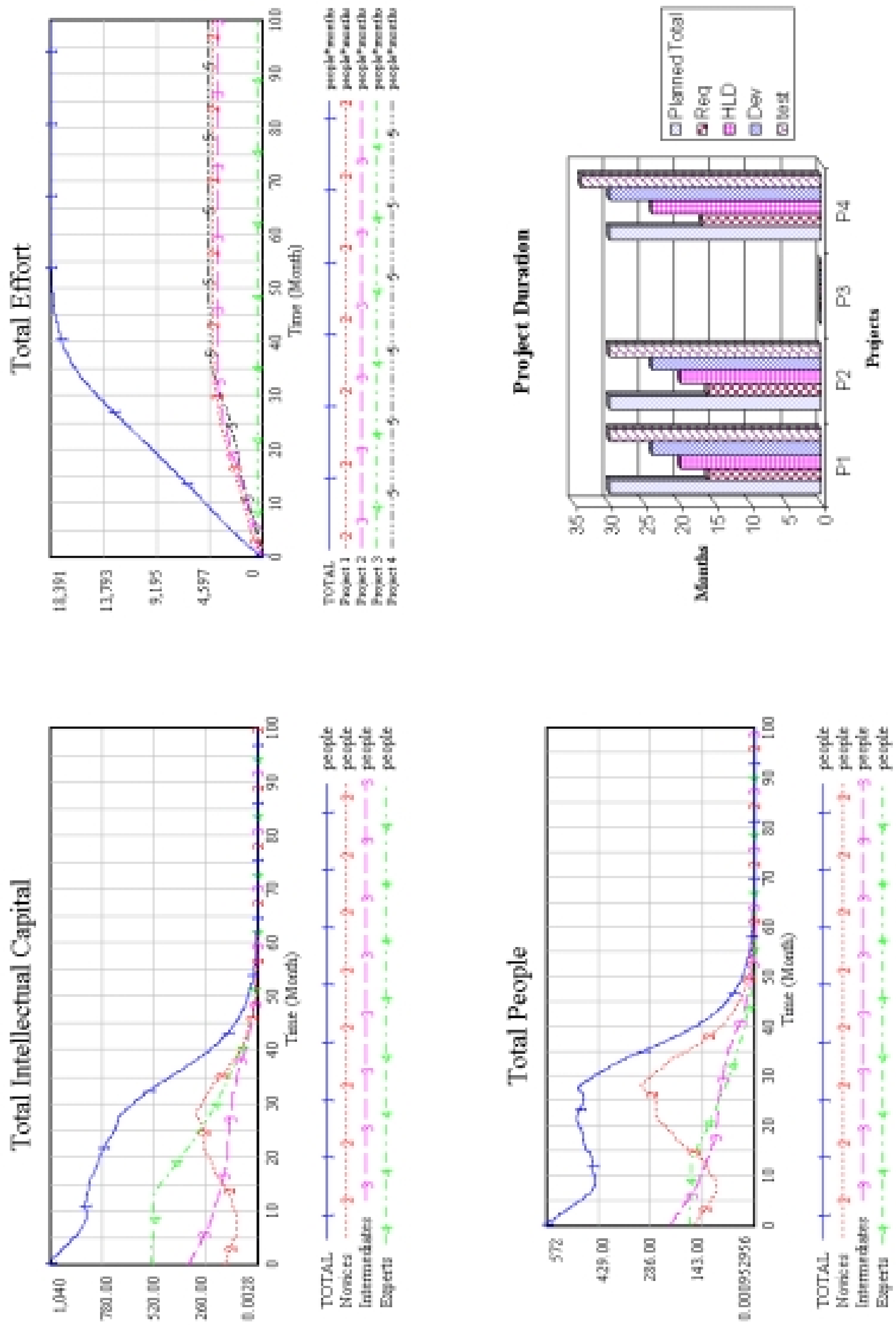


Figure 115 - Case #17 Simulation Results

Appendix B
Simulation Variables

```
Total IC=
      Total Novice IC+Total Intermediate IC+Total Expert IC
      ~     people
      ~           |

Total Intermediate IC=
      Total Intermediates*Base Intermediate Effectiveness
      ~     people
      ~           |

Total Expert IC=
      Total Experts*Base Expert Effectiveness
      ~     people
      ~           |

Total Novice IC=
      Total Novices*Base Novice Effectiveness
      ~     people
      ~           |

Total Control Rate=
      NR Control+IR Control+ER Control+NH Control+IH Control+EH
Control+ND Control+ID Control\
            +ED Control+NT Control+IT Control+ET Control
      ~     people
      ~           |

Total Control PersonMonths= INTEG (
      Total Control Rate,
            0)
      ~     people*months
      ~           |

Total P1 Rate=
      P1NR+P1IR+P1ER+P1NH+P1IH+P1EH+P1ND+P1ID+P1ED+P1NT+P1IT+P1ET
      ~     people
      ~           |

Total P2 PersonMonths= INTEG (
      Total P2 Rate,
            0)
      ~     people*months
      ~           |

Total P2 Rate=
      P2NR+P2IR+P2ER+P2NH+P2IH+P2EH+P2ND+P2ID+P2ED+P2NT+P2IT+P2ET
      ~     people
      ~           |

Total P3 PersonMonths= INTEG (
      Total P3 Rate,
            0)
      ~     people*months
      ~           |

Total P3 Rate=
```

```
	P3NR+P3IR+P3ER+P3NH+P3IH+P3EH+P3ND+P3ID+P3ED+P3NT+P3IT+P3ET
	~	people
	~		|

Total P4 personMonths= INTEG (
	Total P4 Rate,
		0)
	~	people*months
	~		|

Total P1 PersonMonths= INTEG (
	Total P1 Rate,
		0)
	~	people*months
	~		|

Total P4 Rate=
	P4NR+P4IR+P4ER+P4NH+P4IH+P4EH+P4ND+P4ID+P4ED+P4NT+P4IT+P4ET
	~	people
	~		|

Total Novices=
	TotalNR + TotalNH + TotalND + TotalNT
	~	people
	~		|

Total Intermediates=
	TotalIR + TotalIH + TotalID + TotalIT
	~	people
	~		|

Total Experts=
	TotalER + TotalEH + TotalED + TotalET
	~	people
	~		|

Total People=
	Total Novices+Total Intermediates+Total Experts
	~	people
	~		|

Complexity Effect on Attrition T f(
	[(0,0)-(10,10)],(0,1.2),(5,1),(10,0.7))
	~	fraction
	~		|

Complexity Effect on Attrition H f(
	[(0,0)-(10,10)],(0,1.2),(5,1),(10,0.7))
	~	fraction
	~		|

Complexity Effect on Attrition D f(
	[(0,0)-(10,10)],(0,1.2),(5,1),(10,0.7))
	~	fraction
	~		|

Complexity Effect on Attrition R f(
	[(0,0)-(10,10)],(0,1.2),(5,1),(10,0.7))
	~	fraction
	~		|
```

```
Total PersonMonths R=
      TotalNR Months + TotalIR Months + TotalER Months
      ~     people*months
      ~           |

Total PersonMonths T=
      TotalNT Months + TotalIT Months + TotalET Months
      ~     people*months
      ~           |

Total PersonMonths D=
      TotalND Months + TotalID Months + TotalED Months
      ~     people*months
      ~           |

Total PersonMonths=
      Total PersonMonths D+Total PersonMonths H+Total PersonMonths
R+Total PersonMonths T
      ~     people*months
      ~           |

Total PersonMonths H=
      TotalNH Months + TotalIH Months + TotalEH Months
      ~     people*months
      ~           |

TotalED Rate=
      TotalED
      ~     people
      ~           |

TotalNR Months= INTEG (
      TotalNR Rate,
            0)
      ~     people*months
      ~           |

TotalER Months= INTEG (
      TotalER Rate,
            0)
      ~     people*months
      ~           |

TotalNR Rate=
      TotalNR
      ~     people
      ~           |

TotalID Months= INTEG (
      TotalID Rate,
            0)
      ~     people*months
      ~           |

TotalNT Months= INTEG (
      TotalNT Rate,
            0)
      ~     people*months
      ~           |

TotalNT Rate=
```

```
	TotalNT
	~	people
	~		|

TotalIH Months= INTEG (
	TotalIH Rate,
		0)
	~	people*months
	~		|

TotalIH Rate=
	TotalIH
	~	people
	~		|

TotalED Months= INTEG (
	TotalED Rate,
		0)
	~	people*months
	~		|

TotalIR Rate=
	TotalIR
	~	people
	~		|

TotalND Rate=
	TotalND
	~	people
	~		|

TotalEH Months= INTEG (
	TotalEH Rate,
		0)
	~	people*months
	~		|

TotalEH Rate=
	TotalEH
	~	people
	~		|

TotalNH Rate=
	TotalNH
	~	people
	~		|

TotalER Rate=
	TotalER
	~	people
	~		|

TotalET Rate=
	TotalET
	~	people
	~		|

TotalET Months= INTEG (
	TotalET Rate,
		0)
	~	people*months
```

```
	~		|

TotalID Rate=
	TotalID
	~	people
	~		|

TotalIR Months= INTEG (
	TotalIR Rate,
		0)
	~	people*months
	~		|

PDY P1H=
	Complexity effect on PDY P1*Fatigue effect PDY P1H*Normal
Productivity H
	~	lines/(people*Month)
	~		|

TotalIT Rate=
	TotalIT
	~	people
	~		|

TotalNH Months= INTEG (
	TotalNH Rate,
		0)
	~	people*months
	~		|

TotalND Months= INTEG (
	TotalND Rate,
		0)
	~	people*months
	~		|

TotalIT Months= INTEG (
	TotalIT Rate,
		0)
	~	people*months
	~		|

IntMultiplier P1H=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P1H)
	~	fraction
	~		|

IntMultiplier P1T=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P1T)
	~	fraction
	~		|

IntMultiplier P2D=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P2D)
	~	fraction
	~		|

IntMultiplier P2H=
```

```
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P2H)
	~	fraction
	~		|

IntMultiplier P2R=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P2R)
	~	fraction
	~		|

IntMultiplier P2T=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P2T)
	~	fraction
	~		|

IntMultiplier P3D=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P3D)
	~	fraction
	~		|

IntMultiplier P3H=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P3H)
	~	fraction
	~		|

IntMultiplier P3R=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P3R)
	~	fraction
	~		|

IntMultiplier P3T=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P3T)
	~	fraction
	~		|

IntMultiplier P4D=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P4D)
	~	fraction
	~		|

IntMultiplier P4H=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P4H)
	~	fraction
	~		|

IntMultiplier P4R=
	Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P4R)
	~	fraction
	~		|

IntMultiplier P4T=
```

```
      Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P4T)
      ~     fraction
      ~           |

ExpertMultiplier P4D=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P4D)
      ~     fraction
      ~           |

NoviceMultiplier P1D=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P1H=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P3T=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P1T=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P2D=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P2H=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P2R=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P2T=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P3D=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P3H=
      Base Novice Effectiveness
      ~     fraction
      ~           |
```

```
NoviceMultiplier P3R=
      Base Novice Effectiveness
      ~     fraction
      ~           |

ExpertMultiplier P1T=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P1T)
      ~     fraction
      ~           |

NoviceMultiplier P4D=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P4H=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P4R=
      Base Novice Effectiveness
      ~     fraction
      ~           |

NoviceMultiplier P4T=
      Base Novice Effectiveness
      ~     fraction
      ~           |

ExpertMultiplier P4R=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P4R)
      ~     fraction
      ~           |

ExpertMultiplier P4H=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P4H)
      ~     fraction
      ~           |

ExpertMultiplier P3T=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P3T)
      ~     fraction
      ~           |

ExpertMultiplier P4T=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P4T)
      ~     fraction
      ~           |

ExpertMultiplier P2T=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P2T)
      ~     fraction
      ~           |
```

```
ExpertMultiplier P2R=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P2R)
      ~     fraction
      ~           |

ExpertMultiplier P2H=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P2H)
      ~     fraction
      ~           |

IntMultiplier P1D=
      Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P1D)
      ~     fraction
      ~           |

ExpertMultiplier P1D=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P1D)
      ~     fraction
      ~           |

ExpertMultiplier P3R=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P3R)
      ~     fraction
      ~           |

ExpertMultiplier P3H=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P3H)
      ~     fraction
      ~           |

ExpertMultiplier P1H=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P1H)
      ~     fraction
      ~           |

ExpertMultiplier P2D=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P2D)
      ~     fraction
      ~           |

ExpertMultiplier P3D=
      Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P3D)
      ~     fraction
      ~           |

IN Ratio P1H=
      ZIDZ(P1IH,P1NH)
      ~     fraction
      ~           |

IN Ratio P1R=
      ZIDZ(P1IR,P1NR)
```

```
        ~     fraction
        ~           |

IN Ratio P1T=
        ZIDZ(P1IT,P1NT)
        ~     fraction
        ~           |

IN Ratio P2D=
        ZIDZ(P2ID,P2ND)
        ~     fraction
        ~           |

IN Ratio P2H=
        ZIDZ(P2IH,P2NH)
        ~     fraction
        ~           |

IN Ratio P2R=
        ZIDZ(P2IR,P2NR)
        ~     fraction
        ~           |

IN Ratio P2T=
        ZIDZ(P2IT,P2NT)
        ~     fraction
        ~           |

IN Ratio P3D=
        ZIDZ(P3ID,P3ND)
        ~     fraction
        ~           |

IN Ratio P3H=
        ZIDZ(P3IH,P3NH)
        ~     fraction
        ~           |

IN Ratio P3R=
        ZIDZ(P3IR,P3NR)
        ~     fraction
        ~           |

IN Ratio P3T=
        ZIDZ(P3IT,P3NT)
        ~     fraction
        ~           |

IN Ratio P4D=
        ZIDZ(P4ID,P4ND)
        ~     fraction
        ~           |

IN Ratio P4H=
        ZIDZ(P4IH,P4NH)
        ~     fraction
        ~           |

IN Ratio P4R=
        ZIDZ(P4IR,P4NR)
        ~     fraction
```

```
	~		|

IN Ratio P4T=
	ZIDZ(P4IT,P4NT)
	~	fraction
	~		|

EI Ratio P4R=
	ZIDZ(P4ER,P4IR)
	~	fraction
	~		|

EI Ratio P4H=
	ZIDZ(P4EH,P4IH)
	~	fraction
	~		|

EI Ratio P4D=
	ZIDZ(P4ED,P4ID)
	~	fraction
	~		|

EI Ratio P4T=
	ZIDZ(P4ET,P4IT)
	~	fraction
	~		|

EI Ratio P1T=
	ZIDZ(P1ET,P1IT)
	~	fraction
	~		|

EI Ratio P2T=
	ZIDZ(P2ET,P2IT)
	~	fraction
	~		|

EI Ratio P3R=
	ZIDZ(P3ER,P3IR)
	~	fraction
	~		|

EI Ratio effect on effectiveness f(
	[(0,0)-
(1e+010,10)],(0,0),(0.01,0.1),(0.05,0.5),(0.1,0.9),(0.25,1),(0.5,1),(1,1
),(10\
		,1),(100,1),(1e+010,1))
	~	fraction
	~		|

EI Ratio P1D=
	ZIDZ(P1ED,P1ID)
	~	fraction
	~		|

EI Ratio P1H=
	ZIDZ(P1EH,P1IH)
	~	fraction
	~		|

EI Ratio P1R=
```

```
	ZIDZ(P1ER,P1IR)
	~	fraction
	~		|

EI Ratio P2R=
	ZIDZ(P2ER,P2IR)
	~	fraction
	~		|

EI Ratio P2H=
	ZIDZ(P2EH,P2IH)
	~	fraction
	~		|

EI Ratio P2D=
	ZIDZ(P2ED,P2ID)
	~	fraction
	~		|

IN Ratio P1D=
	ZIDZ(P1ID,P1ND)
	~	fraction
	~		|

Base Novice Effectiveness=
	Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b17')
	~	fraction
	~		|

NoviceMultiplier P1R=
	Base Novice Effectiveness
	~	fraction
	~		|

EI Ratio P3D=
	ZIDZ(P3ED,P3ID)
	~	fraction
	~		|

EI Ratio P3T=
	ZIDZ(P3ET,P3IT)
	~	fraction
	~		|

ExpertMultiplier P1R=
	Base Expert Effectiveness*EI Ratio effect on effectiveness f(EI
Ratio P1R)
	~	fraction
	~		|

IN Ratio effect on effectiveness f(
	[(0,0)-
(100,10)],(0,0),(0.01,0.1),(0.05,0.5),(0.1,0.9),(0.25,1),(0.5,1),(1,1),(
10,1)\
		,(100,1))
	~	fraction
	~		|

Base Expert Effectiveness=
```

```
      Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b19')
      ~     fraction
      ~           |

IntMultiplier P1R=
      Base Intermediate Effectiveness*IN Ratio effect on effectiveness
f(IN Ratio P1R)
      ~     fraction
      ~           |

EI Ratio P3H=
      ZIDZ(P3EH,P3IH)
      ~     fraction
      ~           |

Base Intermediate Effectiveness=
      Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b18')
      ~     fraction
      ~           |

TotalP2 H=
      P2NH+P2IH+P2EH
      ~     people
      ~           |

AverageWorkerWeight P2H=
      (RatioNH*NoviceMultiplier P2H+RatioIH*IntMultiplier
P2H+RatioEH*ExpertMultiplier P2H\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P2R=
      (RatioNR*NoviceMultiplier P2R+RatioIR*IntMultiplier
P2R+RatioER*ExpertMultiplier P2R\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P2T=
      (RatioNT*NoviceMultiplier P2T+RatioIT*IntMultiplier
P2T+RatioET*ExpertMultiplier P2T\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P3D=
      (RatioND*NoviceMultiplier P3D+RatioID*IntMultiplier
P3D+RatioED*ExpertMultiplier P3D\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P3H=
      (RatioNH*NoviceMultiplier P3H+RatioIH*IntMultiplier
P3H+RatioEH*ExpertMultiplier P3H\
            )
      ~     fraction
      ~           |
```

```
AverageWorkerWeight P3R=
      (RatioNR*NoviceMultiplier P3R+RatioIR*IntMultiplier
P3R+RatioER*ExpertMultiplier P3R\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P3T=
      (RatioNT*NoviceMultiplier P3T+RatioIT*IntMultiplier
P3T+RatioET*ExpertMultiplier P3T\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P4D=
      (RatioND*NoviceMultiplier P4D+RatioID*IntMultiplier
P4D+RatioED*ExpertMultiplier P4D\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P4H=
      (RatioNH*NoviceMultiplier P4H+RatioIH*IntMultiplier
P4H+RatioEH*ExpertMultiplier P4H\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P4R=
      (RatioNR*NoviceMultiplier P4R+RatioIR*IntMultiplier
P4R+RatioER*ExpertMultiplier P4R\
            )
      ~     fraction
      ~           |

AverageWorkerWeight P4T=
      (RatioNT*NoviceMultiplier P4T+RatioIT*IntMultiplier
P4T+RatioET*ExpertMultiplier P4T\
            )
      ~     fraction
      ~           |

DesiredRealHeads P1D=
      DesiredPeople P1D/AverageWorkerWeight P1D
      ~     people
      ~           |

DesiredRealHeads P1H=
      DesiredPeople P1H/AverageWorkerWeight P1H
      ~     people
      ~           |

RatioED=
      ZIDZ(TotalED,(TotalND + TotalID + TotalED))
      ~     fraction
      ~           |

RatioEH=
      ZIDZ(TotalEH,(TotalNH + TotalIH + TotalEH))
      ~     fraction
```

```
      ~           |

RatioER=
      ZIDZ(TotalER,(TotalNR + TotalIR + TotalER))
      ~     fraction
      ~           |

RatioET=
      ZIDZ(TotalET,(TotalNT + TotalIT + TotalET))
      ~     fraction
      ~           |

RatioID=
      ZIDZ(TotalID,(TotalND + TotalID + TotalED))
      ~     fraction
      ~           |

RatioIH=
      ZIDZ(TotalIH,(TotalNH + TotalIH + TotalEH))
      ~     fraction
      ~           |

RatioIR=
      ZIDZ(TotalIR,(TotalNR + TotalIR + TotalER))
      ~     fraction
      ~           |

RatioIT=
      ZIDZ(TotalIT,(TotalIT + TotalIT + TotalET))
      ~     fraction
      ~           |

RatioND=
      ZIDZ(TotalND,(TotalND + TotalID + TotalED))
      ~     fraction
      ~           |

RatioNH=
      ZIDZ(TotalNH,(TotalNH + TotalIH + TotalEH))
      ~     fraction
      ~           |

RatioNR=
      ZIDZ(TotalNR,(TotalNR + TotalIR + TotalER))
      ~     fraction
      ~           |

RatioNT=
      ZIDZ(TotalNT,(TotalNT + TotalIT + TotalET))
      ~     fraction
      ~           |

DesiredRealHeads P4R=
      DesiredPeople P4R/AverageWorkerWeight P4R
      ~     people
      ~           |

DesiredRealHeads P4T=
      DesiredPeople P4T/AverageWorkerWeight P4T
      ~     people
      ~           |
```

```
AverageWorkerWeight P2D=
	(RatioND*NoviceMultiplier P2D+RatioID*IntMultiplier
P2D+RatioED*ExpertMultiplier P2D\
		)
	~	fraction
	~		|

DesiredRealHeads P2T=
	DesiredPeople P2T/AverageWorkerWeight P2T
	~	people
	~		|

DesiredRealHeads P3D=
	DesiredPeople P3D/AverageWorkerWeight P3D
	~	people
	~		|

DesiredRealHeads P3H=
	DesiredPeople P3H/AverageWorkerWeight P3H
	~	people
	~		|

P1RDesiredE=
	DesiredRealHeads P1R*RatioER
	~	people
	~		|

P1RDesiredI=
	DesiredRealHeads P1R*RatioIR
	~	people
	~		|

P1RDesiredN=
	DesiredRealHeads P1R*RatioNR
	~	people
	~		|

AverageWorkerWeight P1R=
	(RatioNR*NoviceMultiplier P1R+RatioIR*IntMultiplier
P1R+RatioER*ExpertMultiplier P1R\
		)
	~	fraction
	~		|

AverageWorkerWeight P1T=
	(RatioNT*NoviceMultiplier P1T+RatioIT*IntMultiplier
P1T+RatioET*ExpertMultiplier P1T\
		)
	~	fraction
	~		|

DesiredRealHeads P2R=
	DesiredPeople P2R/AverageWorkerWeight P2R
	~	people
	~		|

DesiredRealHeads P3T=
	DesiredPeople P3T/AverageWorkerWeight P3T
	~	people
	~		|
```

```
DesiredRealHeads P4D=
      DesiredPeople P4D/AverageWorkerWeight P4D
      ~     people
      ~           |

DesiredRealHeads P4H=
      DesiredPeople P4H/AverageWorkerWeight P4H
      ~     people
      ~           |

DesiredRealHeads P3R=
      DesiredPeople P3R/AverageWorkerWeight P3R
      ~     people
      ~           |

DesiredPeople P1H=
      ((WorkToDo P1H/Remaining Time P1H)/Percvd PDY P1H+((Initial
WorkToDo P1H/Remaining Time P1H\
            )/Percvd PDY P1H)*0.75*Active P1H)
      ~     people
      ~           |

DesiredRealHeads P1R=
      DesiredPeople P1R/AverageWorkerWeight P1R
      ~     people
      ~           |

DesiredRealHeads P1T=
      DesiredPeople P1T/AverageWorkerWeight P1T
      ~     people
      ~           |

DesiredRealHeads P2H=
      DesiredPeople P2H/AverageWorkerWeight P2H
      ~     people
      ~           |

AverageWorkerWeight P1H=
      (RatioNH*NoviceMultiplier P1H+RatioIH*IntMultiplier
P1H+RatioEH*ExpertMultiplier P1H\
            )
      ~     fraction
      ~           |

DesiredRealHeads P2D=
      DesiredPeople P2D/AverageWorkerWeight P2D
      ~     people
      ~           |

AverageWorkerWeight P1D=
      (RatioND*NoviceMultiplier P1D+RatioID*IntMultiplier
P1D+RatioED*ExpertMultiplier P1D\
            )
      ~     fraction
      ~           |

Total H=
      TotalNH + TotalIH + TotalEH
      ~     people
      ~           |
```

```
Total R=
      TotalNR + TotalIR + TotalER
      ~     people
      ~           |

Total T=
      TotalNT + TotalIT + TotalET
      ~     people
      ~           |

Total Desired R=
      SUM(NRDesired[project!])+SUM(IRDesired[project!])+SUM(ERDesired[pr
oject!])
      ~     people
      ~           |

Total D=
      TotalND + TotalID + TotalED
      ~     people
      ~           |

Total Desired D=
      SUM(NDDesired[project!])+SUM(IDDesired[project!])+SUM(EDDesired[pr
oject!])
      ~     people
      ~           |

Total Desired H=
      SUM(NHDesired[project!])+SUM(IHDesired[project!])+SUM(EHDesired[pr
oject!])
      ~     people
      ~           |

Total Desired T=
      SUM(NTDesired[project!])+SUM(ITDesired[project!])+SUM(ETDesired[pr
oject!])
      ~     people
      ~           |

Maximum Staff D=
      Get XLS Constants('ModelConstants.xls','Phase Constants','d10')
      ~     people
      ~           |

Maximum Staff H=
      Get XLS Constants('ModelConstants.xls','Phase Constants','c10')
      ~     people
      ~           |

Maximum Staff R=
      Get XLS Constants('ModelConstants.xls','Phase Constants','b10')
      ~     people
      ~           |

Maximum Staff T=
      Get XLS Constants('ModelConstants.xls','Phase Constants','e10')
      ~     people
      ~           |

NHHireRate=
```

```
	if then else(Novices to Hire H > 0, Novices to Hire H/(Time to
hire/Gap Effect on Hiring H f\
		(GapRatio H)),0)
	~	people/Month
	~		|

Novices to Hire H=
	Min((Maximum Staff H-TotalNH-TotalIH-
TotalEH),(SUM(NHDesired[project!]) +SUM(IHDesired\
		[project!])+SUM(EHDesired[project!]) - TotalNH-TotalIH-
TotalNH))
	~	people
	~		|

NRHireRate=
	if then else(Novices to Hire R > 0, Novices to Hire R/(Time to
hire/Gap Effect on Hiring R f\
		(GapRatio R)),0)
	~	people/Month
	~		|

NTHireRate=
	if then else(Novices to Hire T > 0, Novices to Hire T/(Time to
hire/Gap Effect on Hiring T f\
		(GapRatio T)),0)
	~	people/Month
	~		|

NDHireRate=
	if then else(Novices to Hire D > 0, Novices to Hire D/(Time to
hire/Gap Effect on Hiring D f\
		(GapRatio D)),0)
	~	people/Month
	~		|

Novices to Hire D=
	Min((Maximum Staff D-TotalND-TotalID-
TotalED),(SUM(NDDesired[project!]) +SUM(IDDesired\
		[project!])+SUM(EDDesired[project!]) - TotalND-TotalID-
TotalND))
	~	people
	~		|

Time to hire=
	Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b11')
	~	Month
	~		|

Novices to Hire T=
	Min((Maximum Staff T-TotalNT-TotalIT-
TotalET),(SUM(NTDesired[project!]) +SUM(ITDesired\
		[project!])+SUM(ETDesired[project!]) - TotalNT-TotalIT-
TotalNT))
	~	people
	~		|

Novices to Hire R=
	Min((Maximum Staff R-TotalNR-TotalIR-
TotalER),(SUM(NRDesired[project!]) +SUM(IRDesired\
```

```
		[project!])+SUM(ERDesired[project!]) - TotalNR-TotalIR-
TotalNR))
	~	people
	~		|

Complexity effect on PDY P1=
	Complexity effect on PDY f(Complexity P1)
	~	fraction
	~		|

Complexity effect on PDY P2=
	Complexity effect on PDY f(Complexity P2)
	~	fraction
	~		|

Complexity effect on PDY P3=
	Complexity effect on PDY f(Complexity P3)
	~	fraction
	~		|

Complexity effect on PDY P4=
	Complexity effect on PDY f(Complexity P4)
	~	fraction
	~		|

Complexity effect on quality f(
	[(0,0)-(10,10)],(0,1),(5,1),(10,1))
	~	fraction
	~		|

Complexity effect on quality P1=
	Complexity effect on quality f(Complexity P1)
	~	fraction
	~		|

Complexity effect on quality P2=
	Complexity effect on quality f(Complexity P2)
	~	fraction
	~		|

Complexity effect on quality P3=
	Complexity effect on quality f(Complexity P3)
	~	fraction
	~		|

Complexity effect on quality P4=
	Complexity effect on quality f(Complexity P4)
	~	fraction
	~		|

Qual P1R=
	Min(1, MaxQuality R*Fatigue effect qual P1R*Average Skill Effect
on Quality P1R*Complexity effect on quality P1\
		)
	~	fraction
	~		|

Complexity P2=
	Get XLS Constants('ModelConstants.xls','Project Constants','x3')
	~	dmnl
	~		|
```

```
Complexity P3=
      Get XLS Constants('ModelConstants.xls','Project Constants','x4')
      ~     dmnl
      ~           |

Complexity P4=
      Get XLS Constants('ModelConstants.xls','Project Constants','x5')
      ~     dmnl
      ~           |

Complexity Weight=
      Get XLS Constants('ModelConstants.xls','Portfolio Constants','b5')
      ~     dmnl
      ~           |

Complexity effect on learning P2=
      Complexity effect on learning f(Complexity P2)
      ~     fraction
      ~           |

Complexity effect on learning P3=
      Complexity effect on learning f(Complexity P3)
      ~     fraction
      ~           |

Complexity effect on learning P4=
      Complexity effect on learning f(Complexity P4)
      ~     fraction
      ~           |

Complexity effect on PDY f(
      [(0,0)-(10,10)],(0,1),(10,1))
      ~     fraction
      ~           |

Complexity effect on attractiveness P2=
      Complexity effect on attractiveness f(Complexity P2)
      ~     fraction
      ~           |

Complexity effect on attractiveness P3=
      Complexity effect on attractiveness f(Complexity P3)
      ~     fraction
      ~           |

PDY P1R=
       Normal Productivity R*Fatigue effect PDY P1R*Complexity effect on
PDY P1
      ~     lines/(people*Month)
      ~           |

Complexity effect on learning f(
      [(0,0)-(10,10)],(0,0.5),(5,1),(6,1.1),(10,1.5))
      ~     fraction
      ~           |

Complexity effect on learning P1=
      Complexity effect on learning f(Complexity P1)
      ~     fraction
      ~           |
```

```
Attractiveness P1R=
      (Bug ratio effect on attractiveness P1R*Bug Ratio Weight+Priority
effect on attractiveness P1R\
            *Priority Weight +Staffing Gap effect on attractiveness P1R
      *Staffing Gap Weight + Complexity effect on attractiveness
P1*Complexity Weight)*Active P1R
      ~     dmnl
      ~           |

Complexity effect on attractiveness f(
      [(0,0)-(10,10)],(0,0.1),(5,0.5),(10,1))
      ~     fraction
      ~           |

Complexity effect on attractiveness P1=
      Complexity effect on attractiveness f(Complexity P1)
      ~     fraction
      ~           |

Complexity P1=
      Get XLS Constants('ModelConstants.xls','Project Constants','x2')
      ~     dmnl
      ~           |

Complexity effect on attractiveness P4=
      Complexity effect on attractiveness f(Complexity P4)
      ~     fraction
      ~           |

Downsize Rate IH=
      if then else(Downsize Goal IH > 0, Min(Downsize Goal IH, IH
Control)/Time to downsize\
            ,0)
      ~     people/Month
      ~           |

Downsize Rate IR=
      if then else(Downsize Goal IR > 0, Min(Downsize Goal IR, IR
Control)/Time to downsize\
            ,0)
      ~     people/Month
      ~           |

Downsize Rate IT=
      if then else(Downsize Goal IT > 0, Min(Downsize Goal IT, IT
Control)/Time to downsize\
            ,0)
      ~     people/Month
      ~           |

NH Control= INTEG (
      -P1NH Rate - P2NH Rate - P3NH Rate - P4NH Rate+NHHireRate-Downsize
Rate NH,
            0)
      ~     people
      ~           |

Downsize Rate NH=
      if then else(Downsize Goal NH > 0, Min(Downsize Goal NH, NH
Control)/Time to downsize\
```

```
        ,0)
    ~   people/Month
    ~       |

Qual P2D=
    Min(1, MaxQuality D*Fatigue effect qual P2D*Average Skill Effect
on Quality P2D*Complexity effect on quality P2\
        )
    ~   fraction
    ~       |

Qual P2H=
    Min(1, MaxQuality H*Fatigue effect qual P2H*Average Skill Effect
on Quality P2H*Complexity effect on quality P2\
        )
    ~   fraction
    ~       |

Qual P2R=
    Min(1, MaxQuality R*Fatigue effect qual P2R*Average Skill Effect
on Quality P2R*Complexity effect on quality P2\
        )
    ~   fraction
    ~       |

Fatigue P2D=
     SMOOTHI(OverTime P2D,TimeToGetFatigued D,1)
    ~   fraction
    ~       |

Fatigue P2H=
     SMOOTHI(OverTime P2H,TimeToGetFatigued H,1)
    ~   fraction
    ~       |

Fatigue P2R=
     SMOOTHI(OverTime P2R,TimeToGetFatigued R,1)
    ~   fraction
    ~       |

Fatigue P2T=
     SMOOTHI(OverTime P2T,TimeToGetFatigued T,1)
    ~   fraction
    ~       |

Fatigue P3D=
     SMOOTHI(OverTime P3D,TimeToGetFatigued D,1)
    ~   fraction
    ~       |

Fatigue P3H=
     SMOOTHI(OverTime P3H,TimeToGetFatigued H,1)
    ~   fraction
    ~       |

Fatigue P3R=
     SMOOTHI(OverTime P3R,TimeToGetFatigued R,1)
    ~   fraction
    ~       |

Fatigue P3T=
```

```
        SMOOTHI(OverTime P3T,TimeToGetFatigued T,1)
       ~     fraction
       ~           |

Fatigue P4D=
        SMOOTHI(OverTime P4D,TimeToGetFatigued D,1)
       ~     fraction
       ~           |

Downsize Goal ED=
       TotalED - SUM(EDDesired[project!])
       ~     people
       ~           |

Downsize Goal EH=
       TotalEH - SUM(EHDesired[project!])
       ~     people
       ~           |

Downsize Goal ER=
       TotalER - SUM(ERDesired[project!])
       ~     people
       ~           |

Downsize Goal ET=
       TotalET - SUM(ETDesired[project!])
       ~     people
       ~           |

Downsize Goal ID=
       TotalID - SUM(IDDesired[project!])
       ~     people
       ~           |

Downsize Goal IH=
       TotalIH - SUM(IHDesired[project!])
       ~     people
       ~           |

Downsize Goal IR=
       TotalIR - SUM(IRDesired[project!])
       ~     people
       ~           |

Downsize Goal IT=
       TotalIT - SUM(ITDesired[project!])
       ~     people
       ~           |

Downsize Goal ND=
       TotalND - SUM(NDDesired[project!])
       ~     people
       ~           |

Downsize Goal NH=
       TotalNH - SUM(NHDesired[project!])
       ~     people
       ~           |

Downsize Goal NR=
       TotalNR - SUM(NRDesired[project!])
```

```
	~	people
	~		|

Downsize Goal NT=
	TotalNT - SUM(NTDesired[project!])
	~	people
	~		|

Downsize Rate ED=
	if then else(Downsize Goal ED > 0, Min(Downsize Goal ED, ED
Control)/Time to downsize\
		,0)
	~	people/Month
	~		|

Downsize Rate EH=
	if then else(Downsize Goal EH > 0, Min(Downsize Goal EH, EH
Control)/Time to downsize\
		,0)
	~	people/Month
	~		|

Downsize Rate ER=
	if then else(Downsize Goal ER > 0, Min(Downsize Goal ER, ER
Control)/Time to downsize\
		,0)
	~	people/Month
	~		|

Downsize Rate ET=
	if then else(Downsize Goal ET > 0, Min(Downsize Goal ET, ET
Control)/Time to downsize\
		,0)
	~	people/Month
	~		|

Downsize Rate ID=
	if then else(Downsize Goal ID > 0, Min(Downsize Goal ID, ID
Control)/Time to downsize\
		,0)
	~	people/Month
	~		|

FindBugs P4H=
	 HiddenBugs P4H/BugFindTime H
	~	lines/Month
	~		|

FindBugs P4R=
	 HiddenBugs P4R/BugFindTime R
	~	lines/Month
	~		|

FindBugs P4T=
	 HiddenBugs P4T/BugFindTime T
	~	lines/Month
	~		|

Downsize Rate ND=
	if then else(Downsize Goal ND > 0, Min(Downsize Goal ND, ND
Control)/Time to downsize\
```

```
        ,0)
    ~   people/Month
    ~       |

IT Control= INTEG (
    -P1IT Rate - P2IT Rate - P3IT Rate - P4IT Rate-Downsize Rate IT,
        0)
    ~   people
    ~       |

Downsize Rate NR=
    if then else(Downsize Goal NR > 0, Min(Downsize Goal NR, NR
Control)/Time to downsize\
        ,0)
    ~   people/Month
    ~       |

Downsize Rate NT=
    if then else(Downsize Goal NT > 0, Min(Downsize Goal NT, NT
Control)/Time to downsize\
        ,0)
    ~   people/Month
    ~       |

Percvd PDY P2R= INTEG (
    (PDY P2R - Percvd PDY P2R)/TimeToPercvPDY R,
        Normal Productivity R)
    ~   lines/(people*Month)
    ~       |

Percvd PDY P2T= INTEG (
    (PDY P2T - Percvd PDY P2T)/TimeToPercvPDY T,
        Normal Productivity T)
    ~   lines/(people*Month)
    ~       |

Percvd PDY P3D= INTEG (
    (PDY P3D - Percvd PDY P3D)/TimeToPercvPDY D,
        Normal Productivity D)
    ~   lines/(people*Month)
    ~       |

Intermediate Advance to Expert Time D=
    Get XLS Constants('ModelConstants.xls','Phase Constants','d3')
    ~   Month
    ~       |

Intermediate Advance to Expert Time H=
    Get XLS Constants('ModelConstants.xls','Phase Constants','c3')
    ~   Month
    ~       |

DueDate P2H=
    if then else(Time>(InitialDueDate P2H-minimum remaining time
H),Time+minimum remaining time H\
        ,InitialDueDate P2H)
    ~   Month
    ~       |

Intermediate Advance to Expert Time T=
    Get XLS Constants('ModelConstants.xls','Phase Constants','e3')
```

```
	~	Month
	~		|

DueDate P2T=
	if then else(Time>(InitialDueDate P2T-minimum remaining time
T),Time+minimum remaining time T\
		,InitialDueDate P2T)
	~	Month
	~		|

DueDate P3D=
	if then else(Time>(InitialDueDate P3D-minimum remaining time
D),Time+minimum remaining time D\
		,InitialDueDate P3D)
	~	Month
	~		|

DueDate P3H=
	if then else(Time>(InitialDueDate P3H-minimum remaining time
H),Time+minimum remaining time H\
		,InitialDueDate P3H)
	~	Month
	~		|

IR Control= INTEG (
	-P1IR Rate - P2IR Rate - P3IR Rate - P4IR Rate-Downsize Rate IR,
		0)
	~	people
	~		|

DueDate P3T=
	if then else(Time>(InitialDueDate P3T-minimum remaining time
T),Time+minimum remaining time T\
		,InitialDueDate P3T)
	~	Month
	~		|

DueDate P4D=
	if then else(Time>(InitialDueDate P4D-minimum remaining time
D),Time+minimum remaining time D\
		,InitialDueDate P4D)
	~	Month
	~		|

DueDate P4H=
	if then else(Time>(InitialDueDate P4H-minimum remaining time
H),Time+minimum remaining time H\
		,InitialDueDate P4H)
	~	Month
	~		|

Staffing Gap effect on learning P1T=
	Staffing Gap effect on learning f(ZIDZ(Workforce P1T,DesiredPeople
P1T))
	~	dmnl
	~		|

DueDate P4T=
	if then else(Time>(InitialDueDate P4T-minimum remaining time
T),Time+minimum remaining time T\
		,InitialDueDate P4T)
```

```
	~	Month
	~		|

ED Control= INTEG (
	-P1ED Rate - P2ED Rate - P3ED Rate - P4ED Rate-EDRetireRate-
Downsize Rate ED,
		0)
	~	people
	~		|

Staffing Gap effect on learning P2T=
	Staffing Gap effect on learning f(ZIDZ(Workforce P2T,DesiredPeople
P2T))
	~	dmnl
	~		|

Staffing Gap effect on learning P3D=
	Staffing Gap effect on learning f(ZIDZ(Workforce P3D,DesiredPeople
P3D))
	~	dmnl
	~		|

Staffing Gap effect on learning P3H=
	Staffing Gap effect on learning f(ZIDZ(Workforce P3H,DesiredPeople
P3H))
	~	dmnl
	~		|

Staffing Gap effect on learning P3R=
	Staffing Gap effect on learning f(ZIDZ(Workforce P3R,DesiredPeople
P3R))
	~	dmnl
	~		|

Staffing Gap effect on learning P3T=
	Staffing Gap effect on learning f(ZIDZ(Workforce P3T,DesiredPeople
P3T))
	~	dmnl
	~		|

Staffing Gap effect on learning P4D=
	Staffing Gap effect on learning f(ZIDZ(Workforce P4D,DesiredPeople
P4D))
	~	dmnl
	~		|

Staffing Gap effect on learning P4H=
	Staffing Gap effect on learning f(ZIDZ(Workforce P4H,DesiredPeople
P4H))
	~	dmnl
	~		|

Staffing Gap effect on learning P4R=
	Staffing Gap effect on learning f(ZIDZ(Workforce P4R,DesiredPeople
P4R))
	~	dmnl
	~		|

Staffing Gap effect on learning P4T=
	Staffing Gap effect on learning f(ZIDZ(Workforce P4T,DesiredPeople
P4T))
```

```
	~	dmnl
	~		|

ND Control= INTEG (
	-P1ND Rate - P2ND Rate - P3ND Rate - P4ND Rate+NDHireRate-Downsize
Rate ND,
		0)
	~	people
	~		|

Qual P4T=
	Min(1, MaxQuality T*Fatigue effect qual P4T*Average Skill Effect
on Quality P4T*Complexity effect on quality P4\
		)
	~	fraction
	~		|

Fatigue P4H=
	 SMOOTHI(OverTime P4H,TimeToGetFatigued H,1)
	~	fraction
	~		|

ER Control= INTEG (
	-P1ER Rate - P2ER Rate - P3ER Rate - P4ER Rate-ERRetireRate-
Downsize Rate ER,
		0)
	~	people
	~		|

Fatigue P4T=
	 SMOOTHI(OverTime P4T,TimeToGetFatigued T,1)
	~	fraction
	~		|

Qual P3R=
	Min(1, MaxQuality R*Fatigue effect qual P3R*Average Skill Effect
on Quality P3R*Complexity effect on quality P3\
		)
	~	fraction
	~		|

Staffing Gap effect on learning P2D=
	Staffing Gap effect on learning f(ZIDZ(Workforce P2D,DesiredPeople
P2D))
	~	dmnl
	~		|

Staffing Gap effect on learning P2H=
	Staffing Gap effect on learning f(ZIDZ(Workforce P2H,DesiredPeople
P2H))
	~	dmnl
	~		|

Percvd PDY P2D= INTEG (
	(PDY P2D - Percvd PDY P2D)/TimeToPercvPDY D,
		Normal Productivity D)
	~	lines/(people*Month)
	~		|

Percvd PDY P2H= INTEG (
	(PDY P2H - Percvd PDY P2H)/TimeToPercvPDY H,
```

```
            Normal Productivity H)
        ~       lines/(people*Month)
        ~           |

Qual P2T=
        Min(1, MaxQuality T*Fatigue effect qual P2T*Average Skill Effect
on Quality P2T*Complexity effect on quality P2\
            )
        ~       fraction
        ~           |

Qual P3D=
        Min(1, MaxQuality D*Fatigue effect qual P3D*Average Skill Effect
on Quality P3D*Complexity effect on quality P3\
            )
        ~       fraction
        ~           |

Qual P3H=
        Min(1, MaxQuality H*Fatigue effect qual P3H*Average Skill Effect
on Quality P3H*Complexity effect on quality P3\
            )
        ~       fraction
        ~           |

Staffing Gap effect on learning P1R=
        Staffing Gap effect on learning f(ZIDZ(Workforce P1R,DesiredPeople
P1R))
        ~       dmnl
        ~           |

Novice Advance to Intermediate Time D=
        Get XLS Constants('ModelConstants.xls','Phase Constants','d2')
        ~       Month
        ~           |

Novice Advance to Intermediate Time H=
        Get XLS Constants('ModelConstants.xls','Phase Constants','c2')
        ~       Month
        ~           |

Qual P4H=
        Min(1, MaxQuality H*Fatigue effect qual P4H*Average Skill Effect
on Quality P4H*Complexity effect on quality P4\
            )
        ~       fraction
        ~           |

Novice Advance to Intermediate Time T=
        Get XLS Constants('ModelConstants.xls','Phase Constants','e2')
        ~       Month
        ~           |

Staffing Gap effect on learning f(
        [(0,0)-
(1e+009,2)],(0,0),(0.25,0.25),(0.5,0.5),(0.75,0.75),(1,1),(1.1,1.25),(2,
1.5),\
            (1e+009,2))
        ~       dmnl
        ~           |
```

```
Staffing Gap effect on learning P1D=
      Staffing Gap effect on learning f(ZIDZ(Workforce P1D,DesiredPeople
P1D))
      ~     dmnl
      ~           |

Staffing Gap effect on learning P1H=
      Staffing Gap effect on learning f(ZIDZ(Workforce P1H,DesiredPeople
P1H))
      ~     dmnl
      ~           |

FindBugs P2T=
       HiddenBugs P2T/BugFindTime T
      ~     lines/Month
      ~           |

NT Control= INTEG (
      -P1NT Rate - P2NT Rate - P3NT Rate - P4NT Rate+NTHireRate-Downsize
Rate NT,
            0)
      ~     people
      ~           |

FindBugs P3H=
       HiddenBugs P3H/BugFindTime H
      ~     lines/Month
      ~           |

DueDate P4R=
      if then else(Time>(InitialDueDate P4R-minimum remaining time
R),Time+minimum remaining time R\
            ,InitialDueDate P4R)
      ~     Month
      ~           |

NR Control= INTEG (
      -P1NR Rate - P2NR Rate - P3NR Rate - P4NR Rate+NRHireRate-Downsize
Rate NR,
            0)
      ~     people
      ~           |

ET Control= INTEG (
      -P1ET Rate - P2ET Rate - P3ET Rate - P4ET Rate-ETRetireRate-
Downsize Rate ET,
            0)
      ~     people
      ~           |

FindBugs P2H=
       HiddenBugs P2H/BugFindTime H
      ~     lines/Month
      ~           |

FindBugs P2R=
       HiddenBugs P2R/BugFindTime R
      ~     lines/Month
      ~           |

DueDate P3R=
```

```
	if then else(Time>(InitialDueDate P3R-minimum remaining time R),Time+minimum remaining time R\
		,InitialDueDate P3R)
	~	Month
	~		|

Fatigue P4R=
	 SMOOTHI(OverTime P4R,TimeToGetFatigued R,1)
	~	fraction
	~		|

DueDate P2D=
	if then else(Time>(InitialDueDate P2D-minimum remaining time D),Time+minimum remaining time D\
		,InitialDueDate P2D)
	~	Month
	~		|

FindBugs P3R=
	 HiddenBugs P3R/BugFindTime R
	~	lines/Month
	~		|

FindBugs P3T=
	 HiddenBugs P3T/BugFindTime T
	~	lines/Month
	~		|

FindBugs P4D=
	 HiddenBugs P4D/BugFindTime D
	~	lines/Month
	~		|

Qual P4R=
	Min(1, MaxQuality R*Fatigue effect qual P4R*Average Skill Effect on Quality P4R*Complexity effect on quality P4\
		)
	~	fraction
	~		|

Percvd PDY P4R= INTEG (
	(PDY P4R - Percvd PDY P4R)/TimeToPercvPDY R,
		Normal Productivity R)
	~	lines/(people*Month)
	~		|

FindBugs P2D=
	 HiddenBugs P2D/BugFindTime D
	~	lines/Month
	~		|

Percvd PDY P3H= INTEG (
	(PDY P3H - Percvd PDY P3H)/TimeToPercvPDY H,
		Normal Productivity H)
	~	lines/(people*Month)
	~		|

Percvd PDY P4T= INTEG (
	(PDY P4T - Percvd PDY P4T)/TimeToPercvPDY T,
		Normal Productivity T)
	~	lines/(people*Month)
```

```
	~		|

IH Control= INTEG (
	-P1IH Rate - P2IH Rate - P3IH Rate - P4IH Rate-Downsize Rate IH,
		0)
	~	people
	~		|

DueDate P2R=
	if then else(Time>(InitialDueDate P2R-minimum remaining time
R),Time+minimum remaining time R\
		,InitialDueDate P2R)
	~	Month
	~		|

Staffing Gap effect on learning P2R=
	Staffing Gap effect on learning f(ZIDZ(Workforce P2R,DesiredPeople
P2R))
	~	dmnl
	~		|

Percvd PDY P4H= INTEG (
	(PDY P4H - Percvd PDY P4H)/TimeToPercvPDY H,
		Normal Productivity H)
	~	lines/(people*Month)
	~		|

ID Control= INTEG (
	-P1ID Rate - P2ID Rate - P3ID Rate - P4ID Rate-Downsize Rate ID,
		0)
	~	people
	~		|

FindBugs P3D=
	 HiddenBugs P3D/BugFindTime D
	~	lines/Month
	~		|

Percvd PDY P4D= INTEG (
	(PDY P4D - Percvd PDY P4D)/TimeToPercvPDY D,
		Normal Productivity D)
	~	lines/(people*Month)
	~		|

Time to downsize=
	Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b12')
	~	Month
	~		|

Qual P3T=
	Min(1, MaxQuality T*Fatigue effect qual P3T*Average Skill Effect
on Quality P3T*Complexity effect on quality P3\
		)
	~	fraction
	~		|

Qual P4D=
	Min(1, MaxQuality D*Fatigue effect qual P4D*Average Skill Effect
on Quality P4D*Complexity effect on quality P4\
		)
```

```
	~	fraction
	~		|

EH Control= INTEG (
	-P1EH Rate - P2EH Rate - P3EH Rate - P4EH Rate-EHRetireRate-
Downsize Rate EH,
		0)
	~	people
	~		|

Percvd PDY P3R= INTEG (
	(PDY P3R - Percvd PDY P3R)/TimeToPercvPDY R,
		Normal Productivity R)
	~	lines/(people*Month)
	~		|

Percvd PDY P3T= INTEG (
	(PDY P3T - Percvd PDY P3T)/TimeToPercvPDY T,
		Normal Productivity T)
	~	lines/(people*Month)
	~		|

P3NT= INTEG (
	P3NT Rate-P3NTtoIT Rate-Attrition Rate P3NT,
		StartP3NT)
	~	people
	~		|

P2ET= INTEG (
	P2ET Rate+P2ITtoET Rate-Attrition Rate P2ET,
		StartP2ET)
	~	people
	~		|

Attrition Rate P4IT=
	P4IT*Intermediate Attrition*Fatigue Effect on Attrition T
f(Fatigue P4T)*Complexity Effect on Attrition T f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P3ET=
	P3ET*Expert Attrition*Fatigue Effect on Attrition T f(Fatigue
P3T)*Complexity Effect on Attrition T f\
		(Complexity P3)/Time for Attrition
	~	people/Month
	~		|

P1IT= INTEG (
	P1IT Rate+P1NTtoIT Rate-P1ITtoET Rate-Attrition Rate P1IT,
		StartP1IT)
	~	people
	~		|

GapRatio T=
	ZIDZ((TotalNT+TotalIT+TotalET),(SUM(NTDesired[project!])+SUM(ITDes
ired[project!])+SUM\
		(ETDesired[project!])))
	~	fraction
	~		|
```

```
Gap Effect on Hiring T f(
	[(0,0)-(1e+009,20)],(0,2),(1e-
005,2),(0.25,2),(0.5,1.5),(0.9,1),(1,0.9),(1.1,0.8),(2\
		,0.7),(10,0.2),(100,0.1),(1e+009,0.01))
	~	fraction
	~		|

P4NT= INTEG (
	P4NT Rate-P4NTtoIT Rate-Attrition Rate P4NT,
		StartP4NT)
	~	people
	~		|

Total Attrition ET= INTEG (
	Attrition Rate P1ET+Attrition Rate P2ET+Attrition Rate
P3ET+Attrition Rate P4ET,
		0)
	~	people
	~		|

Fatigue Effect on Attrition T f(
	[(0,0)-(10,10)],(0,0),(0.8,0.8),(1,1.1),(2,10))
	~	fraction
	~		|

P4ET= INTEG (
	P4ET Rate+P4ITtoET Rate-Attrition Rate P4ET,
		StartP4ET)
	~	people
	~		|

Attrition Rate P3NT=
	P3NT*Novice Attrition*Fatigue Effect on Attrition T f(Fatigue
P3T)*Complexity Effect on Attrition T f\
		(Complexity P3)/Time for Attrition
	~	people/Month
	~		|

Total Attrition IT= INTEG (
	Attrition Rate P1IT+Attrition Rate P2IT+Attrition Rate
P3IT+Attrition Rate P4IT,
		0)
	~	people
	~		|

Attrition Rate P1ET=
	P1ET*Expert Attrition*Fatigue Effect on Attrition T f(Fatigue
P1T)*Complexity Effect on Attrition T f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

Total Attrition NT= INTEG (
	Attrition Rate P1NT+Attrition Rate P2NT+Attrition Rate
P3NT+Attrition Rate P4NT,
		0)
	~	people
	~		|

P1NT= INTEG (
	P1NT Rate-P1NTtoIT Rate-Attrition Rate P1NT,
```

```
		StartP1NT)
	~	people
	~		|

Attrition Rate P1IT=
	P1IT*Intermediate Attrition*Fatigue Effect on Attrition T
f(Fatigue P1T)*Complexity Effect on Attrition T f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

P1ET= INTEG (
	P1ET Rate+P1ITtoET Rate-Attrition Rate P1ET,
		StartP1ET)
	~	people
	~		|

P2NT= INTEG (
	P2NT Rate-P2NTtoIT Rate-Attrition Rate P2NT,
		StartP2NT)
	~	people
	~		|

Attrition Rate P4NT=
	P4NT*Novice Attrition*Fatigue Effect on Attrition T f(Fatigue
P4T)*Complexity Effect on Attrition T f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P1NT=
	P1NT*Novice Attrition*Fatigue Effect on Attrition T f(Fatigue
P1T)*Complexity Effect on Attrition T f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

TotalGap T=
	SUM(NTDesired[project!])-TotalNT+SUM(ITDesired[project!])-
TotalIT+SUM(ETDesired[project\
		!])-TotalET
	~	people
	~		|

Attrition Rate P2ET=
	P2ET*Expert Attrition*Fatigue Effect on Attrition T f(Fatigue
P2T)*Complexity Effect on Attrition T f\
		(Complexity P2)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P2IT=
	P2IT*Intermediate Attrition*Fatigue Effect on Attrition T
f(Fatigue P2T)*Complexity Effect on Attrition T f\
		(Complexity P2)/Time for Attrition
	~	people/Month
	~		|

P3ET= INTEG (
	P3ET Rate+P3ITtoET Rate-Attrition Rate P3ET,
		StartP3ET)
```

```
	~	people
	~		|

Attrition Rate P2NT=
	P2NT*Novice Attrition*Fatigue Effect on Attrition T f(Fatigue
P2T)*Complexity Effect on Attrition T f\
		(Complexity P2)/Time for Attrition
	~	people/Month
	~		|

P3IT= INTEG (
	P3IT Rate+P3NTtoIT Rate-P3ITtoET Rate-Attrition Rate P3IT,
		StartP3IT)
	~	people
	~		|

Attrition Rate P3IT=
	P3IT*Intermediate Attrition*Fatigue Effect on Attrition T
f(Fatigue P3T)*Complexity Effect on Attrition T f\
		(Complexity P3)/Time for Attrition
	~	people/Month
	~		|

P2IT= INTEG (
	P2IT Rate+P2NTtoIT Rate-P2ITtoET Rate-Attrition Rate P2IT,
		StartP2IT)
	~	people
	~		|

Attrition Rate P4ET=
	P4ET*Expert Attrition*Fatigue Effect on Attrition T f(Fatigue
P4T)*Complexity Effect on Attrition T f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

P4IT= INTEG (
	P4IT Rate+P4NTtoIT Rate-P4ITtoET Rate-Attrition Rate P4IT,
		StartP4IT)
	~	people
	~		|

Active P1D=
	if then else((Initial WorkToDo P1D<Init Dev P1D):AND:(((Initial
WorkToDo P1D+WorkToDo P1D\
		)/Init Dev P1D)>0.02),1,0)
	~	dmnl
	~		|

Active P1H=
	if then else((Initial WorkToDo P1H<Init HLD P1H):AND:(((Initial
WorkToDo P1H+WorkToDo P1H\
		)/Init HLD P1H)>0.02),1,0)
	~	dmnl
	~		|

Active P1R=
	if then else((Time>TimeToStart P1R):AND:((WorkToDo
P1R/InitialWorkToDo P1R)>0.02),1,\
		0)
	~	dmnl
```

```
	~		|

Active P1T=
	if then else((Initial WorkToDo P1T<Init Test P1T):AND:(((WorkToDo
P1T+Initial WorkToDo P1T\
		)/Init Test P1T)>0.02),1,0)
	~	dmnl
	~		|

Active P2D=
	if then else((Initial WorkToDo P2D<Init Dev P2D):AND:(((Initial
WorkToDo P2D+WorkToDo P2D\
		)/Init Dev P2D)>0.02),1,0)
	~	dmnl
	~		|

Active P2H=
	if then else((Initial WorkToDo P2H<Init HLD P2H):AND:(((Initial
WorkToDo P2H+WorkToDo P2H\
		)/Init HLD P2H)>0.02),1,0)
	~	dmnl
	~		|

Active P2R=
	if then else((Time>TimeToStart P2R):AND:((WorkToDo
P2R/InitialWorkToDo P2R)>0.02),1,\
		0)
	~	dmnl
	~		|

Active P2T=
	if then else((Initial WorkToDo P2T<Init Test P2T):AND:(((WorkToDo
P2T+Initial WorkToDo P2T\
		)/Init Test P2T)>0.02),1,0)
	~	dmnl
	~		|

Active P3D=
	if then else((Initial WorkToDo P3D<Init Dev P3D):AND:(((Initial
WorkToDo P3D+WorkToDo P3D\
		)/Init Dev P3D)>0.02),1,0)
	~	dmnl
	~		|

Active P3H=
	if then else((Initial WorkToDo P3H<Init HLD P3H):AND:(((Initial
WorkToDo P3H+WorkToDo P3H\
		)/Init HLD P3H)>0.02),1,0)
	~	dmnl
	~		|

Active P3R=
	if then else((Time>TimeToStart P3R):AND:((WorkToDo
P3R/InitialWorkToDo P3R)>0.02),1,\
		0)
	~	dmnl
	~		|

Active P3T=
	if then else((Initial WorkToDo P3T<Init Test P3T):AND:(((Initial
WorkToDo P3T+WorkToDo P3T\
```

```
		)/Init Test P3T)>0.02),1,0)
	~	dmnl
	~		|

Active P4D=
	if then else((Initial WorkToDo P4D<Init Dev P4D):AND:(((Initial 
WorkToDo P4D+WorkToDo P4D\
		)/Init Dev P4D)>0.02),1,0)
	~	dmnl
	~		|

Active P4H=
	if then else((Initial WorkToDo P4H<Init HLD P4H):AND:(((Initial 
WorkToDo P4H+WorkToDo P4H\
		)/Init HLD P4H)>0.02),1,0)
	~	dmnl
	~		|

Active P4R=
	if then else((Time>TimeToStart P4R):AND:((WorkToDo 
P4R/InitialWorkToDo P4R)>0.02),1,\
		0)
	~	dmnl
	~		|

Active P4T=
	if then else((Initial WorkToDo P4T<Init Test P4T):AND:(((Initial 
WorkToDo P4T+WorkToDo P4T\
		)/Init Test P4T)>0.02),1,0)
	~	dmnl
	~		|

AllocatedP1ED=
	EDAllocated[one]
	~	people
	~		|

AllocatedP1EH=
	EHAllocated[one]
	~	people
	~		|

AllocatedP1ER=
	ERAllocated[one]
	~	people
	~		|

AllocatedP1ET=
	ETAllocated[one]
	~	people
	~		|

AllocatedP1ID=
	IDAllocated[one]
	~	people
	~		|

AllocatedP1IH=
	IHAllocated[one]
	~	people
	~		|
```

```
AllocatedP1IR=
      IRAllocated[one]
      ~     people
      ~           |

AllocatedP1IT=
      ITAllocated[one]
      ~     people
      ~           |

AllocatedP1ND=
      NDAllocated[one]
      ~     people
      ~           |

AllocatedP1NH=
      NHAllocated[one]
      ~     people
      ~           |

AllocatedP1NR=
      NRAllocated[one]
      ~     people
      ~           |

AllocatedP1NT=
      NTAllocated[one]
      ~     people
      ~           |

AllocatedP2ED=
      EDAllocated[two]
      ~     people
      ~           |

AllocatedP2EH=
      EHAllocated[two]
      ~     people
      ~           |

AllocatedP2ER=
      ERAllocated[two]
      ~     people
      ~           |

AllocatedP2ET=
      ETAllocated[two]
      ~     people
      ~           |

AllocatedP2ID=
      IDAllocated[two]
      ~     people
      ~           |

AllocatedP2IH=
      IHAllocated[two]
      ~     people
      ~           |
```

```
AllocatedP2IR=
      IRAllocated[two]
      ~     people
      ~           |

AllocatedP2IT=
      ITAllocated[two]
      ~     people
      ~           |

AllocatedP2ND=
      NDAllocated[two]
      ~     people
      ~           |

AllocatedP2NH=
      NHAllocated[two]
      ~     people
      ~           |

AllocatedP2NR=
      NRAllocated[two]
      ~     people
      ~           |

AllocatedP2NT=
      NTAllocated[two]
      ~     people
      ~           |

AllocatedP3ED=
      EDAllocated[three]
      ~     people
      ~           |

AllocatedP3EH=
      EHAllocated[three]
      ~     people
      ~           |

AllocatedP3ER=
      ERAllocated[three]
      ~     people
      ~           |

AllocatedP3ET=
      ETAllocated[three]
      ~     people
      ~           |

AllocatedP3ID=
      IDAllocated[three]
      ~     people
      ~           |

AllocatedP3IH=
      IHAllocated[three]
      ~     people
      ~           |

AllocatedP3IR=
```

```
	IRAllocated[three]
	~	people
	~		|

AllocatedP3IT=
	ITAllocated[three]
	~	people
	~		|

AllocatedP3ND=
	NDAllocated[three]
	~	people
	~		|

AllocatedP3NH=
	NHAllocated[three]
	~	people
	~		|

AllocatedP3NR=
	NRAllocated[three]
	~	people
	~		|

AllocatedP3NT=
	NTAllocated[three]
	~	people
	~		|

AllocatedP4ED=
	EDAllocated[four]
	~	people
	~		|

AllocatedP4EH=
	EHAllocated[four]
	~	people
	~		|

AllocatedP4ER=
	ERAllocated[four]
	~	people
	~		|

AllocatedP4ET=
	ETAllocated[four]
	~	people
	~		|

AllocatedP4ID=
	IDAllocated[four]
	~	people
	~		|

AllocatedP4IR=
	IRAllocated[four]
	~	people
	~		|

AllocatedP4IR 0=
	IHAllocated[four]
```

```
	~	people
	~		|

AllocatedP4IT=
	ITAllocated[four]
	~	people
	~		|

AllocatedP4ND=
	NDAllocated[four]
	~	people
	~		|

AllocatedP4NH=
	NHAllocated[four]
	~	people
	~		|

AllocatedP4NT=
	NTAllocated[four]
	~	people
	~		|

Attractiveness P1D=
	(Bug ratio effect on attractiveness P1D*Bug Ratio Weight+Priority
effect on attractiveness P1D\
		*Priority Weight +Staffing Gap effect on attractiveness P1D
	*Staffing Gap Weight+Complexity effect on attractiveness
P1*Complexity Weight)*Active P1D
	~	dmnl
	~		|

Attractiveness P1H=
	(Bug ratio effect on attractiveness P1H*Bug Ratio Weight+Priority
effect on attractiveness P1H\
		*Priority Weight +Staffing Gap effect on attractiveness P1H
	*Staffing Gap Weight+Complexity effect on attractiveness
P1*Complexity Weight)*Active P1H
	~	dmnl
	~		|

Attractiveness P1T=
	(Bug ratio effect on attractiveness P1T*Bug Ratio Weight+Priority
effect on attractiveness P1T\
		*Priority Weight +Staffing Gap effect on attractiveness P1T
	*Staffing Gap Weight+Complexity effect on attractiveness
P1*Complexity Weight)*Active P1T
	~	dmnl
	~		|

Attractiveness P2D=
	(Bug ratio effect on attractiveness P2D*Bug Ratio Weight+Priority
effect on attractiveness P2D\
		*Priority Weight +Staffing Gap effect on attractiveness P2D
	*Staffing Gap Weight+ Complexity effect on attractiveness
P2*Complexity Weight)*Active P2D
	~	dmnl
	~		|

Attractiveness P2H=
```

```
	(Bug ratio effect on attractiveness P2H*Bug Ratio Weight+Priority
effect on attractiveness P2H\
		*Priority Weight +Staffing Gap effect on attractiveness P2H
	*Staffing Gap Weight+ Complexity effect on attractiveness
P2*Complexity Weight)*Active P2H
	~	dmnl
	~		|

Attractiveness P2R=
	(Bug ratio effect on attractiveness P2R*Bug Ratio Weight+Priority
effect on attractiveness P2R\
		*Priority Weight +Staffing Gap effect on attractiveness P2R
	*Staffing Gap Weight+ Complexity effect on attractiveness
P2*Complexity Weight)*Active P2R
	~	dmnl
	~		|

Attractiveness P2T=
	(Bug ratio effect on attractiveness P2T*Bug Ratio Weight+Priority
effect on attractiveness P2T\
		*Priority Weight +Staffing Gap effect on attractiveness P2T
	*Staffing Gap Weight+ Complexity effect on attractiveness
P2*Complexity Weight)*Active P2T
	~	dmnl
	~		|

Attractiveness P3D=
	(Bug ratio effect on attractiveness P3D*Bug Ratio Weight+Priority
effect on attractiveness P3D\
		*Priority Weight +Staffing Gap effect on attractiveness P3D
	*Staffing Gap Weight + Complexity effect on attractiveness
P3*Complexity Weight)*Active P3D
	~	dmnl
	~		|

Attractiveness P3H=
	(Bug ratio effect on attractiveness P3H*Bug Ratio Weight+Priority
effect on attractiveness P3H\
		*Priority Weight +Staffing Gap effect on attractiveness P3H
	*Staffing Gap Weight + Complexity effect on attractiveness
P3*Complexity Weight)*Active P3H
	~	dmnl
	~		|

Attractiveness P3R=
	(Bug ratio effect on attractiveness P3R*Bug Ratio Weight+Priority
effect on attractiveness P3R\
		*Priority Weight +Staffing Gap effect on attractiveness P3R
	*Staffing Gap Weight + Complexity effect on attractiveness
P3*Complexity Weight)*Active P3R
	~	dmnl
	~		|

Attractiveness P3T=
	(Bug ratio effect on attractiveness P3T*Bug Ratio Weight+Priority
effect on attractiveness P3T\
		*Priority Weight +Staffing Gap effect on attractiveness P3T
	*Staffing Gap Weight + Complexity effect on attractiveness
P3*Complexity Weight)*Active P3T
	~	dmnl
	~		|
```

```
Attractiveness P4D=
      (Bug ratio effect on attractiveness P4D*Bug Ratio Weight+Priority
effect on attractiveness P4D\
            *Priority Weight +Staffing Gap effect on attractiveness P4D
      *Staffing Gap Weight + Complexity effect on attractiveness
P4*Complexity Weight)*Active P4D
      ~     dmnl
      ~           |

Attractiveness P4H=
      (Bug ratio effect on attractiveness P4H*Bug Ratio Weight+Priority
effect on attractiveness P4H\
            *Priority Weight +Staffing Gap effect on attractiveness P4H
      *Staffing Gap Weight + Complexity effect on attractiveness
P4*Complexity Weight)*Active P4H
      ~     dmnl
      ~           |

Attractiveness P4R=
      (Bug ratio effect on attractiveness P4R*Bug Ratio Weight+Priority
effect on attractiveness P4R\
            *Priority Weight +Staffing Gap effect on attractiveness P4R
      *Staffing Gap Weight+ Complexity effect on attractiveness
P4*Complexity Weight)*Active P4R
      ~     dmnl
      ~           |

Attractiveness P4T=
      (Bug ratio effect on attractiveness P4T*Bug Ratio Weight+Priority
effect on attractiveness P4T\
            *Priority Weight +Staffing Gap effect on attractiveness P4T
      *Staffing Gap Weight + Complexity effect on attractiveness
P4*Complexity Weight)*Active P4T
      ~     dmnl
      ~           |

Attrition Rate P1ED=
      P1ED*Expert Attrition*Fatigue Effect on Attrition D f(Fatigue
P1D)*Complexity Effect on Attrition D f\
            (Complexity P1)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P1EH=
      P1EH*Expert Attrition*Fatigue Effect on Attrition H f(Fatigue
P1H)*Complexity Effect on Attrition H f\
            (Complexity P1)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P1ER=
      P1ER*Expert Attrition*Fatigue Effect on Attrition R f(Fatigue
P1R)*Complexity Effect on Attrition R f\
            (Complexity P1)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P1ID=
      P1ID*Intermediate Attrition*Fatigue Effect on Attrition D
f(Fatigue P1D)*Complexity Effect on Attrition D f\
```

```
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P1IH=
	P1IH*Intermediate Attrition*Fatigue Effect on Attrition H
f(Fatigue P1H)*Complexity Effect on Attrition H f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P1IR=
	P1IR*Intermediate Attrition*Fatigue Effect on Attrition R
f(Fatigue P1R)*Complexity Effect on Attrition R f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P1ND=
	P1ND*Novice Attrition*Fatigue Effect on Attrition D f(Fatigue
P1D)*Complexity Effect on Attrition D f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P1NH=
	P1NH*Novice Attrition*Fatigue Effect on Attrition H f(Fatigue
P1H)*Complexity Effect on Attrition H f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P1NR=
	P1NR*Novice Attrition*Fatigue Effect on Attrition R f(Fatigue
P1R)*Complexity Effect on Attrition R f\
		(Complexity P1)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P2ED=
	P2ED*Expert Attrition*Fatigue Effect on Attrition D f(Fatigue
P2D)*Complexity Effect on Attrition D f\
		(Complexity P2)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P2EH=
	P2EH*Expert Attrition*Fatigue Effect on Attrition H f(Fatigue
P2H)*Complexity Effect on Attrition H f\
		(Complexity P2)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P2ER=
	P2ER*Expert Attrition*Fatigue Effect on Attrition R f(Fatigue
P2R)*Complexity Effect on Attrition R f\
		(Complexity P2)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P2ID=
```

```
      P2ID*Intermediate Attrition*Fatigue Effect on Attrition D
f(Fatigue P2D)*Complexity Effect on Attrition D f\
            (Complexity P2)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P2IH=
      P2IH*Intermediate Attrition*Fatigue Effect on Attrition H
f(Fatigue P2H)*Complexity Effect on Attrition H f\
            (Complexity P2)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P2IR=
      P2IR*Intermediate Attrition*Fatigue Effect on Attrition R
f(Fatigue P2R)*Complexity Effect on Attrition R f\
            (Complexity P2)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P2ND=
      P2ND*Novice Attrition*Fatigue Effect on Attrition D f(Fatigue
P2D)*Complexity Effect on Attrition D f\
            (Complexity P2)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P2NH=
      P2NH*Novice Attrition*Fatigue Effect on Attrition H f(Fatigue
P2H)*Complexity Effect on Attrition H f\
            (Complexity P2)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P2NR=
      P2NR*Novice Attrition*Fatigue Effect on Attrition R f(Fatigue
P2R)*Complexity Effect on Attrition R f\
            (Complexity P2)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3ED=
      P3ED*Expert Attrition*Fatigue Effect on Attrition D f(Fatigue
P3D)*Complexity Effect on Attrition D f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3EH=
      P3EH*Expert Attrition*Fatigue Effect on Attrition H f(Fatigue
P3H)*Complexity Effect on Attrition H f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3ER=
      P3ER*Expert Attrition*Fatigue Effect on Attrition R f(Fatigue
P3R)*Complexity Effect on Attrition R f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |
```

```
Attrition Rate P3ID=
      P3ID*Intermediate Attrition*Fatigue Effect on Attrition D
f(Fatigue P3D)*Complexity Effect on Attrition D f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3IH=
      P3IH*Intermediate Attrition*Fatigue Effect on Attrition H
f(Fatigue P3H)*Complexity Effect on Attrition H f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3IR=
      P3IR*Intermediate Attrition*Fatigue Effect on Attrition R
f(Fatigue P3R)*Complexity Effect on Attrition R f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3ND=
      P3ND*Novice Attrition*Fatigue Effect on Attrition D f(Fatigue
P3D)*Complexity Effect on Attrition D f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3NH=
      P3NH*Novice Attrition*Fatigue Effect on Attrition H f(Fatigue
P3H)*Complexity Effect on Attrition H f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P3NR=
      P3NR*Novice Attrition*Fatigue Effect on Attrition R f(Fatigue
P3R)*Complexity Effect on Attrition R f\
            (Complexity P3)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P4ED=
      P4ED*Expert Attrition*Fatigue Effect on Attrition D f(Fatigue
P4D)*Complexity Effect on Attrition D f\
            (Complexity P4)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P4EH=
      P4EH*Expert Attrition*Fatigue Effect on Attrition H f(Fatigue
P4H)*Complexity Effect on Attrition H f\
            (Complexity P4)/Time for Attrition
      ~     people/Month
      ~           |

Attrition Rate P4ER=
      P4ER*Expert Attrition*Fatigue Effect on Attrition R f(Fatigue
P4R)*Complexity Effect on Attrition R f\
            (Complexity P4)/Time for Attrition
```

```
	~	people/Month
	~		|

Attrition Rate P4ID=
	P4ID*Intermediate Attrition*Fatigue Effect on Attrition D
f(Fatigue P4D)*Complexity Effect on Attrition D f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P4IH=
	P4IH*Intermediate Attrition*Fatigue Effect on Attrition H
f(Fatigue P4H)*Complexity Effect on Attrition H f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P4IR=
	P4IR*Intermediate Attrition*Fatigue Effect on Attrition R
f(Fatigue P4R)*Complexity Effect on Attrition R f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P4ND=
	P4ND*Novice Attrition*Fatigue Effect on Attrition D f(Fatigue
P4D)*Complexity Effect on Attrition D f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P4NH=
	P4NH*Novice Attrition*Fatigue Effect on Attrition H f(Fatigue
P4H)*Complexity Effect on Attrition H f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Attrition Rate P4NR=
	P4NR*Novice Attrition*Fatigue Effect on Attrition R f(Fatigue
P4R)*Complexity Effect on Attrition R f\
		(Complexity P4)/Time for Attrition
	~	people/Month
	~		|

Average Skill Effect on Quality P1D=
	((P1ND*NoviceMultiplier P1D*Novice Skill Effect on
Quality)+(P1ID*IntMultiplier P1D*\
		Intermediate Skill Effect on Quality)+(P1ED*ExpertMultiplier
P1D*Expert Skill Effect on Quality\
		))/Workforce P1D
	~	fraction
	~		|

Average Skill Effect on Quality P4T=
	((P4NT*NoviceMultiplier P4T*Novice Skill Effect on
Quality)+(P4IT*IntMultiplier P4T*\
		Intermediate Skill Effect on Quality)+(P4ET*ExpertMultiplier
P4T*Expert Skill Effect on Quality\
		))/Workforce P4T
	~	fraction
```

```
	~		|

Average Skill Effect on Quality P1H=
	((P1NH*NoviceMultiplier P1H*Novice Skill Effect on 
Quality)+(P1IH*IntMultiplier P1H*\
		Intermediate Skill Effect on Quality)+(P1EH*ExpertMultiplier 
P1H*Expert Skill Effect on Quality\
		))/Workforce P1H
	~	fraction
	~		|

Average Skill Effect on Quality P1R=
	((P1NR*NoviceMultiplier P1R*Novice Skill Effect on 
Quality)+(P1IR*IntMultiplier P1R*\
		Intermediate Skill Effect on Quality)+(P1ER*ExpertMultiplier 
P1R*Expert Skill Effect on Quality\
		))/Workforce P1R
	~	fraction
	~		|

Average Skill Effect on Quality P1T=
	((P1NT*NoviceMultiplier P1T*Novice Skill Effect on 
Quality)+(P1IT*IntMultiplier P1T*\
		Intermediate Skill Effect on Quality)+(P1ET*ExpertMultiplier 
P1T*Expert Skill Effect on Quality\
		))/Workforce P1T
	~	fraction
	~		|

Average Skill Effect on Quality P2D=
	((P2ND*NoviceMultiplier P2D*Novice Skill Effect on 
Quality)+(P2ID*IntMultiplier P2D*\
		Intermediate Skill Effect on Quality)+(P2ED*ExpertMultiplier 
P2D*Expert Skill Effect on Quality\
		))/Workforce P2D
	~	fraction
	~		|

Average Skill Effect on Quality P2H=
	((P2NH*NoviceMultiplier P2H*Novice Skill Effect on 
Quality)+(P2IH*IntMultiplier P2H*\
		Intermediate Skill Effect on Quality)+(P2EH*ExpertMultiplier 
P2H*Expert Skill Effect on Quality\
		))/Workforce P2H
	~	fraction
	~		|

Average Skill Effect on Quality P2R=
	((P2NR*NoviceMultiplier P2R*Novice Skill Effect on 
Quality)+(P2IR*IntMultiplier P2R*\
		Intermediate Skill Effect on Quality)+(P2ER*ExpertMultiplier 
P2R*Expert Skill Effect on Quality\
		))/Workforce P2R
	~	fraction
	~		|

Average Skill Effect on Quality P2T=
	((P2NT*NoviceMultiplier P2T*Novice Skill Effect on 
Quality)+(P2IT*IntMultiplier P2T*\
		Intermediate Skill Effect on Quality)+(P2ET*ExpertMultiplier 
P2T*Expert Skill Effect on Quality\
```

```
		))/Workforce P2T
	~	fraction
	~		|

Average Skill Effect on Quality P3D=
	((P3ND*NoviceMultiplier P3D*Novice Skill Effect on 
Quality)+(P3ID*IntMultiplier P3D*\
		Intermediate Skill Effect on Quality)+(P3ED*ExpertMultiplier 
P3D*Expert Skill Effect on Quality\
		))/Workforce P3D
	~	fraction
	~		|

Average Skill Effect on Quality P3H=
	((P3NH*NoviceMultiplier P3H*Novice Skill Effect on 
Quality)+(P3IH*IntMultiplier P3H*\
		Intermediate Skill Effect on Quality)+(P3EH*ExpertMultiplier 
P3H*Expert Skill Effect on Quality\
		))/Workforce P3H
	~	fraction
	~		|

Average Skill Effect on Quality P3R=
	((P3NR*NoviceMultiplier P3R*Novice Skill Effect on 
Quality)+(P3IR*IntMultiplier P3R*\
		Intermediate Skill Effect on Quality)+(P3ER*ExpertMultiplier 
P3R*Expert Skill Effect on Quality\
		))/Workforce P3R
	~	fraction
	~		|

Average Skill Effect on Quality P3T=
	((P3NT*NoviceMultiplier P3T*Novice Skill Effect on 
Quality)+(P3IT*IntMultiplier P3T*\
		Intermediate Skill Effect on Quality)+(P3ET*ExpertMultiplier 
P3T*Expert Skill Effect on Quality\
		))/Workforce P3T
	~	fraction
	~		|

Average Skill Effect on Quality P4D=
	((P4ND*NoviceMultiplier P4D*Novice Skill Effect on 
Quality)+(P4ID*IntMultiplier P4D*\
		Intermediate Skill Effect on Quality)+(P4ED*ExpertMultiplier 
P4D*Expert Skill Effect on Quality\
		))/Workforce P4D
	~	fraction
	~		|

Average Skill Effect on Quality P4H=
	((P4NH*NoviceMultiplier P4H*Novice Skill Effect on 
Quality)+(P4IH*IntMultiplier P4H*\
		Intermediate Skill Effect on Quality)+(P4EH*ExpertMultiplier 
P4H*Expert Skill Effect on Quality\
		))/Workforce P4H
	~	fraction
	~		|

Average Skill Effect on Quality P4R=
	((P4NR*NoviceMultiplier P4R*Novice Skill Effect on 
Quality)+(P4IR*IntMultiplier P4R*\
```

```
            Intermediate Skill Effect on Quality)+(P4ER*ExpertMultiplier
P4R*Expert Skill Effect on Quality\
            ))/Workforce P4R
      ~     fraction
      ~           |

Bug ratio effect on attractiveness f(
      [(0,0)-(2e+030,1)],(0,1),(1,0),(100,0),(1e+030,0))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P1D=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P1D,Done
Right P1D))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P1H=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P1H,Done
Right P1H))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P1R=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P1R,Done
Right P1R))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P1T=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P1T,Done
Right P1T))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P2D=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P2D,Done
Right P2D))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P2H=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P2H,Done
Right P2H))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P2R=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P2R,Done
Right P2R))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P2T=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P2T,Done
Right P2T))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P3D=
```

```
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P3D,Done
Right P3D))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P3H=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P3H,Done
Right P3H))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P3R=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P3R,Done
Right P3R))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P3T=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P3T,Done
Right P3T))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P4D=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P4D,Done
Right P4D))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P4H=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P4H,Done
Right P4H))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P4R=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P4R,Done
Right P4R))
      ~     dmnl
      ~           |

Bug ratio effect on attractiveness P4T=
      Bug ratio effect on attractiveness f(ZIDZ(HiddenBugs P4T,Done
Right P4T))
      ~     dmnl
      ~           |

Bug Ratio Weight=
      Get XLS Constants('ModelConstants.xls','Portfolio Constants','b3')
      ~     dmnl
      ~           |

BugFindTime D=
      Get XLS Constants('ModelConstants.xls','Phase Constants','d9')
      ~     Month
      ~           |

BugFindTime H=
      Get XLS Constants('ModelConstants.xls','Phase Constants','c9')
      ~     Month
      ~           |
```

```
BugFindTime R=
      Get XLS Constants('ModelConstants.xls','Phase Constants','b9')
      ~     Month
      ~          |

BugFindTime T=
      Get XLS Constants('ModelConstants.xls','Phase Constants','e9')
      ~     Month
      ~          |

DesiredP1ED=
      P1DDesiredE
      ~     people
      ~          |

DesiredP1EH=
      P1HDesiredE
      ~     people
      ~          |

DesiredP1ER=
      P1RDesiredE
      ~     people
      ~          |

DesiredP1ET=
      P1TDesiredE
      ~     people
      ~          |

DesiredP1ID=
      P1DDesiredI
      ~     people
      ~          |

DesiredP1IH=
      P1HDesiredI
      ~     people
      ~          |

DesiredP1IR=
      P1RDesiredI
      ~     people
      ~          |

DesiredP1IT=
      P1TDesiredI
      ~     people
      ~          |

DesiredP1ND=
      P1DDesiredN
      ~     people
      ~          |

DesiredP1NH=
      P1HDesiredN
      ~     people
      ~          |
```

```
DesiredP1NR=
      P1RDesiredN
      ~     people
      ~           |

DesiredP1NT=
      P1TDesiredN
      ~     people
      ~           |

DesiredP2ED=
      P2DDesiredE
      ~     people
      ~           |

DesiredP2EH=
      P2HDesiredE
      ~     people
      ~           |

DesiredP2ER=
      P2RDesiredE
      ~     people
      ~           |

DesiredP2ET=
      P2TDesiredE
      ~     people
      ~           |

DesiredP2ID=
      P2DDesiredI
      ~     people
      ~           |

DesiredP2IH=
      P2HDesiredI
      ~     people
      ~           |

DesiredP2IR=
      P2RDesiredI
      ~     people
      ~           |

DesiredP2IT=
      P2TDesiredI
      ~     people
      ~           |

DesiredP2ND=
      P2DDesiredN
      ~     people
      ~           |

DesiredP2NH=
      P2HDesiredN
      ~     people
      ~           |

DesiredP2NR=
```

```
      P2RDesiredN
      ~     people
      ~           |

DesiredP2NT=
      P2TDesiredN
      ~     people
      ~           |

DesiredP3ED=
      P3DDesiredE
      ~     people
      ~           |

DesiredP3EH=
      P3HDesiredE
      ~     people
      ~           |

DesiredP3ER=
      P3RDesiredE
      ~     people
      ~           |

DesiredP3ET=
      P3TDesiredE
      ~     people
      ~           |

DesiredP3ID=
      P3DDesiredI
      ~     people
      ~           |

DesiredP3IH=
      P3HDesiredI
      ~     people
      ~           |

DesiredP3IR=
      P3RDesiredI
      ~     people
      ~           |

DesiredP3IT=
      P3TDesiredI
      ~     people
      ~           |

DesiredP3ND=
      P3DDesiredN
      ~     people
      ~           |

DesiredP3NH=
      P3HDesiredN
      ~     people
      ~           |

DesiredP3NR=
      P3RDesiredN
```

```
	~	people
	~		|

DesiredP3NT=
	P3TDesiredN
	~	people
	~		|

DesiredP4ED=
	P4DDesiredE
	~	people
	~		|

DesiredP4EH=
	P4HDesiredE
	~	people
	~		|

DesiredP4ER=
	P4RDesiredE
	~	people
	~		|

DesiredP4ET=
	P4TDesiredE
	~	people
	~		|

DesiredP4ID=
	P4DDesiredI
	~	people
	~		|

DesiredP4IH=
	P4HDesiredI
	~	people
	~		|

DesiredP4IR=
	P4RDesiredI
	~	people
	~		|

DesiredP4IT=
	P4TDesiredI
	~	people
	~		|

DesiredP4ND=
	P4DDesiredN
	~	people
	~		|

DesiredP4NH=
	P4HDesiredN
	~	people
	~		|

DesiredP4NR=
	P4RDesiredN
	~	people
```

```
      ~            |

DesiredP4NT=
      P4TDesiredN
      ~     people
      ~            |

DesiredPeople P1D=
      (WorkToDo P1D/Remaining Time P1D)/Percvd PDY P1D+((Initial
WorkToDo P1D/Remaining Time P1D\
            )/Percvd PDY P1D)*0.75*Active P1D
      ~     people
      ~            |

DesiredPeople P1R=
      (WorkToDo P1R/Remaining Time P1R)/Percvd PDY P1R
      ~     people
      ~            |

DesiredPeople P1T=
      (WorkToDo P1T/Remaining Time P1T)/Percvd PDY P1T+((Initial
WorkToDo P1T/Remaining Time P1T\
            )/Percvd PDY P1T)*0.75*Active P1T
      ~     people
      ~            |

DesiredPeople P2D=
      (WorkToDo P2D/Remaining Time P2D)/Percvd PDY P2D+((Initial
WorkToDo P2D/Remaining Time P2D\
            )/Percvd PDY P2D)*0.75*Active P2D
      ~     people
      ~            |

DesiredPeople P2H=
      ( (WorkToDo P2H/Remaining Time P2H)/Percvd PDY P2H+((Initial
WorkToDo P2H/Remaining Time P2H\
            )/Percvd PDY P2H)*0.75*Active P2H)
      ~     people
      ~            |

DesiredPeople P2R=
      (WorkToDo P2R/Remaining Time P2R)/Percvd PDY P2R
      ~     people
      ~            |

DesiredPeople P2T=
      (WorkToDo P2T/Remaining Time P2T)/Percvd PDY P2T+((Initial
WorkToDo P2T/Remaining Time P2T\
            )/Percvd PDY P2T)*0.75*Active P2T
      ~     people
      ~            |

DesiredPeople P3D=
      (WorkToDo P3D/Remaining Time P3D)/Percvd PDY P3D+((Initial
WorkToDo P3D/Remaining Time P3D\
            )/Percvd PDY P3D)*0.75*Active P3D
      ~     people
      ~            |

DesiredPeople P3H=
```

```
	if then else(P3 Initial Priority>0,((WorkToDo P3H/Remaining Time P3H)/Percvd PDY P3H\
		+((Initial WorkToDo P3H/Remaining Time P3H)/Percvd PDY P3H)*0.75*Active P3H),0)
	~	people
	~		|

DesiredPeople P3R=
	if then else(P3 Initial Priority>0, (WorkToDo P3R/Remaining Time P3R)/Percvd PDY P3R\
		,0)
	~	people
	~		|

DesiredPeople P3T=
	(WorkToDo P3T/Remaining Time P3T)/Percvd PDY P3T+((Initial WorkToDo P3T/Remaining Time P3T\
		)/Percvd PDY P3T)*0.75*Active P3T
	~	people
	~		|

DesiredPeople P4D=
	(WorkToDo P4D/Remaining Time P4D)/Percvd PDY P4D+((Initial WorkToDo P4D/Remaining Time P4D\
		)/Percvd PDY P4D)*0.75*Active P4D
	~	people
	~		|

DesiredPeople P4H=
	((WorkToDo P4H/Remaining Time P4H)/Percvd PDY P4H+((Initial WorkToDo P4H/Remaining Time P4H\
		)/Percvd PDY P4H)*0.75*Active P4H)
	~	people
	~		|

DesiredPeople P4R=
	(WorkToDo P4R/Remaining Time P4R)/Percvd PDY P4R
	~	people
	~		|

DesiredPeople P4T=
	(WorkToDo P4T/Remaining Time P4T)/Percvd PDY P4T+((Initial WorkToDo P4T/Remaining Time P4T\
		)/Percvd PDY P4T)*0.75*Active P4T
	~	people
	~		|

Doing P1D=
	 Min(WorkToDo P1D/TIME STEP,Effective People P1D*PDY P1D)
	~	lines/Month
	~		|

Doing P1H=
	 Min(WorkToDo P1H/TIME STEP,Effective People P1H*PDY P1H)
	~	lines/Month
	~		|

Doing P1R=
	 Min(WorkToDo P1R/TIME STEP,Effective People P1R*PDY P1R)
	~	lines/Month
	~		|
```

```
Doing P1T=
       Min(WorkToDo P1T/TIME STEP,Effective People P1T*PDY P1T)
      ~     lines/Month
      ~          |

Doing P2D=
       Min(WorkToDo P2D/TIME STEP,Effective People P2D*PDY P2D)
      ~     lines/Month
      ~          |

Doing P2H=
       Min(WorkToDo P2H/TIME STEP,Effective People P2H*PDY P2H)
      ~     lines/Month
      ~          |

Doing P2R=
       Min(WorkToDo P2R/TIME STEP,Effective People P2R*PDY P2R)
      ~     lines/Month
      ~          |

Doing P2T=
       Min(WorkToDo P2T/TIME STEP,Effective People P2T*PDY P2T)
      ~     lines/Month
      ~          |

Doing P3D=
       Min(WorkToDo P3D/TIME STEP,Effective People P3D*PDY P3D)
      ~     lines/Month
      ~          |

Doing P3H=
       Min(WorkToDo P3H/TIME STEP,Effective People P3H*PDY P3H)
      ~     lines/Month
      ~          |

Doing P3R=
       Min(WorkToDo P3R/TIME STEP,Effective People P3R*PDY P3R)
      ~     lines/Month
      ~          |

Doing P3T=
       Min(WorkToDo P3T/TIME STEP,Effective People P3T*PDY P3T)
      ~     lines/Month
      ~          |

Doing P4D=
       Min(WorkToDo P4D/TIME STEP,Effective People P4D*PDY P4D)
      ~     lines/Month
      ~          |

Doing P4H=
       Min(WorkToDo P4H/TIME STEP,Effective People P4H*PDY P4H)
      ~     lines/Month
      ~          |

Doing P4R=
       Min(WorkToDo P4R/TIME STEP,Effective People P4R*PDY P4R)
      ~     lines/Month
      ~          |
```

```
Doing P4T=
	 Min(WorkToDo P4T/TIME STEP,Effective People P4T*PDY P4T)
	~	lines/Month
	~		|

Doing right P1D=
	Doing P1D*Qual P1D
	~	lines/Month
	~		|

Doing right P1H=
	Doing P1H*Qual P1H
	~	lines/Month
	~		|

Doing right P1R=
	Doing P1R*Qual P1R
	~	lines/Month
	~		|

Doing right P1T=
	 Doing P1T*Qual P1T
	~	lines/Month
	~		|

Doing right P2D=
	Doing P2D*Qual P2D
	~	lines/Month
	~		|

Doing right P2H=
	Doing P2H*Qual P2H
	~	lines/Month
	~		|

Doing right P2R=
	 Doing P2R*Qual P2R
	~	lines/Month
	~		|

Doing right P2T=
	 Doing P2T*Qual P2T
	~	lines/Month
	~		|

Doing right P3D=
	 Doing P3D*Qual P3D
	~	lines/Month
	~		|

Doing right P3H=
	 Doing P3H*Qual P3H
	~	lines/Month
	~		|

Doing right P3R=
	 Doing P3R*Qual P3R
	~	lines/Month
	~		|

Doing right P3T=
```

```
        Doing P3T*Qual P3T
       ~     lines/Month
       ~            |

Doing right P4D=
        Doing P4D*Qual P4D
       ~     lines/Month
       ~            |

Doing right P4H=
        Doing P4H*Qual P4H
       ~     lines/Month
       ~            |

Doing right P4R=
        Doing P4R*Qual P4R
       ~     lines/Month
       ~            |

Doing right P4T=
        Doing P4T*Qual P4T
       ~     lines/Month
       ~            |

Doing wrong P1D = Doing P1D*(1-Qual P1D)
       ~     lines/Month
       ~            |

Doing wrong P1H = Doing P1H*(1-Qual P1H)
       ~     lines/Month
       ~            |

Doing wrong P1R = Doing P1R*(1-Qual P1R)
       ~     lines/Month
       ~            |

Doing wrong P1T = Doing P1T*(1-Qual P1T)
       ~     lines/Month
       ~            |

Doing wrong P2D = Doing P2D*(1-Qual P2D)
       ~     lines/Month
       ~            |

Doing wrong P2H = Doing P2H*(1-Qual P2H)
       ~     lines/Month
       ~            |

Doing wrong P2R = Doing P2R*(1-Qual P2R)
       ~     lines/Month
       ~            |

Doing wrong P2T = Doing P2T*(1-Qual P2T)
       ~     lines/Month
       ~            |

Doing wrong P3D = Doing P3D*(1-Qual P3D)
       ~     lines/Month
       ~            |

Doing wrong P3H = Doing P3H*(1-Qual P3H)
```

```
	~	lines/Month
	~		|

Doing wrong P3R = Doing P3R*(1-Qual P3R)
	~	lines/Month
	~		|

Doing wrong P3T = Doing P3T*(1-Qual P3T)
	~	lines/Month
	~		|

Doing wrong P4D = Doing P4D*(1-Qual P4D)
	~	lines/Month
	~		|

Doing wrong P4H = Doing P4H*(1-Qual P4H)
	~	lines/Month
	~		|

Doing wrong P4R = Doing P4R*(1-Qual P4R)
	~	lines/Month
	~		|

Doing wrong P4T = Doing P4T*(1-Qual P4T)
	~	lines/Month
	~		|

Done Right P1D= INTEG (
	Doing right P1D,
		0)
	~	lines
	~		|

Done Right P1H= INTEG (
	Doing right P1H,
		0)
	~	lines
	~		|

Done Right P1R= INTEG (
	Doing right P1R,
		0)
	~	lines
	~		|

Done Right P1T = INTEG(Doing right P1T,   0)
	~	lines
	~		|

Done Right P2D= INTEG (
	Doing right P2D,
		0)
	~	lines
	~		|

Done Right P2H= INTEG (
	Doing right P2H,
		0)
	~	lines
	~		|
```

```
Done Right P2R = INTEG(Doing right P2R,   0)
      ~     lines
      ~           |

Done Right P2T = INTEG(Doing right P2T,   0)
      ~     lines
      ~           |

Done Right P3D = INTEG(Doing right P3D,   0)
      ~     lines
      ~           |

Done Right P3H = INTEG(Doing right P3H,   0)
      ~     lines
      ~           |

Done Right P3R = INTEG(Doing right P3R,   0)
      ~     lines
      ~           |

Done Right P3T = INTEG(Doing right P3T,   0)
      ~     lines
      ~           |

Done Right P4D = INTEG(Doing right P4D,   0)
      ~     lines
      ~           |

Done Right P4H = INTEG(Doing right P4H,   0)
      ~     lines
      ~           |

Done Right P4R = INTEG(Doing right P4R,   0)
      ~     lines
      ~           |

Done Right P4T = INTEG(Doing right P4T,   0)
      ~     lines
      ~           |

DueDate P1D=
      if then else(Time>(InitialDueDate P1D-minimum remaining time
D),Time+minimum remaining time D\
            ,InitialDueDate P1D)
      ~     Month
      ~           |

DueDate P1H=
      if then else(Time>(InitialDueDate P1H-minimum remaining time
H),Time+minimum remaining time H\
            ,InitialDueDate P1H)
      ~     Month
      ~           |

DueDate P1R=
      if then else(Time>(InitialDueDate P1R-minimum remaining time
R),Time+minimum remaining time R\
            ,InitialDueDate P1R)
      ~     Month
      ~           |
```

```
DueDate P1T=
      if then else(Time>(InitialDueDate P1T-minimum remaining time
T),Time+minimum remaining time T\
            ,InitialDueDate P1T)
      ~     Month
      ~           |

EDAllocated[project]=
      Allocate By
Priority(EDDesired[project],EDAttractiveness[project],4,1,TotalED)
      ~     people
      ~           |

EDAttractiveness[one]=
      Attractiveness P1D ~~|
EDAttractiveness[two]=
      Attractiveness P2D ~~|
EDAttractiveness[three]=
      Attractiveness P3D ~~|
EDAttractiveness[four]=
      Attractiveness P4D
      ~     dmnl
      ~           |

EDDesired[one]=
      DesiredP1ED ~~|
EDDesired[two]=
      DesiredP2ED ~~|
EDDesired[three]=
      DesiredP3ED ~~|
EDDesired[four]=
      DesiredP4ED
      ~     people
      ~           |

EDRetired= INTEG (
      EDRetireRate,
            0)
      ~     people
      ~           |

EDRetireRate=
      Min(Experts to retire D, ED Control) / Time to retire
      ~     people/Month
      ~           |

Effective People P1D=
      OverTime P1D*Workforce P1D
      ~     people
      ~           |

Effective People P1H=
      OverTime P1H*Workforce P1H
      ~     people
      ~           |

Effective People P1R=
      OverTime P1R*Workforce P1R
      ~     people
      ~           |
```

```
Effective People P1T=
      OverTime P1T*Workforce P1T
      ~     people
      ~           |

Effective People P2D=
      OverTime P2D*Workforce P2D
      ~     people
      ~           |

Effective People P2H=
      OverTime P2H*Workforce P2H
      ~     people
      ~           |

Effective People P2R=
      OverTime P2R*Workforce P2R
      ~     people
      ~           |

Effective People P2T=
      OverTime P2T*Workforce P2T
      ~     people
      ~           |

Effective People P3D=
      OverTime P3D*Workforce P3D
      ~     people
      ~           |

Effective People P3H=
      OverTime P3H*Workforce P3H
      ~     people
      ~           |

Effective People P3R=
      OverTime P3R*Workforce P3R
      ~     people
      ~           |

Effective People P3T=
      OverTime P3T*Workforce P3T
      ~     people
      ~           |

Effective People P4D=
      OverTime P4D*Workforce P4D
      ~     people
      ~           |

Effective People P4H=
      OverTime P4H*Workforce P4H
      ~     people
      ~           |

Effective People P4R=
      OverTime P4R*Workforce P4R
      ~     people
      ~           |

Effective People P4T=
```

```
      OverTime P4T*Workforce P4T
      ~     people
      ~           |

EffectOfFatigue OnProductivity f(
      [(0,0)-(2,2)],(0,1),(2,1))
      ~     dmnl
      ~           |

EffectOfFatigue OnQuality f(
      [(0,0)-(2,1.1)],(0,1),(2,1))
      ~     dmnl
      ~           |

EHAllocated[project]=
      Allocate By
Priority(EHDesired[project],EHAttractiveness[project],4,1,TotalEH)
      ~     people
      ~           |

EHAttractiveness[one]=
      Attractiveness P1H ~~|
EHAttractiveness[two]=
      Attractiveness P2H ~~|
EHAttractiveness[three]=
      Attractiveness P3H ~~|
EHAttractiveness[four]=
      Attractiveness P4H
      ~     dmnl
      ~           |

EHDesired[one]=
      DesiredP1EH ~~|
EHDesired[two]=
      DesiredP2EH ~~|
EHDesired[three]=
      DesiredP3EH ~~|
EHDesired[four]=
      DesiredP4EH
      ~     people
      ~           |

EHRetired= INTEG (
      EHRetireRate,
            0)
      ~     people
      ~           |

EHRetireRate=
      Min(Experts to retire H, EH Control)/Time to retire
      ~     people/Month
      ~           |

ERAllocated[project]=
      Allocate By
Priority(ERDesired[project],ERAttractiveness[project],4,1,TotalER)
      ~     people
      ~           |

ERAttractiveness[one]=
      Attractiveness P1R ~~|
```

```
ERAttractiveness[two]=
      Attractiveness P2R ~~|
ERAttractiveness[three]=
      Attractiveness P3R ~~|
ERAttractiveness[four]=
      Attractiveness P4R
      ~     dmnl
      ~           |

ERDesired[one]=
      DesiredP1ER ~~|
ERDesired[two]=
      DesiredP2ER ~~|
ERDesired[three]=
      DesiredP3ER ~~|
ERDesired[four]=
      DesiredP4ER
      ~     people
      ~           |

ERRetired= INTEG (
      ERRetireRate,
            0)
      ~     people
      ~           |

ERRetireRate=
      Min(Experts to retire R, ER Control)/Time to retire
      ~     people/Month
      ~           |

ETAllocated[project]=
      Allocate By
Priority(ETDesired[project],ETAttractiveness[project],4,1,TotalET)
      ~     people
      ~           |

ETAttractiveness[one]=
      Attractiveness P1T ~~|
ETAttractiveness[two]=
      Attractiveness P2T ~~|
ETAttractiveness[three]=
      Attractiveness P3T ~~|
ETAttractiveness[four]=
      Attractiveness P4T
      ~     dmnl
      ~           |

ETDesired[one]=
      DesiredP1ET ~~|
ETDesired[two]=
      DesiredP2ET ~~|
ETDesired[three]=
      DesiredP3ET ~~|
ETDesired[four]=
      DesiredP4ET
      ~     people
      ~           |

ETRetired= INTEG (
      ETRetireRate,
```

```
		0)
	~	people
	~		|

ETRetireRate=
	Min(Experts to retire T, ET Control)/Time to retire
	~	people/Month
	~		|

Expert Attrition=
	0.1
	~	fraction
	~	10 percent / year
	|

Expert Skill Effect on Quality=
	Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b22')
	~	fraction
	~		|

Experts to retire D=
	0.02*TotalED
	~	people
	~		|

Experts to retire H=
	0.02*TotalEH
	~	people
	~		|

Experts to retire R=
	0.01*TotalER
	~	people
	~	0.02*TotalER
	|

Experts to retire T=
	0.02*TotalET
	~	people
	~		|

Fatigue Effect on Attrition D f(
	[(0,0)-(10,10)],(0,0),(0.8,0.8),(1,1.1),(2,10))
	~	fraction
	~		|

Fatigue Effect on Attrition H f(
	[(0,0)-(10,10)],(0,0),(0.8,0.8),(1,1.1),(2,10))
	~	fraction
	~		|

Fatigue Effect on Attrition R f(
	[(0,0)-(10,10)],(0,0.1),(0.8,0.8),(1,1.1),(1.5,1.5),(2,10))
	~	fraction
	~		|

Fatigue effect PDY P1D=
	 EffectOfFatigue OnProductivity f(Fatigue P1D)
	~	dmnl
	~		|
```

```
Fatigue effect PDY P1H=
        EffectOfFatigue OnProductivity f(Fatigue P1H)
       ~     dmnl
       ~           |

Fatigue effect PDY P1R=
        EffectOfFatigue OnProductivity f(Fatigue P1R)
       ~     dmnl
       ~           |

Fatigue effect PDY P1T=
        EffectOfFatigue OnProductivity f(Fatigue P1T)
       ~     dmnl
       ~           |

Fatigue effect PDY P2D=
        EffectOfFatigue OnProductivity f(Fatigue P2D)
       ~     dmnl
       ~           |

Fatigue effect PDY P2H=
        EffectOfFatigue OnProductivity f(Fatigue P2H)
       ~     dmnl
       ~           |

Fatigue effect PDY P2R=
        EffectOfFatigue OnProductivity f(Fatigue P2R)
       ~     dmnl
       ~           |

Fatigue effect PDY P2T=
        EffectOfFatigue OnProductivity f(Fatigue P2T)
       ~     dmnl
       ~           |

Fatigue effect PDY P3D=
        EffectOfFatigue OnProductivity f(Fatigue P3D)
       ~     dmnl
       ~           |

Fatigue effect PDY P3H=
        EffectOfFatigue OnProductivity f(Fatigue P3H)
       ~     dmnl
       ~           |

Fatigue effect PDY P3R=
        EffectOfFatigue OnProductivity f(Fatigue P3R)
       ~     dmnl
       ~           |

Fatigue effect PDY P3T=
        EffectOfFatigue OnProductivity f(Fatigue P3T)
       ~     dmnl
       ~           |

Fatigue effect PDY P4D=
        EffectOfFatigue OnProductivity f(Fatigue P4D)
       ~     dmnl
       ~           |
```

```
Fatigue effect PDY P4H=
	EffectOfFatigue OnProductivity f(Fatigue P4H)
	~	dmnl
	~		|

Fatigue effect PDY P4R=
	EffectOfFatigue OnProductivity f(Fatigue P4R)
	~	dmnl
	~		|

Fatigue effect PDY P4T=
	EffectOfFatigue OnProductivity f(Fatigue P4T)
	~	dmnl
	~		|

Fatigue effect qual P1D=
	EffectOfFatigue OnQuality f(Fatigue P1D)
	~	dmnl
	~		|

Fatigue effect qual P1H=
	EffectOfFatigue OnQuality f(Fatigue P1H)
	~	dmnl
	~		|

Fatigue effect qual P1R=
	EffectOfFatigue OnQuality f(Fatigue P1R)
	~	dmnl
	~		|

Fatigue effect qual P1T=
	EffectOfFatigue OnQuality f(Fatigue P1T)
	~	dmnl
	~		|

Fatigue effect qual P2D=
	EffectOfFatigue OnQuality f(Fatigue P2D)
	~	dmnl
	~		|

Fatigue effect qual P2H=
	EffectOfFatigue OnQuality f(Fatigue P2H)
	~	dmnl
	~		|

Fatigue effect qual P2R=
	EffectOfFatigue OnQuality f(Fatigue P2R)
	~	dmnl
	~		|

Fatigue effect qual P2T=
	EffectOfFatigue OnQuality f(Fatigue P2T)
	~	dmnl
	~		|

Fatigue effect qual P3D=
	EffectOfFatigue OnQuality f(Fatigue P3D)
	~	dmnl
	~		|

Fatigue effect qual P3H=
```

```
	EffectOfFatigue OnQuality f(Fatigue P3H)
	~	dmnl
	~		|

Fatigue effect qual P3R=
	EffectOfFatigue OnQuality f(Fatigue P3R)
	~	dmnl
	~		|

Fatigue effect qual P3T=
	EffectOfFatigue OnQuality f(Fatigue P3T)
	~	dmnl
	~		|

Fatigue effect qual P4D=
	EffectOfFatigue OnQuality f(Fatigue P4D)
	~	dmnl
	~		|

Fatigue effect qual P4H=
	EffectOfFatigue OnQuality f(Fatigue P4H)
	~	dmnl
	~		|

Fatigue effect qual P4R=
	EffectOfFatigue OnQuality f(Fatigue P4R)
	~	dmnl
	~		|

Fatigue effect qual P4T=
	EffectOfFatigue OnQuality f(Fatigue P4T)
	~	dmnl
	~		|

Fatigue P1D=
	SMOOTHI(OverTime P1D,TimeToGetFatigued D,1)
	~	fraction
	~		|

Fatigue P1H=
	SMOOTHI(OverTime P1H,TimeToGetFatigued H,1)
	~	fraction
	~		|

Fatigue P1R=
	SMOOTHI(OverTime P1R,TimeToGetFatigued R,1)
	~	fraction
	~		|

Fatigue P1T=
	SMOOTHI(OverTime P1T,TimeToGetFatigued T,1)
	~	fraction
	~		|

FindBugs P1D = HiddenBugs P1D/BugFindTime D
	~	lines/Month
	~		|

FindBugs P1H = HiddenBugs P1H/BugFindTime H
	~	lines/Month
	~		|
```

```
FindBugs P1R = HiddenBugs P1R/BugFindTime R
      ~     lines/Month
      ~           |

FindBugs P1T = HiddenBugs P1T/BugFindTime T
      ~     lines/Month
      ~           |

Gap Effect on Hiring D f(
      [(0,0)-(1e+006,20)],(0,2),(1e-
005,2),(0.25,2),(0.5,1.5),(0.9,1),(1,0.9),(1.1,0.8),(2\
            ,0.7),(10,0.2),(100,0.1),(1e+006,0.01))
      ~     fraction
      ~           |

Gap Effect on Hiring H f(
      [(0,0)-(1e+009,20)],(0,2),(1e-
005,2),(0.25,2),(0.5,1.5),(0.9,1),(1,0.9),(1.1,0.8),(2\
            ,0.7),(10,0.2),(100,0.1),(1e+009,0.01))
      ~     fraction
      ~           |

Gap Effect on Hiring R f(
      [(0,0)-(1e+009,20)],(0,2),(1e-
005,2),(0.25,2),(0.5,1.5),(0.9,1),(1,0.9),(1.1,0.8),(2\
            ,0.7),(10,0.2),(100,0.1),(1e+009,0.01))
      ~     fraction
      ~           |

GapP1ED=
      AllocatedP1ED-P1ED
      ~     people
      ~           |

GapP1EH=
      AllocatedP1EH-P1EH
      ~     people
      ~           |

GapP1ER=
      AllocatedP1ER-P1ER
      ~     people
      ~           |

GapP1ET=
      AllocatedP1ET-P1ET
      ~     people
      ~           |

GapP1ID=
      AllocatedP1ID-P1ID
      ~     people
      ~           |

GapP1IH=
      AllocatedP1IH-P1IH
      ~     people
      ~           |

GapP1IR=
```

```
	AllocatedP1IR-P1IR
	~	people
	~		|

GapP1IT=
	AllocatedP1IT-P1IT
	~	people
	~		|

GapP1ND=
	AllocatedP1ND-P1ND
	~	people
	~		|

GapP1NH=
	AllocatedP1NH-P1NH
	~	people
	~		|

GapP1NR=
	AllocatedP1NR-P1NR
	~	people
	~		|

GapP1NT=
	AllocatedP1NT-P1NT
	~	people
	~		|

GapP2ED=
	AllocatedP2ED-P2ED
	~	people
	~		|

GapP2EH=
	AllocatedP2EH-P2EH
	~	people
	~		|

GapP2ER=
	AllocatedP2ER-P2ER
	~	people
	~		|

GapP2ET=
	AllocatedP2ET-P2ET
	~	people
	~		|

GapP2ID=
	AllocatedP2ID-P2ID
	~	people
	~		|

GapP2IH=
	AllocatedP2IH-P2IH
	~	people
	~		|

GapP2IT=
	AllocatedP2IT-P2IT
```

```
	~	people
	~		|

GapP2ND=
	AllocatedP2ND-P2ND
	~	people
	~		|

GapP2NH=
	AllocatedP2NH-P2NH
	~	people
	~		|

GapP2NR=
	AllocatedP2NR-P2NR
	~	people
	~		|

GapP2NT=
	AllocatedP2NT-P2NT
	~	people
	~		|

GapP3ED=
	AllocatedP3ED-P3ED
	~	people
	~		|

GapP3EH=
	AllocatedP3EH-P3EH
	~	people
	~		|

GapP3ER=
	AllocatedP3ER-P3ER
	~	people
	~		|

GapP3ET=
	AllocatedP3ET-P3ET
	~	people
	~		|

GapP3ID=
	AllocatedP3ID-P3ID
	~	people
	~		|

GapP3IH=
	AllocatedP3IH-P3IH
	~	people
	~		|

GapP3IR=
	AllocatedP3IR-P3IR
	~	people
	~		|

GapP3IT=
	AllocatedP3IT-P3IT
	~	people
```

```
	~		|

GapP3ND=
	AllocatedP3ND-P3ND
	~	people
	~		|

GapP3NH=
	AllocatedP3NH-P3NH
	~	people
	~		|

GapP3NR=
	AllocatedP3NR-P3NR
	~	people
	~		|

GapP3NT=
	AllocatedP3NT-P3NT
	~	people
	~		|

GapP4ED=
	AllocatedP4ED-P4ED
	~	people
	~		|

GapP4EH=
	AllocatedP4EH-P4EH
	~	people
	~		|

GapP4ER=
	AllocatedP4ER-P4ER
	~	people
	~		|

GapP4ET=
	AllocatedP4ET-P4ET
	~	people
	~		|

GapP4ID=
	AllocatedP4ID-P4ID
	~	people
	~		|

GapP4IH=
	AllocatedP4IR 0-P4IH
	~	people
	~		|

GapP4IR=
	AllocatedP4IR-P4IR
	~	people
	~		|

GapP4IT=
	AllocatedP4IT-P4IT
	~	people
	~		|
```

```
GapP4ND=
      AllocatedP4ND-P4ND
      ~     people
      ~           |

GapP4NH=
      AllocatedP4NH-P4NH
      ~     people
      ~           |

GapP4NR=
      AllocatedP4NR-P4NR
      ~     people
      ~           |

GapP4NT=
      AllocatedP4NT-P4NT
      ~     people
      ~           |

GapRatio D=
      ZIDZ((TotalND+TotalID+TotalED),(SUM(NDDesired[project!])+SUM(IDDes
ired[project!])+SUM\
            (EDDesired[project!])))
      ~     fraction
      ~           |

GapRatio H=
      ZIDZ((TotalNH+TotalIH+TotalEH),(SUM(NHDesired[project!])+SUM(IHDes
ired[project!])+SUM\
            (EHDesired[project!])))
      ~     fraction
      ~           |

GapRatio R=
      ZIDZ((TotalNR+TotalIR+TotalER),(SUM(NRDesired[project!])+SUM(IRDes
ired[project!])+SUM\
            (ERDesired[project!])))
      ~     fraction
      ~           |

HiddenBugs P1D = INTEG(Doing wrong P1D - FindBugs P1D,  0)
      ~     lines
      ~           |

HiddenBugs P1H = INTEG(Doing wrong P1H - FindBugs P1H,  0)
      ~     lines
      ~           |

HiddenBugs P1R = INTEG(Doing wrong P1R - FindBugs P1R,  0)
      ~     lines
      ~           |

HiddenBugs P1T = INTEG(Doing wrong P1T - FindBugs P1T,  0)
      ~     lines
      ~           |

HiddenBugs P2D = INTEG(Doing wrong P2D - FindBugs P2D,  0)
      ~     lines
      ~           |
```

```
HiddenBugs P2H = INTEG(Doing wrong P2H - FindBugs P2H,  0)
      ~     lines
      ~           |

HiddenBugs P2R = INTEG(Doing wrong P2R - FindBugs P2R,  0)
      ~     lines
      ~           |

HiddenBugs P2T = INTEG(Doing wrong P2T - FindBugs P2T,  0)
      ~     lines
      ~           |

HiddenBugs P3D = INTEG(Doing wrong P3D - FindBugs P3D,  0)
      ~     lines
      ~           |

HiddenBugs P3H = INTEG(Doing wrong P3H - FindBugs P3H,  0)
      ~     lines
      ~           |

HiddenBugs P3R = INTEG(Doing wrong P3R - FindBugs P3R,  0)
      ~     lines
      ~           |

HiddenBugs P3T = INTEG(Doing wrong P3T - FindBugs P3T,  0)
      ~     lines
      ~           |

HiddenBugs P4D = INTEG(Doing wrong P4D - FindBugs P4D,  0)
      ~     lines
      ~           |

HiddenBugs P4H = INTEG(Doing wrong P4H - FindBugs P4H,  0)
      ~     lines
      ~           |

HiddenBugs P4R = INTEG(Doing wrong P4R - FindBugs P4R,  0)
      ~     lines
      ~           |

HiddenBugs P4T = INTEG(Doing wrong P4T - FindBugs P4T,  0)
      ~     lines
      ~           |

HLD Start Rate P1H=
      if then else(Initial WorkToDo P1H>0,(Doing right
P1R/InitialWorkToDo P1R)*Init HLD P1H\
            ,0)
      ~     lines/Month
      ~           |

HLD Start Rate P2H=
      if then else(Initial WorkToDo P2H>0,(Doing right
P2R/InitialWorkToDo P2R)*Init HLD P2H\
            ,0)
      ~     lines/Month
      ~           |

HLD Start Rate P3H=
```

```
      if then else(Initial WorkToDo P3H>0,(Doing right
P3R/InitialWorkToDo P3R)*Init HLD P3H\
            ,0)
      ~     lines/Month
      ~           |

HLD Start Rate P4H=
      if then else(Initial WorkToDo P4H>0,(Doing right
P4R/InitialWorkToDo P4R)*Init HLD P4H\
            ,0)
      ~     lines/Month
      ~           |

IDAllocated[project]=
      Allocate By
Priority(IDDesired[project],IDAttractiveness[project],4,1,TotalID)
      ~     people
      ~           |

IDAttractiveness[one]=
      Attractiveness P1D ~~|
IDAttractiveness[two]=
      Attractiveness P2D ~~|
IDAttractiveness[three]=
      Attractiveness P3D ~~|
IDAttractiveness[four]=
      Attractiveness P4D
      ~     dmnl
      ~           |

IDDesired[one]=
      DesiredP1ID ~~|
IDDesired[two]=
      DesiredP2ID ~~|
IDDesired[three]=
      DesiredP3ID ~~|
IDDesired[four]=
      DesiredP4ID
      ~     people
      ~           |

IHAllocated[project]=
      Allocate By
Priority(IHDesired[project],IHAttractiveness[project],4,1,TotalIH)
      ~     people
      ~           |

IHAttractiveness[one]=
      Attractiveness P1H ~~|
IHAttractiveness[two]=
      Attractiveness P2H ~~|
IHAttractiveness[three]=
      Attractiveness P3H ~~|
IHAttractiveness[four]=
      Attractiveness P4H
      ~     dmnl
      ~           |

IHDesired[one]=
      DesiredP1IH ~~|
IHDesired[two]=
```

```
	DesiredP2IH ~~|
IHDesired[three]=
	DesiredP3IH ~~|
IHDesired[four]=
	DesiredP4IH
	~	people
	~		|

Indicated overtime P1D=
	DesiredPeople P1D/Workforce P1D
	~	fraction
	~		|

Indicated overtime P1H=
	DesiredPeople P1H/Workforce P1H
	~	fraction
	~		|

Indicated overtime P1R=
	DesiredPeople P1R/(Workforce P1R)
	~	fraction
	~		|

Indicated overtime P1T=
	DesiredPeople P1T/Workforce P1T
	~	fraction
	~		|

Indicated overtime P2D=
	DesiredPeople P2D/Workforce P2D
	~	fraction
	~		|

Indicated overtime P2H=
	DesiredPeople P2H/Workforce P2H
	~	fraction
	~		|

Indicated overtime P2R=
	DesiredPeople P2R/Workforce P2R
	~	fraction
	~		|

Indicated overtime P2T=
	DesiredPeople P2T/Workforce P2T
	~	fraction
	~		|

Indicated overtime P3D=
	DesiredPeople P3D/Workforce P3D
	~	fraction
	~		|

Indicated overtime P3H=
	DesiredPeople P3H/Workforce P3H
	~	fraction
	~		|

Indicated overtime P3R=
	DesiredPeople P3R/Workforce P3R
	~	fraction
```

```
	~		|

Indicated overtime P3T=
	DesiredPeople P3T/Workforce P3T
	~	fraction
	~		|

Indicated overtime P4D=
	DesiredPeople P4D/Workforce P4D
	~	fraction
	~		|

Indicated overtime P4H=
	DesiredPeople P4H/Workforce P4H
	~	fraction
	~		|

Indicated overtime P4R=
	DesiredPeople P4R/Workforce P4R
	~	fraction
	~		|

Indicated overtime P4T=
	DesiredPeople P4T/Workforce P4T
	~	fraction
	~		|

Init Dev P1D=
	Get XLS Constants('ModelConstants.xls','Project Constants','q2')
	~	lines
	~		|

Init Dev P2D=
	Get XLS Constants('ModelConstants.xls','Project Constants','q3')
	~	lines
	~		|

Init Dev P3D=
	Get XLS Constants('ModelConstants.xls','Project Constants','q4')
	~	lines
	~		|

Init Dev P4D=
	Get XLS Constants('ModelConstants.xls','Project Constants','q5')
	~	lines
	~		|

Init HLD P1H=
	Get XLS Constants('ModelConstants.xls','Project Constants','p2')
	~	lines
	~		|

Init HLD P2H=
	Get XLS Constants('ModelConstants.xls','Project Constants','p3')
	~	lines
	~		|

Init HLD P3H=
	Get XLS Constants('ModelConstants.xls','Project Constants','p4')
	~	lines
	~		|
```

```
Init HLD P4H=
      Get XLS Constants('ModelConstants.xls','Project Constants','p5')
      ~     lines
      ~           |

Init Test P1T=
      Get XLS Constants('ModelConstants.xls','Project Constants','r2')
      ~     lines
      ~           |

Init Test P2T=
      Get XLS Constants('ModelConstants.xls','Project Constants','r3')
      ~     lines
      ~           |

Init Test P3T=
      Get XLS Constants('ModelConstants.xls','Project Constants','r4')
      ~     lines
      ~           |

Init Test P4T=
      Get XLS Constants('ModelConstants.xls','Project Constants','r5')
      ~     lines
      ~           |

Initial WorkToDo P1D= INTEG (
      -Start Dev Rate P1D,
            Init Dev P1D)
      ~     lines
      ~           |

Initial WorkToDo P1H= INTEG (
      -HLD Start Rate P1H,
            Init HLD P1H)
      ~     lines
      ~           |

Initial WorkToDo P1T= INTEG (
      -Test Start Rate P1T,
            Init Test P1T)
      ~     lines
      ~           |

Initial WorkToDo P2D= INTEG (
      -Start Dev Rate P2D,
            Init Dev P2D)
      ~     lines
      ~           |

Initial WorkToDo P2H= INTEG (
      -HLD Start Rate P2H,
            Init HLD P2H)
      ~     lines
      ~           |

Initial WorkToDo P2T= INTEG (
      -Test Start Rate P2T,
            Init Test P2T)
      ~     lines
      ~           |
```

```
Initial WorkToDo P3D= INTEG (
      -Start Dev Rate P3D,
            Init Dev P3D)
      ~     lines
      ~           |

Initial WorkToDo P3H= INTEG (
      -HLD Start Rate P3H,
            Init HLD P3H)
      ~     lines
      ~           |

Initial WorkToDo P3T= INTEG (
      -Test Start Rate P3T,
            Init Test P3T)
      ~     lines
      ~           |

Initial WorkToDo P4D= INTEG (
      -Start Dev Rate P4D,
            Init Dev P4D)
      ~     lines
      ~           |

Initial WorkToDo P4H= INTEG (
      -HLD Start Rate P4H,
            Init HLD P4H)
      ~     lines
      ~           |

Initial WorkToDo P4T= INTEG (
      -Test Start Rate P4T,
            Init Test P4T)
      ~     lines
      ~           |

InitialDueDate P1D=
      Get XLS Constants('ModelConstants.xls','Project Constants','u2')
      ~     Month
      ~           |

InitialDueDate P1H=
      Get XLS Constants('ModelConstants.xls','Project Constants','t2')
      ~     Month
      ~           |

InitialDueDate P1R=
      Get XLS Constants('ModelConstants.xls','Project Constants','s2')
      ~     Month
      ~           |

InitialDueDate P1T=
      Get XLS Constants('ModelConstants.xls','Project Constants','v2')
      ~     Month
      ~           |

InitialDueDate P2D=
      Get XLS Constants('ModelConstants.xls','Project Constants','u3')
      ~     Month
      ~           |
```

```
InitialDueDate P2H=
      Get XLS Constants('ModelConstants.xls','Project Constants','t3')
      ~     Month
      ~           |

InitialDueDate P2R=
      Get XLS Constants('ModelConstants.xls','Project Constants','s3')
      ~     Month
      ~           |

InitialDueDate P2T=
      Get XLS Constants('ModelConstants.xls','Project Constants','v3')
      ~     Month
      ~           |

InitialDueDate P3D=
      Get XLS Constants('ModelConstants.xls','Project Constants','u4')
      ~     Month
      ~           |

InitialDueDate P3H=
      Get XLS Constants('ModelConstants.xls','Project Constants','t4')
      ~     Month
      ~           |

InitialDueDate P3R=
      Get XLS Constants('ModelConstants.xls','Project Constants','s4')
      ~     Month
      ~           |

InitialDueDate P3T=
      Get XLS Constants('ModelConstants.xls','Project Constants','v4')
      ~     Month
      ~           |

InitialDueDate P4D=
      Get XLS Constants('ModelConstants.xls','Project Constants','u5')
      ~     Month
      ~           |

InitialDueDate P4H=
      Get XLS Constants('ModelConstants.xls','Project Constants','t5')
      ~     Month
      ~           |

InitialDueDate P4R=
      Get XLS Constants('ModelConstants.xls','Project Constants','s5')
      ~     Month
      ~           |

InitialDueDate P4T=
      Get XLS Constants('ModelConstants.xls','Project Constants','v5')
      ~     Month
      ~           |

InitialWorkToDo P1R=
      Get XLS Constants('ModelConstants.xls','Project Constants','o2')
      ~     lines
      ~           |
```

```
InitialWorkToDo P1R Pulse=
      InitialWorkToDo P1R*pulse(TimeToStart P1R,TIME STEP)*(1/TIME STEP)
      ~     lines/Month
      ~           |

InitialWorkToDo P2R=
      Get XLS Constants('ModelConstants.xls','Project Constants','o3')
      ~     lines
      ~           |

InitialWorkToDo P2R Pulse=
      InitialWorkToDo P2R*pulse(TimeToStart P2R,TIME STEP)*(1/TIME STEP)
      ~     lines/Month
      ~           |

InitialWorkToDo P3R=
      Get XLS Constants('ModelConstants.xls','Project Constants','o4')
      ~     lines
      ~           |

InitialWorkToDo P3R Pulse=
      InitialWorkToDo P3R*pulse(TimeToStart P3R,TIME STEP)*(1/TIME STEP)
      ~     lines/Month
      ~           |

InitialWorkToDo P4R=
      Get XLS Constants('ModelConstants.xls','Project Constants','o5')
      ~     lines
      ~           |

InitialWorkToDo P4R Pulse=
      InitialWorkToDo P4R*pulse(TimeToStart P4R,TIME STEP)*(1/TIME STEP)
      ~     lines/Month
      ~           |

Intermediate Advance to Expert Time R=
      Get XLS Constants('ModelConstants.xls','Phase Constants','b3')
      ~     Month
      ~           |

Intermediate Attrition=
      0.1
      ~     fraction
      ~     10 percent / year
      |

Intermediate Skill Effect on Quality=
      Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b21')
      ~     fraction
      ~           |

IRAllocated[project]=
      Allocate By
Priority(IRDesired[project],IRAttractiveness[project],4,1,TotalIR)
      ~     people
      ~           |

IRAttractiveness[one]=
      Attractiveness P1R ~~|
IRAttractiveness[two]=
```

```
	Attractiveness P2R ~~|
IRAttractiveness[three]=
	Attractiveness P3R ~~|
IRAttractiveness[four]=
	Attractiveness P4R
	~	dmnl
	~		|

IRDesired[one]=
	DesiredP1IR ~~|
IRDesired[two]=
	DesiredP2IR ~~|
IRDesired[three]=
	DesiredP3IR ~~|
IRDesired[four]=
	DesiredP4IR
	~	people
	~		|

ITAllocated[project]=
	Allocate By
Priority(ITDesired[project],ITAttractiveness[project],4,1,TotalIT)
	~	people
	~		|

ITAttractiveness[one]=
	Attractiveness P1T ~~|
ITAttractiveness[two]=
	Attractiveness P2T ~~|
ITAttractiveness[three]=
	Attractiveness P3T ~~|
ITAttractiveness[four]=
	Attractiveness P4T
	~	dmnl
	~		|

ITDesired[one]=
	DesiredP1IT ~~|
ITDesired[two]=
	DesiredP2IT ~~|
ITDesired[three]=
	DesiredP3IT ~~|
ITDesired[four]=
	DesiredP4IT
	~	people
	~		|

MaxQuality D=
	Get XLS Constants('ModelConstants.xls','Phase Constants','d7')
	~	fraction
	~	For full model, change to 0.6
	|

MaxQuality H=
	Get XLS Constants('ModelConstants.xls','Phase Constants','c7')
	~	fraction
	~	For full model, change to 0.6
	|

MaxQuality R=
	Get XLS Constants('ModelConstants.xls','Phase Constants','b7')
```

```
	~	fraction
	~	For full model, change to 0.6
	|

MaxQuality T=
	Get XLS Constants('ModelConstants.xls','Phase Constants','e7')
	~	fraction
	~	For full model, change to 0.6
	|

minimum remaining time D=
	Get XLS Constants('ModelConstants.xls','Phase Constants','d4')
	~	months
	~		|

minimum remaining time H=
	Get XLS Constants('ModelConstants.xls','Phase Constants','c4')
	~	Month
	~		|

minimum remaining time R=
	Get XLS Constants('ModelConstants.xls','Phase Constants','b4')
	~	Month
	~		|

minimum remaining time T=
	Get XLS Constants('ModelConstants.xls','Phase Constants','e4')
	~	Month
	~		|

NDAllocated[project]=
	Allocate By
Priority(NDDesired[project],NDAttractiveness[project],4,1,TotalND)
	~	people
	~		|

NDAttractiveness[one]=
	Attractiveness P1D ~~|
NDAttractiveness[two]=
	Attractiveness P2D ~~|
NDAttractiveness[three]=
	Attractiveness P3D ~~|
NDAttractiveness[four]=
	Attractiveness P4D
	~	dmnl
	~		|

NDDesired[one]=
	DesiredP1ND ~~|
NDDesired[two]=
	DesiredP2ND ~~|
NDDesired[three]=
	DesiredP3ND ~~|
NDDesired[four]=
	DesiredP4ND
	~	people
	~		|

NHAllocated[project]=
	Allocate By
Priority(NHDesired[project],NHAttractiveness[project],4,1,TotalNH)
```

```
	~	people
	~		|

NHAttractiveness[one]=
	Attractiveness P1H ~~|
NHAttractiveness[two]=
	Attractiveness P2H ~~|
NHAttractiveness[three]=
	Attractiveness P3H ~~|
NHAttractiveness[four]=
	Attractiveness P4H
	~	dmnl
	~		|

NHDesired[one]=
	DesiredP1NH ~~|
NHDesired[two]=
	DesiredP2NH ~~|
NHDesired[three]=
	DesiredP3NH ~~|
NHDesired[four]=
	DesiredP4NH
	~	people
	~		|

Normal Productivity D=
	Get XLS Constants('ModelConstants.xls','Phase Constants','d8')
	~	lines/(people*Month)
	~		|

Normal Productivity H=
	Get XLS Constants('ModelConstants.xls','Phase Constants','c8')
	~	lines/(people*Month)
	~		|

Normal Productivity R=
	Get XLS Constants('ModelConstants.xls','Phase Constants','b8')
	~	lines/(people*Month)
	~		|

Normal Productivity T=
	Get XLS Constants('ModelConstants.xls','Phase Constants','e8')
	~	lines/(people*Month)
	~		|

Novice Advance to Intermediate Time R=
	Get XLS Constants('ModelConstants.xls','Phase Constants','b2')
	~	Month
	~		|

Novice Attrition=
	0.1
	~	fraction
	~	10 percent / year
	|

Novice Skill Effect on Quality=
	Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b20')
	~	fraction
	~		|
```

```
NRAllocated[project]=
      Allocate By
Priority(NRDesired[project],NRAttractiveness[project],4,1,TotalNR)
      ~     people
      ~           |

NRAttractiveness[one]=
      Attractiveness P1R ~~|
NRAttractiveness[two]=
      Attractiveness P2R ~~|
NRAttractiveness[three]=
      Attractiveness P3R ~~|
NRAttractiveness[four]=
      Attractiveness P4R
      ~     dmnl
      ~           |

NRDesired[one]=
      DesiredP1NR ~~|
NRDesired[two]=
      DesiredP2NR ~~|
NRDesired[three]=
      DesiredP3NR ~~|
NRDesired[four]=
      DesiredP4NR
      ~     people
      ~           |

NTAllocated[project]=
      Allocate By
Priority(NTDesired[project],NTAttractiveness[project],4,1,TotalNT)
      ~     people
      ~           |

NTAttractiveness[one]=
      Attractiveness P1T ~~|
NTAttractiveness[two]=
      Attractiveness P2T ~~|
NTAttractiveness[three]=
      Attractiveness P3T ~~|
NTAttractiveness[four]=
      Attractiveness P4T
      ~     dmnl
      ~           |

NTDesired[one]=
      DesiredP1NT ~~|
NTDesired[two]=
      DesiredP2NT ~~|
NTDesired[three]=
      DesiredP3NT ~~|
NTDesired[four]=
      DesiredP4NT
      ~     people
      ~           |

OverTime f(
      [(0,0)-
(3.40282e+038,2)],(0,0),(1,1),(1.81269,1.5),(2.5,1.5),(10,1.5),(1e+030,1
.5))
```

```
	~	fraction
	~	[(0,0)-(2.5,2)],(0,0),(1,1),(1.81269,1.60526),(2.5,2)\!\!\!
	|

OverTime P1D=
	OverTime f(Indicated overtime P1D)
	~	fraction
	~		|

OverTime P1H=
	OverTime f(Indicated overtime P1H)
	~	fraction
	~		|

OverTime P1R=
	OverTime f(Indicated overtime P1R)
	~	fraction
	~		|

OverTime P1T=
	OverTime f(Indicated overtime P1T)
	~	fraction
	~		|

OverTime P2D=
	OverTime f(Indicated overtime P2D)
	~	fraction
	~		|

OverTime P2H=
	OverTime f(Indicated overtime P2H)
	~	fraction
	~		|

OverTime P2R=
	OverTime f(Indicated overtime P2R)
	~	fraction
	~		|

OverTime P2T=
	OverTime f(Indicated overtime P2T)
	~	fraction
	~		|

OverTime P3D=
	OverTime f(Indicated overtime P3D)
	~	fraction
	~		|

OverTime P3H=
	OverTime f(Indicated overtime P3H)
	~	fraction
	~		|

OverTime P3R=
	OverTime f(Indicated overtime P3R)
	~	fraction
	~		|

OverTime P3T=
	OverTime f(Indicated overtime P3T)
```

```
	~	fraction
	~		|

OverTime P4D=
	OverTime f(Indicated overtime P4D)
	~	fraction
	~		|

OverTime P4H=
	OverTime f(Indicated overtime P4H)
	~	fraction
	~		|

OverTime P4R=
	OverTime f(Indicated overtime P4R)
	~	fraction
	~		|

OverTime P4T=
	OverTime f(Indicated overtime P4T)
	~	fraction
	~		|

P1 Initial Priority=
	Get XLS Constants('ModelConstants.xls','Project Constants','b2')
	~	dmnl
	~		|

P1DDesiredE=
	DesiredRealHeads P1D*RatioED
	~	people
	~		|

P1DDesiredI=
	DesiredRealHeads P1D*RatioID
	~	people
	~		|

P1DDesiredN=
	DesiredRealHeads P1D*RatioND
	~	people
	~		|

P1ED= INTEG (
	P1ED Rate+P1IDtoED Rate-Attrition Rate P1ED,
		StartP1ED)
	~	people
	~		|

P1ED Rate=
	if then else(GapP1ED>0,Min(GapP1ED, ED
Control)/TimeToMoveIn,GapP1ED/TimeToMoveOut)
	~	people/Month
	~		|

P1EH= INTEG (
	P1EH Rate+P1IHtoEH Rate-Attrition Rate P1EH,
		StartP1EH)
	~	people
	~		|
```

```
P1EH Rate=
      if then else(GapP1EH>0,Min(GapP1EH, EH
Control)/TimeToMoveIn,GapP1EH/TimeToMoveOut)
      ~     people/Month
      ~           |

P1ER= INTEG (
      P1ER Rate+P1IRtoER Rate-Attrition Rate P1ER,
            StartP1ER)
      ~     people
      ~           |

P1ER Rate=
      if then else(GapP1ER>0,Min(GapP1ER, ER
Control)/TimeToMoveIn,GapP1ER/TimeToMoveOut)
      ~     people/Month
      ~           |

P1ET Rate=
      if then else(GapP1ET>0,Min(GapP1ET, ET
Control)/TimeToMoveIn,GapP1ET/TimeToMoveOut)
      ~     people/Month
      ~           |

P1HDesiredE=
      DesiredRealHeads P1H*RatioEH
      ~     people
      ~           |

P1HDesiredI=
      DesiredRealHeads P1H*RatioIH
      ~     people
      ~           |

P1HDesiredN=
      DesiredRealHeads P1H*RatioNH
      ~     people
      ~           |

P1ID= INTEG (
      P1ID Rate+P1NDtoID Rate-P1IDtoED Rate-Attrition Rate P1ID,
            StartP1ID)
      ~     people
      ~           |

P1ID Rate=
      if then else(GapP1ID>0,Min(GapP1ID, ID
Control)/TimeToMoveIn,GapP1ID/TimeToMoveOut)
      ~     people/Month
      ~           |

P1IDtoED Rate=
      (P1ID/Intermediate Advance to Expert Time D)*Staffing Gap effect
on learning P1D*Complexity effect on learning P1
      ~     people/Month
      ~           |

P1IH= INTEG (
      P1IH Rate+P1NHtoIH Rate-P1IHtoEH Rate-Attrition Rate P1IH,
            StartP1IH)
      ~     people
```

```
	~		|

P1IH Rate=
	if then else(GapP1IH>0,Min(GapP1IH, IH
Control)/TimeToMoveIn,GapP1IH/TimeToMoveOut)
	~	people/Month
	~		|

P1IHtoEH Rate=
	(P1IH/Intermediate Advance to Expert Time H)*Staffing Gap effect
on learning P1H*Complexity effect on learning P1
	~	people/Month
	~		|

P1IR= INTEG (
	P1IR Rate+P1NRtoIR Rate-P1IRtoER Rate-Attrition Rate P1IR,
		StartP1IR)
	~	people
	~		|

P1IR Rate=
	if then else(GapP1IR>0,Min(GapP1IR, IR
Control)/TimeToMoveIn,GapP1IR/TimeToMoveOut)
	~	people/Month
	~		|

P1IRtoER Rate=
	(P1IR/Intermediate Advance to Expert Time R)*Staffing Gap effect
on learning P1R*Complexity effect on learning P1
	~	people/Month
	~		|

P1IT Rate=
	if then else(GapP1IT>0,Min(GapP1IT, IT
Control)/TimeToMoveIn,GapP1IT/TimeToMoveOut)
	~	people/Month
	~		|

P1ITtoET Rate=
	(P1IT/Intermediate Advance to Expert Time T)*Staffing Gap effect
on learning P1T*Complexity effect on learning P1
	~	people/Month
	~		|

P1ND= INTEG (
	P1ND Rate-P1NDtoID Rate-Attrition Rate P1ND,
		StartP1ND)
	~	people
	~		|

P1ND Rate=
	if then else(GapP1ND>0,Min(GapP1ND, ND
Control)/TimeToMoveIn,GapP1ND/TimeToMoveOut)
	~	people/Month
	~		|

P1NDtoID Rate=
	(P1ND/Novice Advance to Intermediate Time D)*Staffing Gap effect
on learning P1D*Complexity effect on learning P1
	~	people/Month
	~		|
```

```
P1NH= INTEG (
      P1NH Rate-P1NHtoIH Rate-Attrition Rate P1NH,
            StartP1NH)
      ~     people
      ~           |

P1NH Rate=
      if then else(GapP1NH>0,Min(GapP1NH, NH
Control)/TimeToMoveIn,GapP1NH/TimeToMoveOut)
      ~     people/Month
      ~           |

P1NHtoIH Rate=
      (P1NH/Novice Advance to Intermediate Time H)*Staffing Gap effect
on learning P1H*Complexity effect on learning P1
      ~     people/Month
      ~           |

P1NR= INTEG (
      P1NR Rate-P1NRtoIR Rate-Attrition Rate P1NR,
            StartP1NR)
      ~     people
      ~           |

P1NR Rate=
      if then else(GapP1NR>0,Min(GapP1NR, NR
Control)/TimeToMoveIn,GapP1NR/TimeToMoveOut)
      ~     people/Month
      ~           |

P1NRtoIR Rate=
      (P1NR/Novice Advance to Intermediate Time R)*Staffing Gap effect
on learning P1R*Complexity effect on learning P1
      ~     people/Month
      ~           |

P1NT Rate=
      if then else(GapP1NT>0,Min(GapP1NT, NT
Control)/TimeToMoveIn,GapP1NT/TimeToMoveOut)
      ~     people/Month
      ~           |

P1NTtoIT Rate=
      (P1NT/Novice Advance to Intermediate Time T)*Staffing Gap effect
on learning P1T*Complexity effect on learning P1
      ~     people/Month
      ~           |

P1TDesiredE=
      DesiredRealHeads P1T*RatioET
      ~     people
      ~           |

P1TDesiredI=
      DesiredRealHeads P1T*RatioIT
      ~     people
      ~           |

P1TDesiredN=
      DesiredRealHeads P1T*RatioNT
```

```
	~	people
	~		|

P2 Initial Priority=
	Get XLS Constants('ModelConstants.xls','Project Constants','b3')
	~	dmnl
	~		|

P2DDesiredE=
	DesiredRealHeads P2D*RatioED
	~	people
	~		|

P2DDesiredI=
	DesiredRealHeads P2D*RatioID
	~	people
	~		|

P2DDesiredN=
	DesiredRealHeads P2D*RatioND
	~	people
	~		|

P2ED= INTEG (
	P2ED Rate+P2IDtoED Rate-Attrition Rate P2ED,
		StartP2ED)
	~	people
	~		|

P2ED Rate=
	if then else(GapP2ED>0,Min(GapP2ED, ED
Control)/TimeToMoveIn,GapP2ED/TimeToMoveOut)
	~	people/Month
	~		|

P2EH= INTEG (
	P2EH Rate+P2IHtoEH Rate-Attrition Rate P2EH,
		StartP2EH)
	~	people
	~		|

P2EH Rate=
	if then else(GapP2EH>0,Min(GapP2EH, EH
Control)/TimeToMoveIn,GapP2EH/TimeToMoveOut)
	~	people/Month
	~		|

P2ER= INTEG (
	P2ER Rate+P2IRtoER Rate-Attrition Rate P2ER,
		StartP2ER)
	~	people
	~		|

P2ER Rate=
	if then else(GapP2ER>0,Min(GapP2ER, ER
Control)/TimeToMoveIn,GapP2ER/TimeToMoveOut)
	~	people/Month
	~		|

P2ET Rate=
```

```
	if then else(GapP2ET>0,Min(GapP2ET, ET
Control)/TimeToMoveIn,GapP2ET/TimeToMoveOut)
	~	people/Month
	~		|

P2HDesiredE=
	DesiredRealHeads P2H*RatioEH
	~	people
	~		|

P2HDesiredI=
	DesiredRealHeads P2H*RatioIH
	~	people
	~		|

P2HDesiredN=
	DesiredRealHeads P2H*RatioNH
	~	people
	~		|

P2ID= INTEG (
	P2ID Rate+P2NDtoID Rate-P2IDtoED Rate-Attrition Rate P2ID,
		StartP2ID)
	~	people
	~		|

P2ID Rate=
	if then else(GapP2ID>0,Min(GapP2ID, ID
Control)/TimeToMoveIn,GapP2ID/TimeToMoveOut)
	~	people/Month
	~		|

P2IH= INTEG (
	P2IH Rate+P2NHtoIH Rate-P2IHtoEH Rate-Attrition Rate P2IH,
		StartP2IH)
	~	people
	~		|

P2IH Rate=
	if then else(GapP2IH>0,Min(GapP2IH,IH
Control)/TimeToMoveIn,GapP2IH/TimeToMoveOut)
	~	people/Month
	~		|

P2IR= INTEG (
	P2IR Rate+P2NRtoIR Rate-P2IRtoER Rate-Attrition Rate P2IR,
		StartP2IR)
	~	people
	~		|

P2IR Rate=
	if then else(GapP2IR>0,Min(GapP2IR, IR
Control)/TimeToMoveIn,GapP2IR/TimeToMoveOut)
	~	people/Month
	~		|

P2IT Rate=
	if then else(GapP2IT>0,Min(GapP2IT, IT
Control)/TimeToMoveIn,GapP2IT/TimeToMoveOut)
	~	people/Month
	~		|
```

```
P2ND= INTEG (
      P2ND Rate-P2NDtoID Rate-Attrition Rate P2ND,
            StartP2ND)
      ~     people
      ~           |

P2ND Rate=
      if then else(GapP2ND>0,Min(GapP2ND, ND
Control)/TimeToMoveIn,GapP2ND/TimeToMoveOut)
      ~     people/Month
      ~           |

P2NDtoID Rate=
      (P2ND/Novice Advance to Intermediate Time D)*Staffing Gap effect
on learning P2D*Complexity effect on learning P2
      ~     people/Month
      ~           |

P2NH= INTEG (
      P2NH Rate-P2NHtoIH Rate-Attrition Rate P2NH,
            StartP2NH)
      ~     people
      ~           |

P2NH Rate=
      if then else(GapP2NH>0,Min(GapP2NH, NH
Control)/TimeToMoveIn,GapP2NH/TimeToMoveOut)
      ~     people/Month
      ~           |

P2NHtoIH Rate=
      (P2NH/Novice Advance to Intermediate Time H)*Staffing Gap effect
on learning P2H*Complexity effect on learning P2
      ~     people/Month
      ~           |

P2NR= INTEG (
      P2NR Rate-P2NRtoIR Rate-Attrition Rate P2NR,
            StartP2NR)
      ~     people
      ~           |

P2NR Rate=
      if then else(GapP2NR>0,Min(GapP2NR, NR
Control)/TimeToMoveIn,GapP2NR/TimeToMoveOut)
      ~     people/Month
      ~           |

P2NRtoIR Rate=
      (P2NR/Novice Advance to Intermediate Time R)*Staffing Gap effect
on learning P2R*Complexity effect on learning P2
      ~     people/Month
      ~           |

P2NT Rate=
      if then else(GapP2NT>0,Min(GapP2NT, NT
Control)/TimeToMoveIn,GapP2NT/TimeToMoveOut)
      ~     people/Month
      ~           |
```

```
P2NTtoIT Rate=
      (P2NT/Novice Advance to Intermediate Time T)*Staffing Gap effect
on learning P2T*Complexity effect on learning P2
      ~     people/Month
      ~           |

P2RDesiredE=
      DesiredRealHeads P2R*RatioER
      ~     people
      ~           |

P2RDesiredI=
      DesiredRealHeads P2R*RatioIR
      ~     people
      ~           |

P2RDesiredN=
      DesiredRealHeads P2R*RatioNR
      ~     people
      ~           |

P2TDesiredE=
      DesiredRealHeads P2T*RatioET
      ~     people
      ~           |

P2TDesiredI=
      DesiredRealHeads P2T*RatioIT
      ~     people
      ~           |

P2TDesiredN=
      DesiredRealHeads P2T*RatioNT
      ~     people
      ~           |

P3 Initial Priority=
      Get XLS Constants('ModelConstants.xls','Project Constants','b4')
      ~     dmnl
      ~           |

P3DDesiredE=
      DesiredRealHeads P3D*RatioED
      ~     people
      ~           |

P3DDesiredI=
      DesiredRealHeads P3D*RatioID
      ~     people
      ~           |

P3DDesiredN=
      DesiredRealHeads P3D*RatioND
      ~     people
      ~           |

P3ED= INTEG (
      P3ED Rate+P3IDtoED Rate-Attrition Rate P3ED,
            StartP3ED)
      ~     people
      ~           |
```

```
P3ED Rate=
      if then else(GapP3ED>0,Min(GapP3ED, ED
Control)/TimeToMoveIn,GapP3ED/TimeToMoveOut)
      ~     people/Month
      ~           |

P3EH= INTEG (
      P3EH Rate+P3IHtoEH Rate-Attrition Rate P3EH,
            StartP3EH)
      ~     people
      ~           |

P3EH Rate=
      if then else(GapP3EH>0,Min(GapP3EH, EH
Control)/TimeToMoveIn,GapP3EH/TimeToMoveOut)
      ~     people/Month
      ~           |

P3ER= INTEG (
      P3ER Rate+P3IRtoER Rate-Attrition Rate P3ER,
            StartP3ER)
      ~     people
      ~           |

P3ER Rate=
      if then else(GapP3ER>0,Min(GapP3ER, ER
Control)/TimeToMoveIn,GapP3ER/TimeToMoveOut)
      ~     people/Month
      ~           |

P3ET Rate=
      if then else(GapP3ET>0,Min(GapP3ET, ET
Control)/TimeToMoveIn,GapP3ET/TimeToMoveOut)
      ~     people/Month
      ~           |

P3HDesiredE=
      DesiredRealHeads P3H*RatioEH
      ~     people
      ~           |

P3HDesiredI=
      DesiredRealHeads P3H*RatioIH
      ~     people
      ~           |

P3HDesiredN=
      DesiredRealHeads P3H*RatioNH
      ~     people
      ~           |

P3ID= INTEG (
      P3ID Rate+P3NDtoID Rate-P3IDtoED Rate-Attrition Rate P3ID,
            StartP3ID)
      ~     people
      ~           |

P3ID Rate=
      if then else(GapP3ID>0,Min(GapP3ID, ID
Control)/TimeToMoveIn,GapP3ID/TimeToMoveOut)
```

```
	~	people/Month
	~		|

P3IDtoED Rate=
	(P3ID/Intermediate Advance to Expert Time D)*Staffing Gap effect
on learning P3D*Complexity effect on learning P3
	~	people/Month
	~		|

P3IH= INTEG (
	P3IH Rate+P3NHtoIH Rate-P3IHtoEH Rate-Attrition Rate P3IH,
		StartP3IH)
	~	people
	~		|

P3IH Rate=
	if then else(GapP3IH>0,Min(GapP3IH, IH
Control)/TimeToMoveIn,GapP3IH/TimeToMoveOut)
	~	people/Month
	~		|

P3IHtoEH Rate=
	(P3IH/Intermediate Advance to Expert Time H)*Staffing Gap effect
on learning P3H*Complexity effect on learning P3
	~	people/Month
	~		|

P3IR= INTEG (
	P3IR Rate+P3NRtoIR Rate-P3IRtoER Rate-Attrition Rate P3IR,
		StartP3IR)
	~	people
	~		|

P3IR Rate=
	if then else(GapP3IR>0,Min(GapP3IR, IR
Control)/TimeToMoveIn,GapP3IR/TimeToMoveOut)
	~	people/Month
	~		|

P3IRtoER Rate=
	(P3IR/Intermediate Advance to Expert Time R)*Staffing Gap effect
on learning P3R
	~	people/Month
	~		|

P3IT Rate=
	if then else(GapP3IT>0,Min(GapP3IT, IT
Control)/TimeToMoveIn,GapP3IT/TimeToMoveOut)
	~	people/Month
	~		|

P3ITtoET Rate=
	(P3IT/Intermediate Advance to Expert Time T)*Staffing Gap effect
on learning P3T*Complexity effect on learning P3
	~	people/Month
	~		|

P3ND= INTEG (
	P3ND Rate-P3NDtoID Rate-Attrition Rate P3ND,
		StartP3ND)
	~	people
```

```
      ~           |

P3ND Rate=
      if then else(GapP3ND>0,Min(GapP3ND, ND
Control)/TimeToMoveIn,GapP3ND/TimeToMoveOut)
      ~     people/Month
      ~           |

P3NDtoID Rate=
      (P3ND/Novice Advance to Intermediate Time D)*Staffing Gap effect
on learning P3D*Complexity effect on learning P3
      ~     people/Month
      ~           |

P3NH= INTEG (
      P3NH Rate-P3NHtoIH Rate-Attrition Rate P3NH,
            StartP3NH)
      ~     people
      ~           |

P3NH Rate=
      if then else(GapP3NH>0,Min(GapP3NH, NH
Control)/TimeToMoveIn,GapP3NH/TimeToMoveOut)
      ~     people/Month
      ~           |

P3NHtoIH Rate=
      (P3NH/Novice Advance to Intermediate Time H)*Staffing Gap effect
on learning P3H*Complexity effect on learning P3
      ~     people/Month
      ~           |

P3NR= INTEG (
      P3NR Rate-P3NRtoIR Rate-Attrition Rate P3NR,
            StartP3NR)
      ~     people
      ~           |

P3NR Rate=
      if then else(GapP3NR>0,Min(GapP3NR, NR
Control)/TimeToMoveIn,GapP3NR/TimeToMoveOut)
      ~     people/Month
      ~           |

P3NRtoIR Rate=
      (P3NR/Novice Advance to Intermediate Time R)*Staffing Gap effect
on learning P3R*Complexity effect on learning P3
      ~     people/Month
      ~           |

P3NT Rate=
      if then else(GapP3NT>0,Min(GapP3NT, NT
Control)/TimeToMoveIn,GapP3NT/TimeToMoveOut)
      ~     people/Month
      ~           |

P3NTtoIT Rate=
      (P3NT/Novice Advance to Intermediate Time T)*Staffing Gap effect
on learning P3T*Complexity effect on learning P3
      ~     people/Month
      ~           |
```

```
P3RDesiredE=
      DesiredRealHeads P3R*RatioER
      ~     people
      ~           |

P3RDesiredI=
      DesiredRealHeads P3R*RatioIR
      ~     people
      ~           |

P3RDesiredN=
      DesiredRealHeads P3R*RatioNR
      ~     people
      ~           |

P3TDesiredE=
      DesiredRealHeads P3T*RatioET
      ~     people
      ~           |

P3TDesiredI=
      DesiredRealHeads P3T*RatioIT
      ~     people
      ~           |

P3TDesiredN=
      DesiredRealHeads P3T*RatioNT
      ~     people
      ~           |

P4 Initial Priority=
      Get XLS Constants('ModelConstants.xls','Project Constants','b5')
      ~     dmnl
      ~           |

P4DDesiredE=
      DesiredRealHeads P4D*RatioED
      ~     people
      ~           |

P4DDesiredI=
      DesiredRealHeads P4D*RatioID
      ~     people
      ~           |

P4DDesiredN=
      DesiredRealHeads P4D*RatioND
      ~     people
      ~           |

P4ED= INTEG (
      P4ED Rate+P4IDtoED Rate-Attrition Rate P4ED,
            StartP4ED)
      ~     people
      ~           |

P4ED Rate=
      if then else(GapP4ED>0,Min(GapP4ED, ED
Control)/TimeToMoveIn,GapP4ED/TimeToMoveOut)
      ~     people/Month
```

```
      ~           |

P4EH= INTEG (
      P4EH Rate+P4IHtoEH Rate-Attrition Rate P4EH,
            StartP4EH)
      ~     people
      ~           |

P4EH Rate=
      if then else(GapP4EH>0,Min(GapP4EH, EH
Control)/TimeToMoveIn,GapP4EH/TimeToMoveOut)
      ~     people/Month
      ~           |

P4ER= INTEG (
      P4ER Rate+P4IRtoER Rate-Attrition Rate P4ER,
            StartP4ER)
      ~     people
      ~           |

P4ER Rate=
      if then else(GapP4ER>0,Min(GapP4ER, ER
Control)/TimeToMoveIn,GapP4ER/TimeToMoveOut)
      ~     people/Month
      ~           |

P4ET Rate=
      if then else(GapP4ET>0,Min(GapP4ET, ET
Control)/TimeToMoveIn,GapP4ET/TimeToMoveOut)
      ~     people/Month
      ~           |

P4HDesiredE=
      DesiredRealHeads P4H*RatioEH
      ~     people
      ~           |

P4HDesiredI=
      DesiredRealHeads P4H*RatioIH
      ~     people
      ~           |

P4HDesiredN=
      DesiredRealHeads P4H*RatioNH
      ~     people
      ~           |

P4ID= INTEG (
      P4ID Rate+P4NDtoID Rate-P4IDtoED Rate-Attrition Rate P4ID,
            StartP4ID)
      ~     people
      ~           |

P4ID Rate=
      if then else(GapP4ID>0,Min(GapP4ID, ID
Control)/TimeToMoveIn,GapP4ID/TimeToMoveOut)
      ~     people/Month
      ~           |

P4IDtoED Rate=
```

```
      (P4ID/Intermediate Advance to Expert Time D)*Staffing Gap effect
on learning P4D*Complexity effect on learning P4
      ~     people/Month
      ~           |

P4IH= INTEG (
      P4IH Rate+P4NHtoIH Rate-P4IHtoEH Rate-Attrition Rate P4IH,
            StartP4IH)
      ~     people
      ~           |

P4IH Rate=
      if then else(GapP4IH>0,Min(GapP4IH,IH
Control)/TimeToMoveIn,GapP4IH/TimeToMoveOut)
      ~     people/Month
      ~           |

P4IHtoEH Rate=
      (P4IH/Intermediate Advance to Expert Time H)*Staffing Gap effect
on learning P4H*Complexity effect on learning P4
      ~     people/Month
      ~           |

P4IR= INTEG (
      P4IR Rate+P4NRtoIR Rate-P4IRtoER Rate-Attrition Rate P4IR,
            StartP4IR)
      ~     people
      ~           |

P4IR Rate=
      if then else(GapP4IR>0,Min(GapP4IR, IR
Control)/TimeToMoveIn,GapP4IR/TimeToMoveOut)
      ~     people/Month
      ~           |

P4IRtoER Rate=
      (P4IR/Intermediate Advance to Expert Time R)*Staffing Gap effect
on learning P4R*Complexity effect on learning P4
      ~     people/Month
      ~           |

P4IT Rate=
      if then else(GapP4IT>0,Min(GapP4IT, IT
Control)/TimeToMoveIn,GapP4IT/TimeToMoveOut)
      ~     people/Month
      ~           |

P4ITtoET Rate=
      (P4IT/Intermediate Advance to Expert Time T)*Staffing Gap effect
on learning P4T*Complexity effect on learning P4
      ~     people/Month
      ~           |

P4ND= INTEG (
      P4ND Rate-P4NDtoID Rate-Attrition Rate P4ND,
            StartP4ND)
      ~     people
      ~           |

P4ND Rate=
```

```
	if then else(GapP4ND>0,Min(GapP4ND, ND
Control)/TimeToMoveIn,GapP4ND/TimeToMoveOut)
	~	people/Month
	~		|

P4NDtoID Rate=
	(P4ND/Novice Advance to Intermediate Time D)*Staffing Gap effect
on learning P4D*Complexity effect on learning P4
	~	people/Month
	~		|

P4NH= INTEG (
	P4NH Rate-P4NHtoIH Rate-Attrition Rate P4NH,
		StartP4NH)
	~	people
	~		|

P4NH Rate=
	if then else(GapP4NH>0,Min(GapP4NH, NH
Control)/TimeToMoveIn,GapP4NH/TimeToMoveOut)
	~	people/Month
	~		|

P4NHtoIH Rate=
	(P4NH/Novice Advance to Intermediate Time H)*Staffing Gap effect
on learning P4H*Complexity effect on learning P4
	~	people/Month
	~		|

P4NR= INTEG (
	P4NR Rate-P4NRtoIR Rate-Attrition Rate P4NR,
		StartP4NR)
	~	people
	~		|

P4NR Rate=
	if then else(GapP4NR>0,Min(GapP4NR, NR
Control)/TimeToMoveIn,GapP4NR/TimeToMoveOut)
	~	people/Month
	~		|

P4NRtoIR Rate=
	(P4NR/Novice Advance to Intermediate Time R)*Staffing Gap effect
on learning P4R*Complexity effect on learning P4
	~	people/Month
	~		|

P4NT Rate=
	if then else(GapP4NT>0,Min(GapP4NT, NT
Control)/TimeToMoveIn,GapP4NT/TimeToMoveOut)
	~	people/Month
	~		|

P4NTtoIT Rate=
	(P4NT/Novice Advance to Intermediate Time T)*Staffing Gap effect
on learning P4T*Complexity effect on learning P4
	~	people/Month
	~		|

P4RDesiredE=
	DesiredRealHeads P4R*RatioER
```

```
      ~     people
      ~           |

P4RDesiredI=
      DesiredRealHeads P4R*RatioIR
      ~     people
      ~           |

P4RDesiredN=
      DesiredRealHeads P4R*RatioNR
      ~     people
      ~           |

P4TDesiredE=
      DesiredRealHeads P4T*RatioET
      ~     people
      ~           |

P4TDesiredI=
      DesiredRealHeads P4T*RatioIT
      ~     people
      ~           |

P4TDesiredN=
      DesiredRealHeads P4T*RatioNT
      ~     people
      ~           |

PDY P1D=
      ( Normal Productivity D*Fatigue effect PDY P1D)*Complexity effect
on PDY P1
      ~     lines/(people*Month)
      ~           |

PDY P1T=
       Normal Productivity T*Fatigue effect PDY P1T*Complexity effect on
PDY P1
      ~     lines/people/Month
      ~           |

PDY P2D=
       Normal Productivity D*Fatigue effect PDY P2D*Complexity effect on
PDY P2
      ~     lines/(people*Month)
      ~           |

PDY P2H=
       Normal Productivity H*Fatigue effect PDY P2H*Complexity effect on
PDY P2
      ~     lines/(people*Month)
      ~           |

PDY P2R=
       Normal Productivity R*Fatigue effect PDY P2R*Complexity effect on
PDY P2
      ~     lines/(people*Month)
      ~           |

PDY P2T=
       Normal Productivity T*Fatigue effect PDY P2T*Complexity effect on
PDY P2
```

```
	~	lines/(people*Month)
	~		|

PDY P3D=
	Normal Productivity D*Fatigue effect PDY P3D*Complexity effect on
PDY P3
	~	lines/people/Month
	~		|

PDY P3H=
	Normal Productivity H*Fatigue effect PDY P3H*Complexity effect on
PDY P3
	~	lines/(people*Month)
	~		|

PDY P3R=
	Normal Productivity R*Fatigue effect PDY P3R*Complexity effect on
PDY P3
	~	lines/(people*Month)
	~		|

PDY P3T=
	Normal Productivity T*Fatigue effect PDY P3T*Complexity effect on
PDY P3
	~	lines/people/Month
	~		|

PDY P4D=
	Normal Productivity D*Fatigue effect PDY P4D*Complexity effect on
PDY P4
	~	lines/(people*Month)
	~		|

PDY P4H=
	Normal Productivity H*Fatigue effect PDY P4H*Complexity effect on
PDY P4
	~	lines/(people*Month)
	~		|

PDY P4R=
	Normal Productivity R*Fatigue effect PDY P4R*Complexity effect on
PDY P4
	~	lines/people/Month
	~		|

PDY P4T=
	Normal Productivity T*Fatigue effect PDY P4T*Complexity effect on
PDY P4
	~	lines/(people*Month)
	~		|

Percvd PDY P1D =INTEG((PDY P1D - Percvd PDY P1D)/TimeToPercvPDY D,Normal
Productivity D\
		)
	~	lines/(people*Month)
	~		|

Percvd PDY P1H =INTEG((PDY P1H - Percvd PDY P1H)/TimeToPercvPDY H,Normal
Productivity H\
		)
	~	lines/(people*Month)
```

```
	~		|

Percvd PDY P1R =INTEG((PDY P1R - Percvd PDY P1R)/TimeToPercvPDY R,Normal
Productivity R\
		)
	~	lines/(people*Month)
	~		|

Percvd PDY P1T =INTEG((PDY P1T - Percvd PDY P1T)/TimeToPercvPDY T,Normal
Productivity T\
		)
	~	lines/(people*Month)
	~		|

Priority effect on attractiveness f(
	[(0,0)-(10,10)],(0,0),(10,10))
	~	dmnl
	~		|

Priority effect on attractiveness P1D=
	Priority effect on attractiveness f(P1 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P1H=
	Priority effect on attractiveness f(P1 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P1R=
	Priority effect on attractiveness f(P1 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P1T=
	Priority effect on attractiveness f(P1 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P2D=
	Priority effect on attractiveness f(P2 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P2H=
	Priority effect on attractiveness f(P2 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P2R=
	Priority effect on attractiveness f(P2 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P2T=
	Priority effect on attractiveness f(P2 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P3D=
	Priority effect on attractiveness f(P3 Initial Priority)
```

```
	~	dmnl
	~		|

Priority effect on attractiveness P3H=
	Priority effect on attractiveness f(P3 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P3R=
	Priority effect on attractiveness f(P3 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P3T=
	Priority effect on attractiveness f(P3 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P4D=
	Priority effect on attractiveness f(P4 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P4H=
	Priority effect on attractiveness f(P4 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P4R=
	Priority effect on attractiveness f(P4 Initial Priority)
	~	dmnl
	~		|

Priority effect on attractiveness P4T=
	Priority effect on attractiveness f(P4 Initial Priority)
	~	dmnl
	~		|

Priority Weight=
	Get XLS Constants('ModelConstants.xls','Portfolio Constants','b2')
	~	dmnl
	~		|

project:
	one,two,three,four
	~
	~		|

Qual P1D=
	Min(1, MaxQuality D*Fatigue effect qual P1D*Average Skill Effect
on Quality P1D*Complexity effect on quality P1\
		)
	~	fraction
	~		|

Qual P1H=
	Min(1, MaxQuality H*Fatigue effect qual P1H*Average Skill Effect
on Quality P1H*Complexity effect on quality P1\
		)
	~	fraction
	~		|
```

```
Qual P1T=
      Min(1, MaxQuality T*Fatigue effect qual P1T*Average Skill Effect
on Quality P1T*Complexity effect on quality P1\
            )
      ~     fraction
      ~           |

Remaining Time P1D=
      DueDate P1D - Time
      ~     Month
      ~           |

Remaining Time P1H=
      DueDate P1H - Time
      ~     Month
      ~           |

Remaining Time P1R=
      DueDate P1R - Time
      ~     Month
      ~           |

Remaining Time P1T=
      DueDate P1T - Time
      ~     Month
      ~           |

Remaining Time P2D=
      DueDate P2D - Time
      ~     Month
      ~           |

Remaining Time P2H=
      DueDate P2H - Time
      ~     Month
      ~           |

Remaining Time P2R=
      DueDate P2R - Time
      ~     Month
      ~           |

Remaining Time P2T=
      DueDate P2T - Time
      ~     Month
      ~           |

Remaining Time P3D=
      DueDate P3D - Time
      ~     Month
      ~           |

Remaining Time P3H=
      DueDate P3H - Time
      ~     Month
      ~           |

Remaining Time P3R=
      DueDate P3R - Time
      ~     Month
```

```
	~		|

Remaining Time P3T=
	DueDate P3T - Time
	~	Month
	~		|

Remaining Time P4D=
	DueDate P4D - Time
	~	Month
	~		|

Remaining Time P4H=
	DueDate P4H - Time
	~	Month
	~		|

Remaining Time P4R=
	DueDate P4R - Time
	~	Month
	~		|

Remaining Time P4T=
	DueDate P4T - Time
	~	Month
	~		|

Staffing Gap effect on attractiveness f(
	[(0,0)-
(1e+009,1)],(0,1),(0.5,0.9),(1,0.8),(2,0.5),(4,0.1),(10,0),(1e+009,0))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P1D=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P1D,DesiredPeople P1D))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P1H=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P1H,DesiredPeople P1H))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P1R=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P1R,DesiredPeople P1R))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P1T=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P1T,DesiredPeople P1T))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P2D=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P2D,DesiredPeople P2D))
	~	dmnl
```

```
	~		|

Staffing Gap effect on attractiveness P2H=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P2H,DesiredPeople P2H))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P2R=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P2R,DesiredPeople P2R))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P2T=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P2T,DesiredPeople P2T))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P3D=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P3D,DesiredPeople P3D))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P3H=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P3H,DesiredPeople P3H))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P3R=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P3R,DesiredPeople P3R))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P3T=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P3T,DesiredPeople P3T))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P4D=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P4D,DesiredPeople P4D))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P4H=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P4H,DesiredPeople P4H))
	~	dmnl
	~		|

Staffing Gap effect on attractiveness P4R=
	Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P4R,DesiredPeople P4R))
	~	dmnl
	~		|
```

```
Staffing Gap effect on attractiveness P4T=
      Staffing Gap effect on attractiveness f(ZIDZ(Workforce
P4T,DesiredPeople P4T))
      ~     dmnl
      ~           |

Staffing Gap Weight=
      Get XLS Constants('ModelConstants.xls','Portfolio Constants','b4')
      ~     dmnl
      ~           |

Start Dev Rate P1D=
      if then else(Initial WorkToDo P1D>0,(Doing right P1H/Init HLD
P1H)*Init Dev P1D,0)
      ~     lines/Month
      ~           |

Start Dev Rate P2D=
      if then else(Initial WorkToDo P2D>0,(Doing right P2H/Init HLD
P2H)*Init Dev P2D,0)
      ~     lines/Month
      ~           |

Start Dev Rate P3D=
      if then else(Initial WorkToDo P3D>0,(Doing right P3H/Init HLD
P3H)*Init Dev P3D,0)
      ~     lines/Month
      ~           |

Start Dev Rate P4D=
      if then else(Initial WorkToDo P4D>0,(Doing right P4H/Init HLD
P4H)*Init Dev P4D,0)
      ~     lines/Month
      ~           |

StartP1ED=
      Get XLS Constants('ModelConstants.xls','Project Constants','k2')
      ~     people
      ~           |

StartP1EH=
      Get XLS Constants('ModelConstants.xls','Project Constants','h2')
      ~     people
      ~           |

StartP1ER=
      Get XLS Constants('ModelConstants.xls','Project Constants','e2')
      ~     people
      ~           |

StartP1ET=
      Get XLS Constants('ModelConstants.xls','Project Constants','n2')
      ~     people
      ~           |

StartP1ID=
      Get XLS Constants('ModelConstants.xls','Project Constants','j2')
      ~     people
      ~           |
```

```
StartP1IH=
      Get XLS Constants('ModelConstants.xls','Project Constants','g2')
      ~     people
      ~           |

StartP1IR=
      Get XLS Constants('ModelConstants.xls','Project Constants','d2')
      ~     people
      ~           |

StartP1IT=
      Get XLS Constants('ModelConstants.xls','Project Constants','m2')
      ~     people
      ~           |

StartP1ND=
      Get XLS Constants('ModelConstants.xls','Project Constants','i2')
      ~     people
      ~           |

StartP1NH=
      Get XLS Constants('ModelConstants.xls','Project Constants','f2')
      ~     people
      ~           |

StartP1NR=
      Get XLS Constants('ModelConstants.xls','Project Constants','c2')
      ~     people
      ~           |

StartP1NT=
      Get XLS Constants('ModelConstants.xls','Project Constants','l2')
      ~     people
      ~           |

StartP2ED=
      Get XLS Constants('ModelConstants.xls','Project Constants','k3')
      ~     people
      ~           |

StartP2EH=
      Get XLS Constants('ModelConstants.xls','Project Constants','h3')
      ~     people
      ~           |

StartP2ER=
      Get XLS Constants('ModelConstants.xls','Project Constants','e3')
      ~     people
      ~           |

StartP2ET=
      Get XLS Constants('ModelConstants.xls','Project Constants','n3')
      ~     people
      ~           |

StartP2ID=
      Get XLS Constants('ModelConstants.xls','Project Constants','j3')
      ~     people
      ~           |

StartP2IH=
```

```
      Get XLS Constants('ModelConstants.xls','Project Constants','g3')
      ~     people
      ~           |

StartP2IR=
      Get XLS Constants('ModelConstants.xls','Project Constants','d3')
      ~     people
      ~           |

StartP2IT=
      Get XLS Constants('ModelConstants.xls','Project Constants','m3')
      ~     people
      ~           |

StartP2ND=
      Get XLS Constants('ModelConstants.xls','Project Constants','i3')
      ~     people
      ~           |

StartP2NH=
      Get XLS Constants('ModelConstants.xls','Project Constants','f3')
      ~     people
      ~           |

StartP2NR=
      Get XLS Constants('ModelConstants.xls','Project Constants','c3')
      ~     people
      ~           |

StartP2NT=
      Get XLS Constants('ModelConstants.xls','Project Constants','l3')
      ~     people
      ~           |

StartP3ED=
      Get XLS Constants('ModelConstants.xls','Project Constants','k4')
      ~     people
      ~           |

StartP3EH=
      Get XLS Constants('ModelConstants.xls','Project Constants','h4')
      ~     people
      ~           |

StartP3ER=
      Get XLS Constants('ModelConstants.xls','Project Constants','e4')
      ~     people
      ~           |

StartP3ET=
      Get XLS Constants('ModelConstants.xls','Project Constants','n4')
      ~     people
      ~           |

StartP3ID=
      Get XLS Constants('ModelConstants.xls','Project Constants','j4')
      ~     people
      ~           |

StartP3IH=
      Get XLS Constants('ModelConstants.xls','Project Constants','g4')
```

```
	~	people
	~		|

StartP3IR=
	Get XLS Constants('ModelConstants.xls','Project Constants','d4')
	~	people
	~		|

StartP3IT=
	Get XLS Constants('ModelConstants.xls','Project Constants','m4')
	~	people
	~		|

StartP3ND=
	Get XLS Constants('ModelConstants.xls','Project Constants','i4')
	~	people
	~		|

StartP3NH=
	Get XLS Constants('ModelConstants.xls','Project Constants','f4')
	~	people
	~		|

StartP3NR=
	Get XLS Constants('ModelConstants.xls','Project Constants','c4')
	~	people
	~		|

StartP3NT=
	Get XLS Constants('ModelConstants.xls','Project Constants','l4')
	~	people
	~		|

StartP4ED=
	Get XLS Constants('ModelConstants.xls','Project Constants','k5')
	~	people
	~		|

StartP4EH=
	Get XLS Constants('ModelConstants.xls','Project Constants','h5')
	~	people
	~		|

StartP4ER=
	Get XLS Constants('ModelConstants.xls','Project Constants','e5')
	~	people
	~		|

StartP4ET=
	Get XLS Constants('ModelConstants.xls','Project Constants','n5')
	~	people
	~		|

StartP4ID=
	Get XLS Constants('ModelConstants.xls','Project Constants','j5')
	~	people
	~		|

StartP4IH=
	Get XLS Constants('ModelConstants.xls','Project Constants','g5')
	~	people
```

```
	~		|

StartP4IR=
	Get XLS Constants('ModelConstants.xls','Project Constants','d5')
	~	people
	~		|

StartP4IT=
	Get XLS Constants('ModelConstants.xls','Project Constants','m5')
	~	people
	~		|

StartP4ND=
	Get XLS Constants('ModelConstants.xls','Project Constants','i5')
	~	people
	~		|

StartP4NH=
	Get XLS Constants('ModelConstants.xls','Project Constants','f5')
	~	people
	~		|

StartP4NR=
	Get XLS Constants('ModelConstants.xls','Project Constants','c5')
	~	people
	~		|

StartP4NT=
	Get XLS Constants('ModelConstants.xls','Project Constants','l5')
	~	people
	~		|

Test Start Rate P1T=
	if then else(Initial WorkToDo P1T>0,(Doing right P1D/Init Dev
P1D)*Init Test P1T,0)
	~	lines/Month
	~		|

Test Start Rate P2T=
	if then else(Initial WorkToDo P2T>0,(Doing right P2D/Init Dev
P2D)*Init Test P2T,0)
	~	lines/Month
	~		|

Test Start Rate P3T=
	if then else(Initial WorkToDo P3T>0,(Doing right P3D/Init Dev
P3D)*Init Test P3T,0)
	~	lines/Month
	~		|

Test Start Rate P4T=
	if then else(Initial WorkToDo P4T>0,(Doing right P4D/Init Dev
P4D)*Init Test P4T,0)
	~	lines/Month
	~		|

Time for Attrition=
	Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b13')
	~	Month
	~		|
```

```
Time to retire=
      1
      ~     Month
      ~           |

TimeToGetFatigued D=
      Get XLS Constants('ModelConstants.xls','Phase Constants','d5')
      ~     Month
      ~     Should be 0.25 or so?
      |

TimeToGetFatigued H=
      Get XLS Constants('ModelConstants.xls','Phase Constants','c5')
      ~     Month
      ~     Should be 0.25 or so?
      |

TimeToGetFatigued R=
      Get XLS Constants('ModelConstants.xls','Phase Constants','b5')
      ~     Month
      ~     Should be 0.25 or so?
      |

TimeToGetFatigued T=
      Get XLS Constants('ModelConstants.xls','Phase Constants','e5')
      ~     Month
      ~     Should be 0.25 or so?
      |

TimeToMoveIn=
      Get XLS Constants('ModelConstants.xls','Portfolio Constants','b9')
      ~     Month
      ~           |

TimeToMoveOut=
      Get XLS Constants('ModelConstants.xls','Portfolio
Constants','b10')
      ~     Month
      ~           |

TimeToPercvPDY D=
      Get XLS Constants('ModelConstants.xls','Phase Constants','d6')
      ~     Month
      ~           |

TimeToPercvPDY H=
      Get XLS Constants('ModelConstants.xls','Phase Constants','c6')
      ~     Month
      ~           |

TimeToPercvPDY R=
      Get XLS Constants('ModelConstants.xls','Phase Constants','b6')
      ~     Month
      ~           |

TimeToPercvPDY T=
      Get XLS Constants('ModelConstants.xls','Phase Constants','e6')
      ~     Month
      ~           |
```

```
TimeToStart P1R=
      Get XLS Constants('ModelConstants.xls','Project Constants','w2')
      ~     Month
      ~           |

TimeToStart P2R=
      Get XLS Constants('ModelConstants.xls','Project Constants','w3')
      ~     Month
      ~           |

TimeToStart P3R=
      Get XLS Constants('ModelConstants.xls','Project Constants','w4')
      ~     Month
      ~           |

TimeToStart P4R=
      Get XLS Constants('ModelConstants.xls','Project Constants','w5')
      ~     Month
      ~           |

Total Attrition ED= INTEG (
      Attrition Rate P1ED+Attrition Rate P2ED+Attrition Rate
P3ED+Attrition Rate P4ED,
            0)
      ~     people
      ~           |

Total Attrition EH= INTEG (
      Attrition Rate P1EH+Attrition Rate P2EH+Attrition Rate
P3EH+Attrition Rate P4EH,
            0)
      ~     people
      ~           |

Total Attrition ID= INTEG (
      Attrition Rate P1ID+Attrition Rate P2ID+Attrition Rate
P3ID+Attrition Rate P4ID,
            0)
      ~     people
      ~           |

Total Attrition IH= INTEG (
      Attrition Rate P1IH+Attrition Rate P2IH+Attrition Rate
P3IH+Attrition Rate P4IH,
            0)
      ~     people
      ~           |

Total Attrition IR= INTEG (
      Attrition Rate P1IR+Attrition Rate P2IR+Attrition Rate
P3IR+Attrition Rate P4IR,
            0)
      ~     people
      ~           |

Total Attrition NH= INTEG (
      Attrition Rate P1NH+Attrition Rate P2NH+Attrition Rate
P3NH+Attrition Rate P4NH,
            0)
      ~     people
      ~           |
```

```
Total Attrition NR= INTEG (
      Attrition Rate P1NR+Attrition Rate P2NR+Attrition Rate
P3NR+Attrition Rate P4NR,
            0)
      ~     people
      ~           |

Total Attrition ER= INTEG (
      Attrition Rate P1ER+Attrition Rate P2ER+Attrition Rate
P3ER+Attrition Rate P4ER,
            0)
      ~     people
      ~           |

Total Attritition ND= INTEG (
      Attrition Rate P1ND+Attrition Rate P2ND+Attrition Rate
P3ND+Attrition Rate P4ND,
            0)
      ~     people
      ~           |

TotalDesiredP1=
      DesiredP1NR+DesiredP1IR+DesiredP1ER+DesiredP1NH+DesiredP1IH+Desire
dP1EH+DesiredP1ND+\
            DesiredP1ID+DesiredP1ED+DesiredP1NT+DesiredP1IT+DesiredP1ET
      ~     people
      ~           |

TotalDesiredP2=
      DesiredP2NR+DesiredP2IR+DesiredP2ER+DesiredP2NH+DesiredP2IH+Desire
dP2EH+DesiredP2ND+\
            DesiredP2ID+DesiredP2ED+DesiredP2NT+DesiredP2IT+DesiredP2ET
      ~     people
      ~           |

TotalDesiredP3=
      DesiredP3NR+DesiredP3IR+DesiredP3ER+DesiredP3NH+DesiredP3IH+Desire
dP3EH+DesiredP3ND+\
            DesiredP3ID+DesiredP3ED+DesiredP3NT+DesiredP3IT+DesiredP3ET
      ~     people
      ~           |

TotalDesiredP4=
      DesiredP4NR+DesiredP4IR+DesiredP4ER+DesiredP4NH+DesiredP4IH+Desire
dP4EH+DesiredP4ND+\
            DesiredP4ID+DesiredP4ED+DesiredP4NT+DesiredP4IT+DesiredP4ET
      ~     people
      ~           |

TotalED=
      (P1ED+P2ED+ED Control+P3ED+P4ED)
      ~     people
      ~           |

TotalEH=
      (P1EH+P2EH+EH Control+P3EH+P4EH)
      ~     people
      ~           |

TotalER=
```

```
	(P1ER+P2ER+ER Control+P3ER+P4ER)
	~	people
	~		|

TotalET=
	(P1ET+P2ET+ET Control+P3ET+P4ET)
	~	people
	~		|

TotalGap D=
	SUM(NDDesired[project!])-TotalND+SUM(IDDesired[project!])-
TotalID+SUM(EDDesired[project\
		!])-TotalED
	~	people
	~		|

TotalGap H=
	SUM(NHDesired[project!])-TotalNH+SUM(IHDesired[project!])-
TotalIH+SUM(EHDesired[project\
		!])-TotalEH
	~	people
	~		|

TotalGap R=
	SUM(NRDesired[project!])-TotalNR+SUM(IRDesired[project!])-
TotalIR+SUM(ERDesired[project\
		!])-TotalER
	~	people
	~		|

TotalID=
	P1ID+P2ID+ID Control+P3ID+P4ID
	~	people
	~		|

TotalIH=
	P1IH+P2IH+IH Control+P3IH+P4IH
	~	people
	~		|

TotalIT=
	P1IT+P2IT+IT Control+P3IT+P4IT
	~	people
	~		|

TotalND=
	P1ND+P2ND+ND Control+P3ND+P4ND
	~	people
	~		|

TotalNH=
	P1NH+P2NH+NH Control+P3NH+P4NH
	~	people
	~		|

TotalNR=
	P1NR+P2NR+NR Control+P3NR+P4NR
	~	people
	~		|

TotalNT=
```

```
	P1NT+P2NT+NT Control+P3NT+P4NT
	~	people
	~		|

TotalP1=
	P1NR+P1IR+P1ER+P1NH+P1IH+P1EH+P1ND+P1ID+P1ED+P1NT+P1IT+P1ET
	~	people
	~		|

TotalP3=
	P3NR+P3IR+P3ER+P3NH+P3IH+P3EH+P3ND+P3ID+P3ED+P3NT+P3IT+P3ET
	~	people
	~		|

TotalP4=
	P4NR+P4IR+P4ER+P4NH+P4IH+P4EH+P4ND+P4ID+P4ED+P4NT+P4IT+P4ET
	~	people
	~		|

Workforce P1D=
	if then else(P1ND*NoviceMultiplier P1D+P1ID*IntMultiplier
P1D+P1ED*ExpertMultiplier P1D\
		<0.01,0.01,P1ND*NoviceMultiplier P1D+P1ID*IntMultiplier
P1D+P1ED*ExpertMultiplier P1D\
		)
	~	people
	~		|

Workforce P1H=
	if then else(P1NH*NoviceMultiplier P1H+P1IH*IntMultiplier
P1H+P1EH*ExpertMultiplier P1H\
		<0.01,0.01,P1NH*NoviceMultiplier P1H+P1IH*IntMultiplier
P1H+P1EH*ExpertMultiplier P1H\
		)
	~	people
	~		|

Workforce P1R=
	if then else(P1NR*NoviceMultiplier P1R+P1IR*IntMultiplier
P1R+P1ER*ExpertMultiplier P1R\
		<0.01,0.01,P1NR*NoviceMultiplier P1R+P1IR*IntMultiplier
P1R+P1ER*ExpertMultiplier P1R\
		)
	~	people
	~		|

Workforce P1T=
	if then else(P1NT*NoviceMultiplier P1T+P1IT*IntMultiplier
P1T+P1ET*ExpertMultiplier P1T\
		<0.01,0.01,P1NT*NoviceMultiplier P1T+P1IT*IntMultiplier
P1T+P1ET*ExpertMultiplier P1T\
		)
	~	people
	~		|

Workforce P2D=
	if then else(P2ND*NoviceMultiplier P2D+P2ID*IntMultiplier
P2D+P2ED*ExpertMultiplier P2D\
		<0.01,0.01,P2ND*NoviceMultiplier P2D+P2ID*IntMultiplier
P2D+P2ED*ExpertMultiplier P2D\
		)
```

```
	~	people
	~		|

Workforce P2H=
	if then else(P2NH*NoviceMultiplier P2H+P2IH*IntMultiplier
P2H+P2EH*ExpertMultiplier P2H\
		<0.01,0.01,P2NH*NoviceMultiplier P2H+P2IH*IntMultiplier
P2H+P2EH*ExpertMultiplier P2H\
		)
	~	people
	~		|

Workforce P2R=
	if then else(P2NR*NoviceMultiplier P2R+P2IR*IntMultiplier
P2R+P2ER*ExpertMultiplier P2R\
		<0.01,0.01,P2NR*NoviceMultiplier P2R+P2IR*IntMultiplier
P2R+P2ER*ExpertMultiplier P2R\
		)
	~	people
	~		|

Workforce P2T=
	if then else(P2NT*NoviceMultiplier P2T+P2IT*IntMultiplier
P2T+P2ET*ExpertMultiplier P2T\
		<0.01,0.01,P2NT*NoviceMultiplier P2T+P2IT*IntMultiplier
P2T+P2ET*ExpertMultiplier P2T\
		)
	~	people
	~		|

Workforce P3D=
	if then else(P3ND*NoviceMultiplier P3D+P3ID*IntMultiplier
P3D+P3ED*ExpertMultiplier P3D\
		<0.01,0.01,P3ND*NoviceMultiplier P3D+P3ID*IntMultiplier
P3D+P3ED*ExpertMultiplier P3D\
		)
	~	people
	~		|

Workforce P3H=
	if then else(P3NH*NoviceMultiplier P3H+P3IH*IntMultiplier
P3H+P3EH*ExpertMultiplier P3H\
		<0.01,0.01,P3NH*NoviceMultiplier P3H+P3IH*IntMultiplier
P3H+P3EH*ExpertMultiplier P3H\
		)
	~	people
	~		|

Workforce P3R=
	if then else(P3NR*NoviceMultiplier P3R+P3IR*IntMultiplier
P3R+P3ER*ExpertMultiplier P3R\
		<0.01,0.01,P3NR*NoviceMultiplier P3R+P3IR*IntMultiplier
P3R+P3ER*ExpertMultiplier P3R\
		)
	~	people
	~		|

Workforce P3T=
	if then else(P3NT*NoviceMultiplier P3T+P3IT*IntMultiplier
P3T+P3ET*ExpertMultiplier P3T\
```

```
		<0.01,0.01,P3NT*NoviceMultiplier P3T+P3IT*IntMultiplier
P3T+P3ET*ExpertMultiplier P3T\
		)
	~	people
	~		|

Workforce P4D=
	if then else(P4ND*NoviceMultiplier P4D+P4ID*IntMultiplier
P4D+P4ED*ExpertMultiplier P4D\
		<0.01,0.01,P4ND*NoviceMultiplier P4D+P4ID*IntMultiplier
P4D+P4ED*ExpertMultiplier P4D\
		)
	~	people
	~		|

Workforce P4H=
	if then else(P4NH*NoviceMultiplier P4H+P4IH*IntMultiplier
P4H+P4EH*ExpertMultiplier P4H\
		<0.01,0.01,P4NH*NoviceMultiplier P4H+P4IH*IntMultiplier
P4H+P4EH*ExpertMultiplier P4H\
		)
	~	people
	~		|

Workforce P4R=
	if then else(P4NR*NoviceMultiplier P4R+P4IR*IntMultiplier
P4R+P4ER*ExpertMultiplier P4R\
		<0.01,0.01,P4NR*NoviceMultiplier P4R+P4IR*IntMultiplier
P4R+P4ER*ExpertMultiplier P4R\
		)
	~	people
	~		|

Workforce P4T=
	if then else(P4NT*NoviceMultiplier P4T+P4IT*IntMultiplier
P4T+P4ET*ExpertMultiplier P4T\
		<0.01,0.01,P4NT*NoviceMultiplier P4T+P4IT*IntMultiplier
P4T+P4ET*ExpertMultiplier P4T\
		)
	~	people
	~		|

WorkToDo P1D= INTEG (
	FindBugs P1D-Doing P1D+Start Dev Rate P1D,
		0)
	~	lines
	~		|

WorkToDo P1H= INTEG (
	FindBugs P1H-Doing P1H+HLD Start Rate P1H,
		0)
	~	lines
	~		|

WorkToDo P1R= INTEG (
	FindBugs P1R-Doing P1R+InitialWorkToDo P1R Pulse,
		0)
	~	lines
	~		|

WorkToDo P1T= INTEG (
```

```
      FindBugs P1T-Doing P1T+Test Start Rate P1T,
            0)
      ~     lines
      ~           |

WorkToDo P2D= INTEG (
      FindBugs P2D-Doing P2D+Start Dev Rate P2D,
            0)
      ~     lines
      ~           |

WorkToDo P2H= INTEG (
      FindBugs P2H-Doing P2H+HLD Start Rate P2H,
            0)
      ~     lines
      ~           |

WorkToDo P2R= INTEG (
      FindBugs P2R-Doing P2R+InitialWorkToDo P2R Pulse,
            0)
      ~     lines
      ~           |

WorkToDo P2T= INTEG (
      FindBugs P2T-Doing P2T+Test Start Rate P2T,
            0)
      ~     lines
      ~           |

WorkToDo P3D= INTEG (
      FindBugs P3D-Doing P3D+Start Dev Rate P3D,
            0)
      ~     lines
      ~           |

WorkToDo P3H= INTEG (
      FindBugs P3H-Doing P3H+HLD Start Rate P3H,
            0)
      ~     lines
      ~           |

WorkToDo P3R= INTEG (
      FindBugs P3R-Doing P3R+InitialWorkToDo P3R Pulse,
            0)
      ~     lines
      ~           |

WorkToDo P3T= INTEG (
      FindBugs P3T-Doing P3T+Test Start Rate P3T,
            0)
      ~     lines
      ~           |

WorkToDo P4D= INTEG (
      FindBugs P4D-Doing P4D+Start Dev Rate P4D,
            0)
      ~     lines
      ~           |

WorkToDo P4H= INTEG (
      FindBugs P4H-Doing P4H+HLD Start Rate P4H,
```

```
		0)
	~	lines
	~		|

WorkToDo P4R= INTEG (
	FindBugs P4R-Doing P4R+InitialWorkToDo P4R Pulse,
		0)
	~	lines
	~		|

WorkToDo P4T= INTEG (
	FindBugs P4T-Doing P4T+Test Start Rate P4T,
		0)
	~	lines
	~		|

********************************************************
	.Control
********************************************************~
		Simulation Control Parameters
	|

AllocatedP4NR=
	NRAllocated[four]
	~	people
	~		|

FINAL TIME  = 100
	~	Month
	~	The final time for the simulation.
	|

GapP2IR=
	AllocatedP2IR-P2IR
	~	people
	~		|

INITIAL TIME  = 0
	~	Month
	~	The initial time for the simulation.
	|

P2IDtoED Rate=
	(P2ID/Intermediate Advance to Expert Time D)*Staffing Gap effect
on learning P2D*Complexity effect on learning P2
	~	people/Month
	~		|

P2IHtoEH Rate=
	(P2IH/Intermediate Advance to Expert Time H)*Staffing Gap effect
on learning P2H*Complexity effect on learning P2
	~	people/Month
	~		|

P2IRtoER Rate=
	(P2IR/Intermediate Advance to Expert Time R)*Staffing Gap effect
on learning P2R*Complexity effect on learning P2
	~	people/Month
	~		|

P2ITtoET Rate=
```

```
      (P2IT/Intermediate Advance to Expert Time T)*Staffing Gap effect
on learning P2T*Complexity effect on learning P2
      ~     people/Month
      ~           |

SAVEPER  =
        TIME STEP
      ~     Month
      ~     The frequency with which output is stored.
      |

TIME STEP  = 0.0625
      ~     Month
      ~     The time step for the simulation.
      |

TotalIR=
      P1IR+P2IR+IR Control+P3IR+P4IR
      ~     people
      ~           |

TotalP2=
      P2NR+P2IR+P2ER+P2NH+P2IH+P2EH+P2ND+P2ID+P2ED+P2NT+P2IT+P2ET
      ~     people
      ~           |
```

(This page intentionally left blank for duplex printing.)

Appendix C
The System Dynamics Method[30]

The system dynamics process comprises five steps. The steps are:

1. Problem Definition
   a. List of Variables
      List all of the variables that are believed to be important to the problem.
   b. Reference Modes
      Graph the behavior of the most important variables that characterize the problem.
   c. Problem Statement
      Phrase the true concern in terms of the reference modes.

2. Momentum Policies
   The policies that would be implemented immediately in order to solve the problem. These are recorded early and may be used to suggest tests or directions of inquiry.

3. Dynamic Hypotheses (Causal Loop Diagrams)
   a. A causal map or loops that describe the feedback processes capable of generating the patterns of behavior identified in the reference modes. These loops are not intended to represent descriptions of the system dynamics model at the detailed level of the equations.
      There are two Basic Loop Types, Self-Reinforcing and Goal Seeking.
      - Self-Reinforcing loops lead to exponential growth that arises from positive feedback.
      - Goal Seeking or balancing loops counteract disturbances that move the system away from the goal. These negative loops seek equilibrium.
   b. Insights and Recommendations
      Examine the loops. Consider how the loops could be changed to improve behavior.

[30] Hines, Jim., Lecture Materials, System Dynamics II Course 15.876. Sloan School of Management Massachusetts Institute of Technology, Fall 2000.

4. Model Development

   Map the graphical description of the system into a mathematical description. Now the computer may simulate the problem. Model a hypothesis that is easy, central, or interesting in that order or importance.

   - Easy - Start with the hypothesis that is easiest to model or represents a real system that is familiar.
   - Central - Model a hypothesis that is structurally central. That is loops into which hook many other loops.
   - Interesting - Model the behavior pattern that you think is important in generating the dynamics.

5. Model Analysis

   a. Validate the Model

      Scrutinize the behavior of the model to insure that it properly captures known characteristics of the system.

   b. Simulate the model through the adjustment of variables in order to test policy alternatives.
   c. Record insights and conclusions throughout the modeling process.

References

Ambrose, M., et al; “Organizational Initiative Analysis, Sikorsky Aircraft Reengineering”. MIT Sloan School of Management Organizational Processes Course, March 1998.

Baum, Geoff, et al; "Introducing the New Value Creation Index ", Forbes ASAP, April 3, 2000.

Buckley, E., “Corporate Chairman Commentary Regarding Product Platform Team Responsibility”. Executive Management Council Meeting, Sikorsky Aircraft Corporation, March 1999.

Edvinsson, Leif, "Developing Intellectual Capital at Skandia", Long Range Planning, Vol. 30, June 1997.

Ford, David N., and Sterman, John D., "Dynamic modeling of product development processes", System Dynamic Review, Vol. 14, No. 1, Spring 1998.

Forrester, J.W., "Counterintuitive behavior of social systems", Technology Review, 73(3), 1971.

Goldratt, Eliyahu M., The Goal. New York: North River Press, 1992.

Gover, D., “Vice President of Production Engineering, New Engineering Organization Address to Engineering Staff”. Sikorsky Aircraft Corporation, February 1998.

Hayes, R.H., Wheelwright, S.C. and Clark, K.B., Dynamic Manufacturing. New York: The Free Press, 1988.

Hines, Jim., Lecture Materials, System Dynamics II Course 15.876. Sloan School of Management Massachusetts Institute of Technology, Fall 2000.

Hines, Jim, Molecules of Structure Version 1.4 Building Blocks for System Dynamics Models, LeapTec and Ventana Systems, 1996, 1997.

Keough, Mark, and Doman, Andrew, "The CEO as organization designer". The McKinsey Quarterly Number 2, 1992.

Lucas, William A., et al; "The Wrong Kind of Lean: Over-Commitment and Under-represented Skills on Technology Teams", The LeanTEC Project, Sloan School of Management Massachusetts Institute of Technology, May 2000.

Repenning, Nelson P., "A Dynamic Model of Resource Allocation in Multi-Project Research and Development Systems", Sloan School of Management Massachusetts Institute of Technology, Version 2.0, September 1999.

Repenning, Nelson P., "Resource Dependence in Product Development Improvement Efforts", Sloan School of Management Massachusetts Institute of Technology, Version 1.0, December 1999.

Repenning, Nelson P., "Why good processes sometimes produce bad results: A formal model of self-reinforcing dynamics in product development", Sloan School of Management Massachusetts Institute of Technology, Version 2.1, July 1999.

Roos, Goran, and Roos, Johan, "Measuring your Company's Intellectual Performance", Long Range Planning, Vol. 30, No. 3, 1997.

Slack, R., "The Application of Lean Principles to the Military Aerospace Product Development Process", Unpublished Master's Thesis. Cambridge, MA.: Massachusetts Institute of Technology, December 1998.

Sterman, John D., Business Dynamics, Systems Thinking and Modeling for a Complex World. New York: Irwin McGraw-Hill, 2000.

Stewart, Thomas A., "Your Company's Most Valuable Asset: Intellectual Capital", Fortune, October 3, 1994.

VENSIM System Dynamics Model

Womack, James P., and Jones, Daniel T., Lean Thinking: Banish Waste and Create Wealth in Your Corporation. New York: Simon & Shuster, 1996.