CIFE CENTER FOR INTEGRATED FACILITY ENGINEERING

Improving Project Performance by Predicting Engineering Impacts on Operations Failure Risks Using a Quantitative Model of Product, Organization, Process, and Environment

By

John Chachere

CIFE Working Paper #096
JUNE 2005

STANFORD UNIVERSITY

WE ARE GRATEFUL TO NASA AMES RESEARCH CENTER'S ENGINEERING FOR COMPLEX SYSTEMS PROGRAM FOR SUPPORTING THIS WORK UNDER GRANT NUMBER NNA04CK20A, AND FOR PROVIDING VALUABLE FEEDBACK.

If you would like to contact the authors, please write to:

c/o CIFE, Civil and Environmental Engineering Dept.,
Stanford University
Terman Engineering Center
Mail Code: 4020
Stanford, CA 94305-4020

Abstract

Producing manned space missions, new cancer drugs, and civil facilities all involve an expensive design and development effort that culminates in operations that are at risk of technical failure. Today, few existing social science and engineering theories offer sufficient precision to support specific decisions involving project tradeoffs among operations failure risks and programmatic considerations (such as development schedule and cost). The aim of this research is to precisely identify theory-based mechanisms by which management choices influence engineering processes—specifically the failure to complete necessary rework— that creates product flaws which jeopardize downstream operational success, and to quantify the degree to which these phenomena are likely to manifest for a given project. We propose a quantitative method for analyzing these risks by modeling the interdependencies among upstream engineering and downstream operations considerations. The proposed model integrates the Probabilistic Risk Analysis (PRA) model of product functions and operating environments, with the Virtual Design Team (VDT) simulation model of engineering organizations and processes. This research offers a formal definition of the ways in which many theoretical factors (such as component redundancy, human error, constructive oversight, and information processing) that are chosen in the design phase may subsequently interact to determine operational phase failure risk. The integrated model is intuitively more justified for interdependent project planning applications than either model alone because most failures involve interactions among product, organization, process, and environmental factors. The proposed model offers project planners a more holistic assessment of operational failure risks and a broader range of testable mitigation strategies than models that are limited to consider the operations stage alone.

Table of Contents

List of Tables

List of Figures

Chapter 1 - Introduction

1.1 PRACTICAL PROBLEM

Attempting to manage high-risk technologies while minimizing failures is an extraordinary challenge. By their nature, these complex technologies are intricate, with many interrelated parts. Standing alone, the components may be well understood and have failure modes that can be anticipated. Yet when these components are integrated into a larger system, unanticipated interactions can occur that lead to catastrophic outcomes. ***The risk of these complex systems is increased when they are produced and operated by complex organizations that also break down in unanticipated ways.*** *-NASA 2003*

Many projects, such as space missions and pharmaceutical development, involve a complex and interdependent design and development effort that culminates in operations that are at risk of failure. In many industries, these projects result in operational failure far more frequently than competent and careful human planners predict. Recent advances in risk analysis and project management indicate that to walk the surface of Mars, humanity need not completely master its organizations, processes, products, and environmental factors. Instead what we require is a way of understanding how the strengths and weaknesses in these four factors can complement or conflict with one another.

The web of interdependencies among early phase engineering activities and operations failures is complex, dynamic, and acts over a long period. Nevertheless, engineering flaws introduced during design and development frequently do result in catastrophic failure during operations. Our intuition is that in these projects, the probability of failure depends greatly but also somewhat predictably on the way that engineering activities are planned and managed early in the project. Stakeholders who

are serious about improving risk assessment and mitigation should consider their complex operations from a perspective that integrates more diverse upstream sources of error than traditionally have been considered.

The National Aeronautics and Space Administration (NASA) workforce lives by the creed "Failure is not an option" [Kranz 2000], and strives also to be "Faster, better, cheaper" [NASA 2003]. These noble goals frequently conflict, however, and each of NASA's public failings can be traced to honorable choices in service of one, but to the detriment of another. This research will provide a tool that can help organizations like NASA become faster, better, cheaper, reliable, sustainable, and more, without "operating too close to too many margins" [NASA 2003].

1.2 EXISTING APPROACHES

NASA has devoted tremendous resources to risk management and accident investigations, and they have consistently traced errors upstream from operations to roots in development and design [Bergner 2005]. Researchers estimate the fraction of major system failures that can be traced to human and organizational shortcomings to range from fifty to ninety percent [Paté-Cornell 1990, Murphy and Paté-Cornell 1996].

In response to these phenomena, social science and engineering researchers have developed rich theories of collaborative activities and their relationships to risk (Most notably Bem et al 1965, Perrow 1986, Roberts 1990; For a literature review see Ciaverelli 2003 and Cooke et al 2003). Unfortunately, because these theories lack a quantitative definition, it is difficult to rigorously evaluate their range of valid application, their potential interactions, and their relative importance when in conflict with other theories.

Modern project planners today find that few research findings provide enough precision to support analysis of common but difficult practical decisions involving specific project risks, costs, schedules, and other objectives. Today some organizations reap important benefits from quantitative programmatic and risk models developed by engineers (Most notably Paté-Cornell 1990, Paté-Cornell and Fischbeck 1993.1 and 1993.2, Murphy and Paté-Cornell 1996, Paté-Cornell et al 1996, and Dillon and Paté-Cornell 2001), but we believe that increasing these tools' level of integration can further improve decision making.

NASA has used Probabilistic Risk Analysis (PRA) to quantitatively estimate the failure probabilities of complex engineered systems [Paté-Cornell and Fischbeck 1993.1, 1993.2], and is continuing to apply the technology on the International Space Station. Because the PRA method does not provide a model of the project's upstream engineering organizations or processes, it cannot estimate the influence these factors will have on risk.

NASA has also used the Virtual Design Team (VDT) simulation to quantitatively predict the behaviors of complex engineering activities, including many that are associated with risk [ePM 2003; see also Kunz et al., 1998; Levitt et al 1999]. Unfortunately, this model does not include products or their operating environments, and is unable to assess the impacts these behaviors will have on the probability of failure in operations.

1.3 RESEARCH QUESTIONS

We pose two questions:

1. *What are some mechanisms by which engineering design activities create product flaws that later increase the probability of downstream operational failure?*
2. *What is a method that quantifies the degree to which specific engineering (design and development) phase choices change the operational phase failure probability of a given project?*

1.4 PROPOSED MODEL

Figure 1 illustrates our proposal to quantitatively represent and interrelate in a theory-founded manner both upstream engineering organization- and process-contingencies and possible downstream product- and environment-related factors. The proposed solution integrates a PRA model of product functions with a VDT simulation of engineering organization and process. The result is an estimate of operations failure probability that considers flaws that are introduced during the early design and development stages. The proposal offers a structured method for the formal evaluation of the risk effects of many controllable design and management choices.

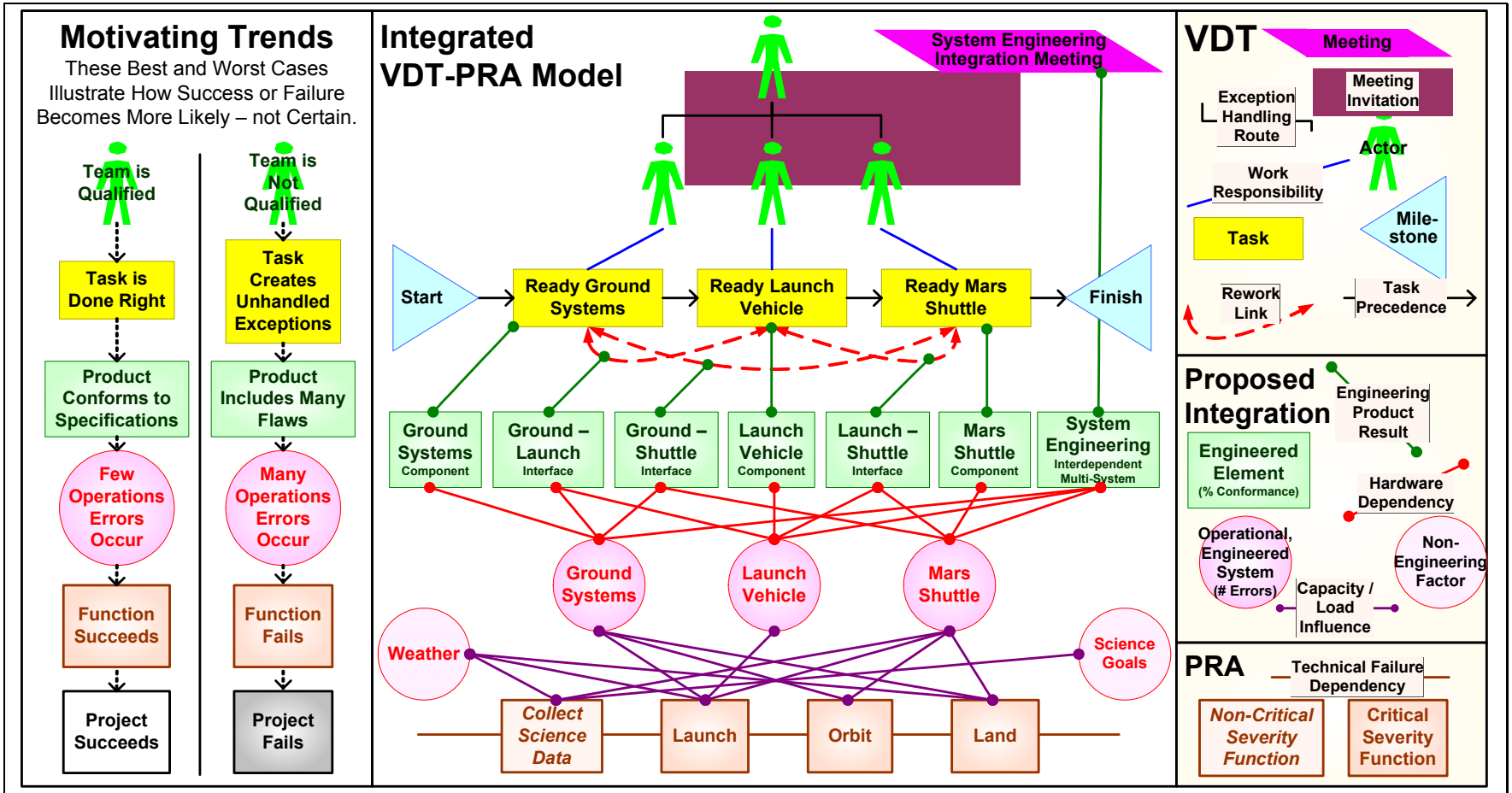


Figure 1 Proposed synthesis of VDT and PRA models.
This schematic illustrates how we propose to link outcomes of VDT's model of engineering organization and product with PRA's estimates of product failure risk. In the example, one manager oversees three teams with distinct and interdependent tasks. If the organization cannot adequately meet the process requirements, our integration predicts that the reliability of corresponding engineered elements will tend to suffer. During operations, flawed elements, in turn, can create errors that reduce the capacity of engineered systems, making functional and project failures more likely. We propose using VDT to evalutate specific engineering project conditions and using PRA to determine their impacts on project failure probability for a specific product. Figure 6 illustrates the required analytic steps, Figure 7 shows the relationships among data, and Table 4 provides calculations for a specific example.

1.5 CONTRIBUTION

This research extends project planning and risk management research traditions. Its contribution to engineering risk analysis is that it formally defines theory-founded methods to relate risks in the operational phase product with design phase choices of organization and process design and the design phase environment. More specifically, we use VDT to show how the match between organization and process influences a new *rework deficit* measure, and we use PRA and related methods to show the ways in which a high rework deficit can lead to errors and failure in operations. This paper provides several illustrative examples, and discusses methods for further justifying the contribution's power and generality.

1.6 IMPLICATIONS

It is unlikely that we will ever develop perfect knowledge and total control of our complex and interdependent organizations, processes, products, and environmental factors. Taking an integrative view on recent advances in risk analysis and project management suggests that we can use these four factors to complement or constructively conflict with one another, building on strengths and diminishing weaknesses, even when our information and control are limited and uncertain.

Because the specific synthesis we propose preserves the core, theory-founded PRA and VDT models, it provides a formal definition of the ways in which many theoretical factors—such as component redundancy, human error, constructive oversight, and information processing—interact to determine and decrease technical failure risk. We hope that this contribution will lend precision to the communications among traditionally engineering and social science disciplines, and that this can improve the rates of constructive consolidation and agreement in the field.

If successfully implemented in practice, project managers will be able to use our proposed method to coordinate the mission product, organization support and processes of the operations phase by considering the risk elements in early design and development. Planning decisions that appear to fit easily in our integrated model include product component and subsystem redundancy and configuration; organizational participants

skills and structure; processes of design and development; and engineering collaboration mechanisms and authority distribution (centralization). We believe that the proposed model's broad view will provide a more realistic assessment of operational failure risks than models that are limited to consider operations or engineering alone, and that the method will make a broad range of mitigation strategies analytically tractable. With a united model of the engineered system, engineers will be better equipped to make decisions and allocate resources across the complex system consistently, and in alignment with management objectives.

1.7 ORGANIZATION OF THIS PAPER

The remainder of this paper is organized as follows:

Chapter 2- Existing Theory and Practice summarizes the current state of theory and practice in technical failure risk analysis and engineering program risk management, and then describes two important works that, like our proposed model, attempt to balance the two considerations.

Chapter 3- Conceptual Model provides qualitative definitions, intuition and reasoning that give our proposed method face validity.

Chapter 4- Analytical Model provides a mathematically formal, quantitative algorithm for calculating a project's technical failure risks using a variety of methods.

Chapter 5- Discussion offers a detailed description of the research contribution, a procedure for its ongoing justification, predicted impacts on theory and practice, and some promising next steps.

Chapter 6- Conclusion reviews the methods and contributions of the research presented in this paper.

Chapter 2 - Existing Theory and Practice

In this section, we summarize the research and field applications that our contribution builds upon. We also assemble a language for describing projects with the consistency that our contribution's analytical presentation requires.

A *project* is a collaborative endeavor undertaken in a limited timeframe by one or more organizations to create a specified product, using a particular process, and within an environmental context of uncontrolled factors. This research targets projects that have a clear *project objective*— a goal of benefits such as a moon landing, or a new, approved drug— while subject to *project constraints*— limits on the schedule and resources that can be consumed to achieve the objective. *Project planning* is the activity of evaluating and selecting among the available alternative organizations, products, processes, and operating environments.

In planning a project, and in directing one during execution, managers continually commit time and resources to the project in order to balance two considerations. The first is *project failure risk*— the probability that in the end, the product will fail to meet the project objective. We explain that when resources allow it, the PRA method is ideal for analyzing project failure risk in the first section of this chapter.

We call the second consideration *program risk*. Program risk is the chance of exceeding project constraints such as available schedule or resources. We explain that VDT operationalizes a range of organization and process theories that address program risks in this chapter's second section.

Many (if not most) project planning decisions impact both program risk and project failure risk, so decision makers must understand the impacts their interdependent decisions have upon both constraints and objectives. We introduce two well-formulated

integrative models that assess the available alternatives and that recommend a course of action in this chapter's third section.

2.1 PRA MODEL OF PROJECT FAILURE RISK

Motivation

The volume of research and industry attention that product failure risk garners is appropriate because it is both important and complex. Project risk is important because for many stakeholders, success provides the project's only benefits. It is also difficult to analyze because failures typically have many interdependent and obscure causes. Some researchers suggest that the difficulty of risk management does, and should, limit the complexity and interdependence of viable human activity [Perrow 1984, Perrow 1994].

Method

Risk analysis is the identification, assessment, and mitigation of important events that cannot be predicted with certainty [Paté-Cornell 2004]. Risk analysis is particularly valuable in the scrutiny of complex products that contain many interdependent components (such as space craft and nuclear power plants), because their aggregate dependability is difficult to assess.

Some complex systems are *single string*—they require reliability in all of their subsystems, while others are *redundant*—they can withstand certain individual component failures without causing a total system failure. When a system includes a single-string component with a high failure risk, it is common to recommend redesigning the component with redundancy. However, risk analysts know that functions' probabilistic independence determines the effectiveness of redundancy as a risk reduction strategy.

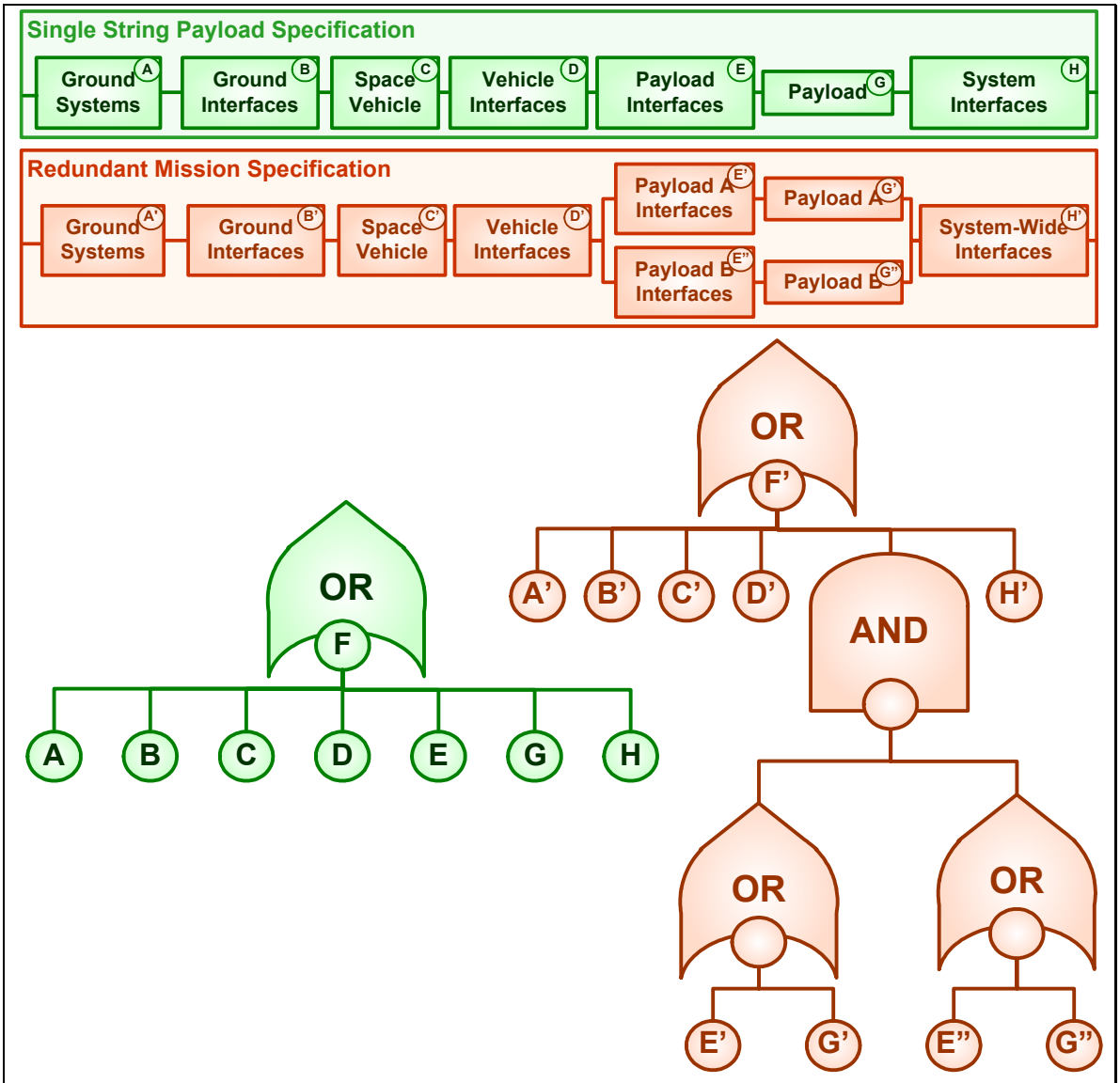


Figure 2: Functional block diagrams and corresponding fault trees.
We can derive a functional block diagram from process definitions starting with one essential function per design task. For example, a project with three tasks includes three sources of possible unidisciplinary risk. We will transform VDT's measures of information and rework dependencies into instances of interface failure risk. Finally, VDT measures of project meeting attendance provide information about subtle, multiple-system interaction risks.

Probabilistic Risk Analysis (PRA) forecasts complex systems' failure probabilities by characterizing each potential contributor and defining their interactions. PRA calculates a product design's reliability by decomposing it conceptually into functional blocks, assessing component and subsystem reliabilities when subjected to external events, and aggregating to a total failure probability. Figure 3 illustrates the principle using two

common PRA tools, functional block diagrams and fault trees, and explains the underlying logic—that aggregating constituent elements' failure probabilities up to a total product failure probability allows us to relate subtle design project behavior to product outcomes.

Limitations

In many projects the upstream design organization and process is a common failure source that influences all engineered elements, and that is therefore of the utmost importance in estimating failure risk. Regardless of their physical relationships, the manifestations of engineering errors during operations are probabilistically dependent on one another because the upstream processes interact in complex ways (Pooled or stronger interdependence, see Thompson 1967).

The System-Actions-Management (SAM) Framework

Understanding the risks of failure that engineered components present requires a deep understanding of engineering project performance that PRA leaves to domain experts under a guided conversation (given that few relevant statistics are available). The tools' predictive accuracy therefore remains limited by experts' ability to assess human and organizational risks, at the same time that the competence of experts is a common source of criticism in PRA applications [Paté-Cornell 2004].

Fortunately, the recent development of the System-Actions-Management model (SAM) [Murphy and Paté-Cornell 1996] offers a clear framework for integrating PRA with more advanced probabilistic models of human and organizational behavior. The method shows how we can extend our analysis of engineered systems to the actions that impact it, and in turn, to the management decisions that lead to those actions. The original formulation provides several examples of action models, including rational, boundedly rational, rule-based, and execution (under limited effectiveness).

The model that we propose uses the SAM extensions to extend a base PRA model of an engineered system in operations. Specifically, we tie the overall formulation to the SAM structure, and use two of the action models as guidelines for interpreting a sophisticated, theory-based model of engineering behavior and program risks.

2.2 VDT MODEL OF PROGRAM RISKS

Motivation

> *Flawed practices embedded in NASA's organizational system continued for 20 years and made substantial contributions to both accidents ... For all its cutting-edge technologies, "diving-catch" rescues, and imaginative plans for the technology and the future of space exploration, NASA has shown very little understanding of the inner workings of its own organization…*
>
> *Although schedule deadlines are an important management tool, those deadlines must be regularly evaluated to ensure that any additional risk incurred to meet the schedule is recognized, understood, and acceptable. -NASA 2003*

In many industries such as aerospace and construction, complex and unique projects routinely overrun schedule and cost budgets, despite careful advance planning and mid-stream adjustments by experienced managers. Unfortunately, it is extremely difficult to correctly assess the extent of program and project risks, and to determine the implications of possible mitigation strategies. Well-meaning project managers frequently make mistakes that only compound the existing problems. For example, the common practice of adding software engineers to a project in response to schedule slippage can do more harm than good [Brooks 1975].

Galbraith (1973) indicates that organizations like these behave as if their primary function is to route and process information. Shortcomings in information flow or knowledge in an organization produce *exceptions*— events that require information to be referred to management for decision making. Exceptions occur during work with a frequency based on task complexity, as well as on the adequacy of the assigned actor's experience and skills. *Exception handling* is the manner in which organizations respond by routing information or queries to complementary resources such as management or technical experts. *Hidden work* is the coordination and exception handling efforts that can represent a substantial fraction of the total labor and schedule pressures in complex and interdependent projects. One reason why project managers underestimate the emergent workloads of subordinates whose work is highly interdependent is that hidden work is hard to predict and not explicit in traditional planning theories and schedule tracking systems.

When a supervisor oversees many actors, each of whom has complex tasks that are being performed in parallel, the exception handling workload sometimes becomes

unmanageable. The resulting backlog can cause a failure to respond to coordination attempts, creating a ripple effect of problems extending through all of the interdependent activities. Overloaded workers who fail to respond to communications also compound the information supply problem and compromise others' performance. Projects that involve complex and interdependent tasks impose additional direct and communication requirements, and tend to create more unhandled exceptions.

A second factor that critically impacts the model behavior is the amount of time between a request for action or information. This is known as *response latency*, and this metric is both a cause and consequence of diverse and critically important success factors [Chachere et al 2004.1, Chachere et al 2004.2, Chachere et al 2004.3]. When projects fall far behind schedule due to managerial or technical bottlenecks, latency reaches a point at which rework decisions are no longer made. Under these circumstances, rework requirements are more frequently ignored, often leading to a rapid degradation of process quality [Jin and Levitt 1996].

Under these conditions, project performance can falter and can degrade rapidly in a manner that is analogous to the emergence of turbulence in fluid flows [Fayol]. When projects fall behind, well-founded decision making and corner cutting alike frequently push risks from the program into the product [Garber 2005]. Predicting the conditions under which project performance enters this stage is a challenging research question that has practical importance, as illustrated in Table 3.2.1.

[NASA Administrator] Goldin was also instrumental in gaining acceptance of the "faster, better, cheaper" approach … and downsizing … He rejected the criticism that he was sacrificing safety in the name of efficiency… ***"When I ask for the budget to be cut, I'm told it's going to impact safety on the Space Shuttle … I think that's a bunch of crap."*** *-NASA 2003*	*"A decade of downsizing and budget tightening has left NASA … with a less experienced staff and older equipment." … The Program was operating too close to too many margins…* ***NASA has recently recognized that providing an adequately sized and appropriately trained workforce is critical*** *to the agency's future success. -NASA 2003*

Table 2.2.1 The Fine Line Between Pushing and Breaking the Envelope

The memory of leaders who dramatically cut budgets depends on their projects' subsequent successes or failures. Not cutting enough can cause program failure, which risks termination by the financiers, and can sacrifice the organization's other projects' prospects. However, cutting too much can result in a project failure, in which all benefits are lost and sometimes a lawsuit can bankrupt the organization. Integrated project planning methods, including APRAM, VDT-GA, Pugnetti 1997, and the method we propose, can help decision makers to identify which aspects of a project should be cut and which should be preserved—or even enhanced.

Method

Computational Organizational Modeling

Computational organizational modeling quantitatively operationalizes established organizational theories, some of which are relevant to the study of risk. Its practical appeal is that virtual testing of project plans and interventions can provide valuable insights before committing project resources. Although schedule tracking systems such as Primavera and Microsoft Project are frequently consulted as quantitative project models, they depend on users to forecast the interactions among interdependent tasks and teams. In contrast, the Virtual Design Team simulation system (VDT) was created in part to address project managers' difficulty in predicting emergent project behavior.

By grounding a computational model explicitly in a theoretical framework, researchers can explore complex ramifications of a theory (or set of theories) that extend qualitatively beyond the reach of human intuition. Although some of the earliest and most influential work in organizational science was developed in concert with formal models [Cyert et al 1959, March and Olsen, March and Cohen], the method has never become a standard in the discipline. In recent years, however, the computational modeling of organizations has enjoyed a popular resurgence among researchers seeking to better understand new and established theories [March 2001 and Burton 2001].

VDT

We propose a method of understanding engineering project participants' behavior that employs the Virtual Design Team (VDT) model, which is based on several of the most established theories of organizations (notably Galbraith 1977 and Thompson 1967). For a review of VDT's theoretical basis see Christiansen 1994, and for a technical explanation of its internal mechanics see also [Jin and Levitt 1996].

Whereas most project planners forecast the behavior of engineering efforts based largely on personal experience, those who employ VDT focus on accurately assessing engineering project parameters such as tasks' complexities and workers' skills. Based on this information, the VDT model simulates the engineering work and coordination that established organizational theories predict would emerge in practice.

By testing a project plan in the simulator, planners can predict participants' backlog, coordination effectiveness, schedule risk, labor costs, and other objectives [Kunz et al 1998; Jin et al 1995, Levitt et al 1999]. By comparing the predictions from alternative cases, VDT users can assess which proposed organizations, processes, and culture are most likely to meet their project's goals. Our method extends this capability by linking VDT outcomes to a model of an engineered product whose reliability impacts the probability of project failure in a later operations stage.

Input

VDT analysis begins by identifying the actors of organization hierarchy, and the tasks within a precedence network. A second, iteration specifies details cultural measures such as centralization, formalization, and matrix strength (project versus functional orientation); Organizational factors such as actors' varying levels of skill and experience; and process definitions including tasks with varying levels of procedural uncertainty, complexity, and required skills. The VDT model also defines relationships among these basic elements, including authority hierarchies that interrelate actors; primary and secondary task assignments indicating which actors address which tasks; and links interrelating tasks that have rework and information exchange dependencies.

Processing

VDT probabilistically simulates the complex and subtle implications of project plans using a discrete event simulation that emulates actors processing work, attending to communications, handling exceptions, making decisions about rework, attending meetings. Of equal importance, the method simulates the distribution of attention among these activities when more than one action is pending. Simulated actors process information at a rate determined by task complexity and actor skill and experience. VDT models exception handling as involving an upward flow of exception handling requests

and a downward flow of rework, quick fix, or no action choices along a fixed exception handling hierarchy. When necessary rework is not performed, VDT predicts that engineering performance suffers, and our proposed model predicts that flaws in the engineered product are more likely.

Figure 3 Typical Virtual Design Team (VDT) Project Model.
VDT models design project participants' individual characteristics, organizational exception handling structure, scheduled meetings, task characteristics and precedence, information exchange requirements, and rework dependencies. VDT compares the process's information processing load with the organization's information processing capacity. It produces detailed estimates of a project's emergent cost, schedule, quality, and other measures. We combine VDT's strengths with those of PRA, creating a model that is more valuable in some cases than the sum of its parts.

Output

VDT offers a range of performance predictions, including emergent work volumes, a project schedule, and coordination rates. VDT estimates overall design quality using coordination time such as information exchange and meetings, and decision waiting time. We can view these metrics at an aggregate project level, or drill down to understand individual actor or task predictions in detail.

Limitations

The goal of planning projects with a comparable methodology and accuracy as is demonstrated in planning today's bridges [Levitt and Kunz] is exciting, but many years away. The VDT model in particular has shown some remarkable successes in predicting the emergence of engineering phenomena that lead to operations failures [For an aerospace example see Levitt et al 1999], but it requires a significant calibration effort to be accurate in a predictive sense.

In addition to calibration, the VDT system includes some important theoretical gaps in emulating projects. For example, it does not model uncertainty in most input quantities, even though many experts are uncertain about the details of team or task composition (Chachere 2004.1 proposes a solution to this problem). The system also does not simulate actor decisions to change the organization or process during the project, even though this behavior is common in large projects.

As its name suggests, the Virtual Design Team Simulator was originally built to model routine design tasks. Since its inception in 1988 [Cohen], researchers have developed enhancements to apply the tool to address operations [] and service processes [], fast-tracking [Salazar], learning [Oralkan], and trust in distributed teams [Zolin]. These enhancements intuitively match specific applications more closely than the core model, but most require additional justification steps.

As a practical tool, VDT also has room to grow. It has no explicit product model, and does not directly address the impacts that engineering processes have on a product. VDT makes no effort to automatically compare outcomes to determine which is most preferred. The later chapters propose a model that uses PRA to may help to strengthen VDT in these areas.

2.3 INTEGRATED MODELS OF PROJECT AND PROGRAM RISK

In this section, I compare the proposed method and two existing project planning methods that consider both project failure risk and program risk.

APRAM

The APRAM method [Dillon and Paté-Cornell 2001, Dillon et al 2003] is quite similar to the one proposed here, in that it addresses program and project risks simultaneously. It achieves this by defining a budget that is divided between expenditures on the configuration and reinforcement of the engineered system, and a budgetary reserve that enables expenditures to counter uncertain programmatic events.

APRAM uses decision trees to model the set of uncertainties and decisions that determine program risks. Although this method is quite broadly applicable and is theoretically able to accommodate an organization theory-based model such as VDT within the event trees, it offers no specific guidelines on how to do so. Section 2.2 describes several benefits that the proposed method derives from linking to VDT.

In the APRAM model, uncertain programmatic costs limit the budget that is appropriate to spend on reinforcing the operational system. However, APRAM does not directly model the direct impact on project failure risk that engineering activities such as rework can have. In contrast, the model we propose in this paper focuses on this relationship.

An important strength of APRAM is that in may cases it is prescriptive, meaning that is able to mathematically select the best among several choices using decision trees and a Langrangian method. In contrast, the proposed method is merely predictive, meaning that it estimates the impacts of discrete alternatives that decision makers must evaluate individually. We may be able to extend the proposed method to optimize a broad range of important factors, such as management priorities, hardware reinforcement, and leadership, as long as they do not influence the predictions of the VDT simulation. Section 11.5 explains this and other optimization methods, and Chachere 2004.2 provides one formulation of a VDT-APRAM synthesis.

It may be possible to enhance the proposed model by formulating the utility function similarly to APRAM using variables that represent operational function configuration, for example, or alternative base capacities that reflect choices of hardware materials.

Pugnetti 1997

In his 1997 doctoral dissertation, Carlo Pugnetti also approaches the problem of simultaneously estimating program and project failure risks using PRA and VDT. His approach is primarily to use a pure mathematical formulation based on discrete time queuing, and to use VDT sparingly to verify the organizational viability of optimized process and product configurations. The method proposed in this paper satisfies the most critical of Pugnetti's criteria for further enhancing the VDT-PRA integration (pp. 141-144).

For the most part, we agree with Pugnetti's idea that unfixed engineering problems are an important part of the operations risk management problem. Pugnetti [1997] p.71 uses "undetected exceptions" to measure the risk associated with a simulated engineering project:

$$E(U_i(t)) = K * S_i * T_i^t$$

Two minor problems with Pugnetti's formulation are that the term exception is more generally used to refer to a procedural event, rather than an engineering fact (that we will term *defect*), and more importantly, that this formula always produces $E(U_i(t)) = 0$. The correct formulation would be

$$E(U_i(t)) = K* T_i^t * S_i$$

More significantly, Pugnetti assumes that all detected problems are fixed, and that a reworked element does not contribute to failure probability. Field evidence including NASA 2003 appears to contradict these assumptions.

Finally, in Equation 4.5 (p. 49) Pugnetti assumes that exceptions contribute to error probability in a linear fashion thus:

$$\mathbf{F} = \mathbf{F}_0 + \mathbf{N} * \mathbf{E}$$

or, in a linear fashion, thus:

$$\mathbf{F} = \mathbf{F}_0 + \mathbf{N}^{\gamma} * \mathbf{E}$$

When there are many errors, these equations can produce failure probabilities greater than one. Nevertheless, we sustain the spirit of the latter equation by employing a geometric model of engineered subsystems' anomalies' impacts on functions during operations.

Another interesting result in the thesis is Pugnetti's establishment of criteria that define the convergence of project duration. When conditions are at their worst, more exceptions may be created than are fixed, and so the project may go on forever. By providing convergence criteria that define the conditions under which this occurs, Pugnetti sheds light on the very important "turbulence" problem we described above. In contrast with Pugnetti's provision of crisp criteria, VDT merely balks when asked to run a simulation that produces task durations (or other intermediate parameters) that exceed certain feasibility limits. The method we propose cannot offer convergence criteria because it relies on VDT to estimate the behavior of engineering projects. In practice, we do not expect this to be an important differentiator between Pugnetti's work and the proposed method, because managers generally prevent these degenerate behaviors by intervening during the course of real projects.

Pugnetti's "undetected exception" measure is closely related to the VDT-based "rework deficit" concept that we claim influences the likelihoods of product defects, subsystem anomalies, and project failures. Pugnetti uses the results of engineering processes somewhat directly to estimate the probability of failure in operations functions. In contrast, the model we propose follows a sequence of inferences from engineering rework deficits, to product defects, to engineered subsystems' anomalies, to the loss of capacity, to functional failure, and finally to failure risks of various severities. This additional detail requires more data gathering, but in return, it offers more calibration opportunities and enables the analysis of a wider range of intervention strategies.

Chapter 3 - Conceptual Model

When a program agrees to spend less money or accelerate a schedule beyond what the engineers and program managers think is reasonable, a small amount of overall risk is added. These little pieces of risk add up until managers are no longer aware of the total program risk, and are, in fact, gambling. Little by little, NASA was accepting more and more risk in order to stay on schedule. -NASA 2003

3.1 OVERVIEW

Theoretical Conceptualization

We propose to predict operations phase risks that result from shortcomings in the engineering stages by using an integrated model that follows three intuitive steps. The first of these is embodied in VDT, the second is a novel contribution, and the third is fundamental to PRA. More specifically, the VDT model compares an organizational hierarchy's information processing capacity against a task network's information processing load to predict emergent engineering behaviors. Our proposed model extends this with the notion that this conduct influences the target product's conformance to specification, and therefore, the probability that its elements will meet project objectives. Finally, we use a PRA model of the product to predict the combined significance that these impacts on engineered elements will have on project failure risk.

To lend specificity to this claim, our model operationalizes the following observed and theory-founded trends. Competent engineering organizations (those with adequate information processing capacity) are often able to recognize the need for and dedicate the time that is necessary to handle most emergent exceptions, by performing rework if necessary, and to fully staff coordination meetings. The additional time and attention spent on engineering processes may increase the number of engineering flaws that are discovered and rectified, thus improving the probability that the resulting product will

conform to the specification. The more engineering tasks are conducted in this way, the greater the improvements to conformance and to the probability that the final product will meet the project objectives during operations.

In contrast, an inadequate design organization will have more mishandled and ignored exceptions, and lower attendance at meetings. Each of these phenomena represents an opportunity to permit engineering flaws into the product, which probabilistically leads to a less conforming and more error-prone result.

Formulation Method

Figure 1 on page 6 shows how we operationalize the claim that engineering exception handling behaviors influence product failure risk. We use a quantitative model that synthesizes VDT's engineering behavior model with PRA's model of the influence that engineered elements have on product failure probability. In the remainder of this chapter, we present the theoretical conceptualization for each of our method's four parts:

1 **Frame and Foundation** We must first define the project objectives, from which we derive definitions of success, failure, and project failure risk. We can then identify at a high level the project structure, as well as the products, organizations, processes, and environmental factors that can influence the probability of meeting the project objectives.

2 **VDT Model of Organization and Process** Using VDT, we can identify and model the organization and process that designs and develops the specification elements into operational elements upon which project success relies. Then we can use VDT to predict the joint probability distributions of exception handling and meeting attendance to use in the estimation of failure risk. Finally, we can apply the proposed method to estimate the degree to which each element conforms to specifications in design and development.

3 **PRA Model of Product and Environment** Using PRA, we can break down the operations' total failure probability into functional components whose failure probabilities can be assessed independently. This process should continue to the level of individual components, interfaces, and interdependent multi-systems.

Using statistics, expert assessments, or other methods appropriate to PRA, we can estimate the failure probabilities of functions that do not involve designed or developed elements.

4 **Model Synthesis** In software engineering, the term "*integration points"* refers to data types and structures that are linked directly from one software model to another software model. In this step, we can calculate the rate of failure for each element based on engineering conformance estimates from VDT. Finally, we can use the PRA methods and functional failure probabilities to calculate the probability of project failure.

Chapter 4 presents a mathematical description of the problem we address, and formally derives our proposed solution.

3.2 FRAME AND FOUNDATION

In this step, we define the project objectives, from which we derive definitions of success, failure, and project failure risk. We identify at a high level the project structure, as well as the products, organizations, processes, and environmental factors that influence the probability of meeting the project objectives. Practitioners must assess the high-level project structure and each of these factors early in their planning process because the relationships among them determine the points at which VDT and PRA models should be integrated.

POPE factors

The method we propose is an example of a *POPE model*—a formal description of the planned engineering Process and Organization, the Product to be created, and the uncontrolled factors in the Environment where the product will operate. Figure 5 provides definitions of these four *project factors*.

3.3 PROJECT STRUCTURE

In the application areas we have targeted, (including construction, aerospace, consumer products, and software development) large projects almost universally employ a *stage-gate* structure. Stage-gate projects are large efforts that are composed of a sequence of smaller segments of qualitatively different activity called *stages*. Between each pair of

stages is a *gate* in which major decisions or events occur. Table 1 describes each of four typical stages that we develop further in our example application.

Stages

We intend our model to address only those project activities that occur after the approval of a *specification*—a (usually written) declarative statement of subgoals that are believed necessary and sufficient to achieve the project objective. In most applications, specifications are only approved to become projects after a thorough review process that we presume ensures a negligible probability of fundamental error.

In a typical project, such as the one we illustrate in this paper, *design stage* activities elaborate and instantiate this specification into a blueprint for an organization and process that can achieve those goals. *Development stage* activities translate the design blueprint into one or more physical artifacts that will enter operations.

In VDT, simulated designers assess the degree of conformance between a design and specification explicitly after completing each subtask (portion of a design task). When the verification is successful, a design actor is free to enter the design into the final product and attend to the next work item. When they feel that the conformance is questionable, they raise exceptions, or work items that require decision making. A decision-maker may choose to rework the subtask, which increases the designer's work volume but evidences attention to the conformance of a subtask.

Table 3.3.1 Example Project Stages

Complex projects often consist of sequences of smaller segments that aim to achieve a progression of subgoals. Our method defines these project stages in order to show how flaws from early stage behaviors develop into operations errors and project failure.

Stage	Description
Specification	Assumed, initial declaration of project goals. We model this stage implicitly by assuming it to be fully validated and approved.
Design	VDT model—stage one. In it, the team translates project goals into specific recommendations for the organizations, processes, and products that will be employed in development and operations.
Development	VDT model—stage two. Based on the design (mostly information) product, a physical spacecraft, ground systems, and set of formal procedures are created, tested and packaged for operations.
Testing	VDT model—stage three. Engineered elements that were designed and developed in earlier stages undergo an evaluation of conformance by an external team. Problems that surface in testing often lead to partial redesign or redevelopment.
Operations	PRA model. The space mission is executed during this stage, and each of its functions may be called upon to ensure success. Some of these functions are designed, and the robustness of the design and development will determine whether they fail, and possibly cause a mission failure.

Under ideal circumstances, each stage of engineering proceeds smoothly and creates a product that conforms to the requirements handed down from the previous stage. Shortcomings in the match between process and organization generally result in exceptions being generated, however, and in some cases decision-makers may choose to ignore verification failures when an error has been committed.

The VDT simulator predicts the "degree of conformance to requirements"; we claim that it can enable us to estimate the impact on failure probability that results from project stage behaviors occurring upstream from operations. More specifically, we assert first that the degree of conformance between product component designs and their requirements influences function success probabilities. By adopting the VDT model of information processing, we further assert that the design-phase disposition (designer and manager choices) and context that enable thoughtful design decision-making tends to improve design conformity.

Each stage culminates in the delivery of a *product*—all the work that participants judge to be of possible value during later stages, but that does not include intermediate results that are of little anticipated value.

Gates

Typically, each pair of adjacent stages is divided by a *gate*—a step in which management first translates the previous stage's product (compiled results) into an estimation of the future project's prospects, and then makes a "go/nogo" decision on whether to proceed with later stages. When a gate is passed, the content of a previous stage's product also influences the makeup of the next project stage (along with other factors such as earmarked resources and prevailing "standards of practice"). For example, a spacecraft design that selects nuclear power will require a different development team than one that relies on conventional fuel.

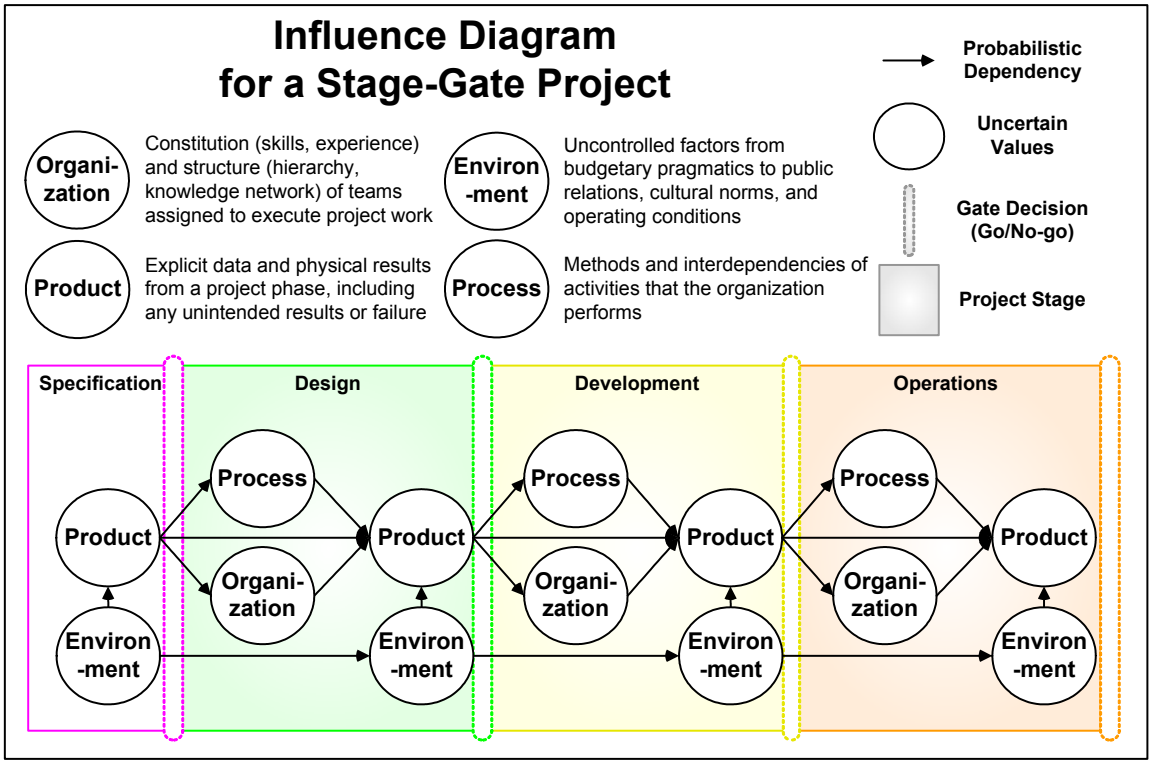


Figure 4 Influence Diagram for a Stage-Gate Project
The stage-gate structure helps decision makers to analyze decisions in the same way that many operations research algorithms employ "pinch points". Organizations prepare for gate decisions by distilling a wealth of diverse early stage phenomena into an assessment of their far simpler influence over a standardized product that passes into the subsequent stages. Their next step is to calculate the impact that important differences in these materials can have over the next stage. Our analysis employs a similar method through multiple stages to model a project of arbitrary length, and finally estimate the distribution of operations stage outcomes. Even late in the project, our estimate of early stage performance can provide information about the operational product that decision makers often lose at the pinch points.

We believe that gates enable managers to understand and to control large and unique projects' complexity by decomposing, routinizing, and distributing the decisions to specialized organizations and processes. For example, decision makers often claim to base gate decisions on standardized criteria, such as confidence in the ability to meet a company-wide target for return on investment. It is far simpler to base a gate decision on current product alone than it is to consider earlier phases' organizations, processes, and emergent engineering behavior, so many organizations manage program risks and project failure risk with different authorities and do not explicitly consider the relationship between them during gate decisions.

Conformance and Failure Risk

In most cases, a very complex combination of factors involving every project phase determines the failure or success of an engineered element. We say that performance is *conformance-dependent* when it is affected by the degree to which objectives specified on completion of one phase are met through the next phase. Failing to meet design and development objectives can alter operations behavior, so we say that operations behavior is both design-conformance-dependent and development-conformance-dependent.

We term *engineering-dependent failure risk* the chance that behavior that is dependent on the conformance of engineering stages causing project failure, and we model it with probabilities derived from an analysis of organizational and procedural conditions. We use the phrase *design-dependent failure risk* to describe the likelihood that design choices are fundamentally invalid or inconsistent with the target specification. Understanding design-dependent failure risk is important because failing to do so may lead to an improper decision over whether to proceed with a project, or to a project that is needlessly costly, risky, or extended in schedule. We can define *development-dependent failure risk* similarly, where the physical product of a development phase is the primary focus.

Value of Programmatic Information to Failure Risk Analysis

Although the stage-gate project form keeps complexity under control, it does this by hiding information behind standardized deliverables. In theory, *uncertainty absorption* is the loss of information that occurs when subordinates report their condensed findings to supervisors, and while it can help managers to make decisions by controlling complexity, it can also hide important and political information [March and Simon 1958, Simon 1977]. In particular, standardizing deliverables and decision-making criteria before gates can occlude important and available information about the likelihood of flaws in the product that engineering activities produce. When information about the engineering behavior is erased, we make decisions about whether to proceed with a mission more tractable. However, we also lose important information about the likelihood of hidden flaws in the product. Consequently, well-intentioned managers looking at operations failures have difficulty detecting the root causes of many engineering failures, because

records of the conditions that created those flaws are often discarded as relevant only to program risk analysis.

We believe that including a VDT-PRA analysis with the product can provide some of this information and improve decision-making. Information gathered about the product indirectly through knowledge of preceding organizations and processes can add value to the gate decision [Howard]. Most projects do not track engineering activities at the level of detail that VDT reports- for example, VDT predicts the backlog that can lead to corner-cutting, while many teams have no way to track this measure. For example, if the VDT analysis performed before one Lockheed mission [Jin and Levitt] had been institutionalized, it might have triggered a more focused review of the project plan and prevented the mission's operational failure.

3.4 MODEL SYNTHESIS

The synthesis we propose translates information over time, from engineering stages to operations, and relates information between factors, from organization and process to product. VDT predicts engineering stage *activities*—behaviors that result from a particular organization executing a specified process—that emerge from an organization and process plan. PRA predicts the operations behaviors that result from an engineered product within an environment context.

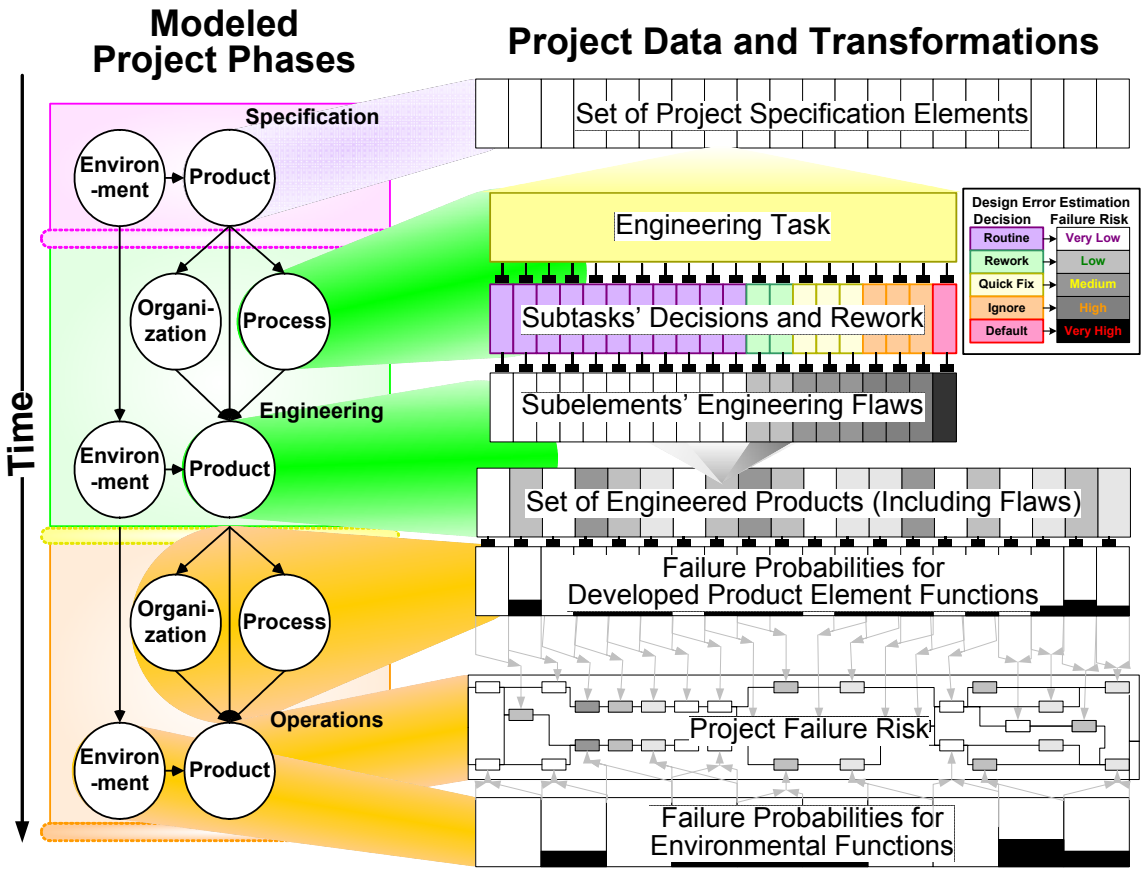


Figure 5 Relationship Between POPE Stage-Gate Model and Integrated VDT-PRA Method
This schematic illustrates the formal reasoning behind our proposed method. An initial specification that we presume correct becomes the target of engineering stage activities. These activities introduce possible flaws into the product wherever VDT predicts the organization may compromise its process. If they are exposed to adverse environmental conditions (as PRA predicts), these flaws can become product flaws and cause failure during operations. The calculations synthesize previously established VDT and PRA methods using a set of integration points. Figure 1 illustrates the involved objects, Figure 7 shows the relationships among data, and Table 4 provides the calculations for a specific example.

The PRA method allows modelers to adapt their formulation to accommodate the available data sources in relating complex phenomena to underlying or constituent phenomena. We can use standard PRA formulation techniques to assess the importance of, and relationships among, variables that influence project failure risk but that engineering activities do not influence. We also can relate total failure probability to the probability of failure for three types of engineered elements, *components*, *interfaces*, and *interdependent multi-systems*, which we define below. We can calculate the failure

probability of each of these elements based on output from the VDT simulation. Finally, we can use the PRA fault tree and functional failure probabilities to calculate the probability of total failure.

There are limits to the extent that engineered systems operations depend upon the quality of engineering organizations' performance. In particular, the engineering teams' target specifications often stipulate that reliability under certain circumstances need not be assured, because the difficulty of achieving low-risk operations are too great under those circumstances. For example, the power systems of NASA's Mars Rovers, Spirit and Opportunity, are not designed to endure the occasional storms that occur on the red planet [JPL]. We can model this fact using a probability of failure in each engineering element that is not engineering-conformance-dependent. Similarly, each engineered element has a probability of success that is not engineering-conformance-dependent, occurring for example if operations exactly match trivial conditions that are fully simulated in testing.

Once the PRA framework is complete, we can trace the engineering roots of each component, interface, and multi-system to determine a scope of upstream projects whose performance we must predict using VDT. For each component, we make sure that a task and corresponding team appear in the model, while interfaces map to rework links and multi-systems map to meeting schedules. We must adapt our formula at integration points where the actual project plan deviates from this template. In addition, we must add to the VDT model any external processes or organizational elements that may impact engineering performance (such as management multi-tasking), in accordance with standard VDT model development practices.

Our proposed method calculates the probability of operations failure by relating lower-level engineered element failures to engineering conformance using three *integration points*. In software engineering, the term "*Integration points*" refers to data types and structures that are linked directly between two software models. The relationships between project structure and the four factors motivate our integration points, as we illustrate in Figure 6.

This model defines integration points that we believe are sufficient to predict some important influences that shortcomings in engineering conformance will have on the probability of operations failure. Our integration addresses events where the engineered components must operate in a non-trivial manner, according to the specification originally provided to engineering. At a high level, we claim that the number of unfixed design verification failure events that VDT predicts influences the component and interface failure probabilities in a PRA model. For parallel reasons, multi-system failures are probabilistically dependent upon the predicted attendance at design coordination meetings. For each element, we can develop a measure of engineered conformance, or flaw rate, that ranges from zero (unlikely to have significant flaws) to one (high probability of numerous flaws).

Figure 7 provides an influence diagram showing the proposed model's three integration points, each of which we describe in detail and illustrate below. We can assess these probabilities using standard PRA techniques, such as working with human experts to precisely define engineered elements' functional triviality, impossibility, and engineering-dependency, then predicting the probability of each case. We provide mathematically precise derivations in the next chapter.

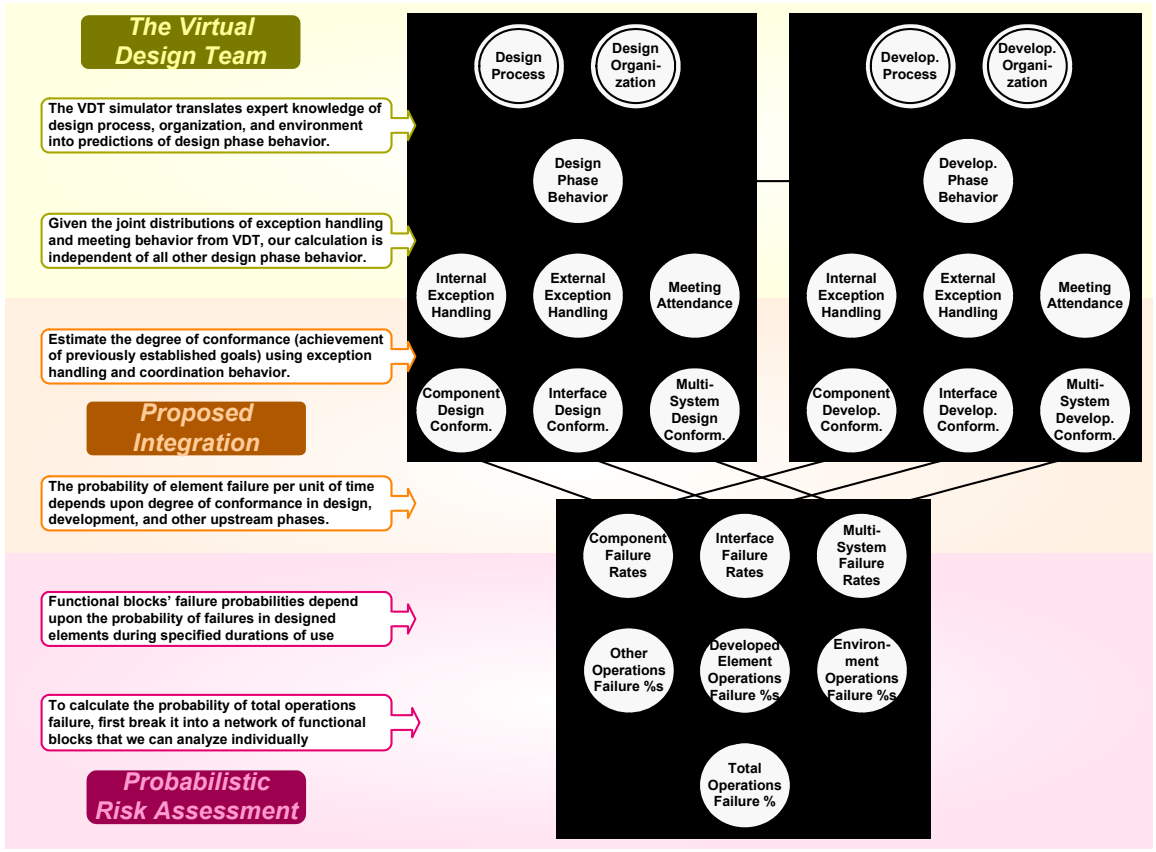


Figure 6 Influence Diagram Showing VDT-PRA Integration Points
VDT analyzes each engineering stage's organization and process and predicts three types of risk related data. We claim that each of these measures of engineering conformance relates to a distinct risk to the engineered product, and we operationalize this claim using three corresponding data integration points. PRA calculates the significance of these risks within the broader product and environmental context. Quantitatively linking outcome failure probability to upstream decisions like design team centralization would be a new capability for project planners and accident investiators alike. Figure 1 shows the objects that are involved, Figure 6 illustrates the analytic integration, and Table 4 provides the calculations for a specific example.

1. Internal Exceptions and Component Failures

Intuition

For our first integration point, we argue that the effective management of an engineering task's internal exceptions decreases (but does not eliminate) the failure probability of its corresponding functional component. Our method is to translate design subtask processing results into an estimate of the probability of component design flaws that can induce operations failure.

Theoretical Basis

In a well-structured project, we claim that the information dependency relationships among engineering tasks are often isomorphic to the operational behavior dependencies among engineered product components. When it comes to internal exception handling behavior, dependence among engineering tasks is limited to the sharing of resources such as management, which Thompson defined as "pooled" dependence in 1967. We claim that a component is an appropriate definition of tasks' engineered products because this relative independence in engineering leads to a clear distinction between relatively independent artifacts in operations.

Internal Exceptions

The first measure we use to predict failure risk is the rate of rework (or design iteration) that is ordered for individual tasks in an effort to guarantee correctness. The best task processing behavior that VDT can forecast consists of each subtask being completed routinely, and we define this to produce the highest achievable degree of confidence in the element's conformance. In the other extreme case, each subtask produces an exception, but the organization is too overloaded to respond to *any* of them. This case is quite unlikely, since in this case the simulation would almost certainly show a complete program failure. Nevertheless, it defines the other end of our scale, and in this case, it is very unlikely but still possible that the engineered element conforms to specification.

Component Failures

Components are elements whose engineering and operation is largely autonomous to other work. Common examples include an electrical, structural, or air circulation system, although an explicit organizational design or cost estimate can also be a product.

Integration Points

In our model, a component is generally distinguished as a single PRA functional block in operations, and its design and development are individual, dedicated VDT tasks. We define a *subcomponent* as a part of a component that may or may not be used at any given time in operations, and that has corresponding subtasks (low-level work items) in design and development. At any given moment, we assume that operations are equally likely to actively rely on the performance of any of these subcomponents. Functional failure

depends upon the active subcomponents' design effectiveness (measured as conformance of design subtasks) and development effectiveness (measured as conformance of development subtasks). We estimate the probability at any point in time of a failure in a component, given VDT predicted engineering behavior, as the average conformance over all of the subcomponents.

Aerospace Examples

An example of development-dependent component failure is the Challenger accident, in which the O-rings were developed to dimensions and material that could not perform as required under launch temperatures within the specified range [Vaughan 1996]. As a prominent design-dependent component failure, a comet return vehicle was developed and deployed in accordance with the design, but unfortunately, the original design oriented a battery in the reverse position- fundamentally inoperable. We claim that in both of these cases, more effective engineering and internal exception handling methods might have been able to prevent the original error, or to discover and correct the error through rework.

2. External Exceptions and Interface Failures

Intuition

In practice, flaws very often occur in the interface between two components [Weiss 2004]. This frequently results from ineffectively handling coordination between the components' teams in design and development. This relationship manifests in VDT as rework links between two tasks. Therefore, we calculate the probability of interface failure (which is independent of failure in either component) using a parallel method as with components, but using external exception handling.

Theoretical Basis

In a well-structured project, we claim that the information dependency relationships among engineering tasks are often isomorphic to the operational behavior dependencies among engineered product components. When it comes to external exception handling behavior, dependence among engineering tasks consists of the direct transfer of information or physical product from one task to another, which Thompson defined as "sequential" dependence in 1967. We claim that an interface is an appropriate definition

of rework-link-driven engineered elements, because it is an easily circumscribed interaction between the behaviors of artifacts in operations that motivates the engineering stage dependency.

External Exceptions

The next measure we use to predict failure risk is the rate of rework (or design iteration) that is ordered for individual tasks in an effort to accommodate engineering-stage choices that were made in related tasks. We can calculate this measure by applying to external (project) exceptions the same method as we used for Integration Point 1: Internal Exceptions and Component Failures.

Interface Failures

We define *interfaces* as the engineered elements that determine whether two components will interoperate in the desired (specified) manner. For example, the interfaces between an electrical component and a telecommunications component guarantees that power is supplied adequately, that the power does not interfere with communications signals, and that the components fit together physically.

Integration Points

In our model, interfaces correspond to PRA functional blocks that appear where either of the related components blocks does, and their design and development corresponds to rework links between the components' engineering tasks. We claim that the higher the rework deficit at a particular rework dependency link in engineering, the more likely it is that there will be faults at the interface between elements engineered by the linked stations in operations.

Aerospace Examples

A prominent example of design-dependent interface failure is from the Columbia accident, where insulation that was systematically loosened by a fuel line broke off during launch and caused a breach in the thermal protection system. We claim that in this case, more effective external exception handling methods might have been able to prevent the original error, or to discover and correct it through rework.

3. Meetings and Interdependent Multi-System Failures

Intuition

Our model presumes that interactions may occur among any subset of components that are represented by invitation to a meeting, and so meeting attendance provides all of the information from the design phase results that we evaluate in estimating a multi-system function's failure probability.

Theoretical Basis

In a well-structured project, we claim that the information dependency relationships among engineering tasks are often isomorphic to the operational behavior dependencies among engineered product components. When it comes to meeting attendance, dependence among engineering tasks consists of mutual adjustment among many tasks with mutually conflicting subgoals, which Thompson defined as "reciprocal" dependence in 1967. We claim that an interdependent multi-system is an appropriate definition of meeting attendance-driven engineered elements, because it is the complex interactions among the behaviors of engineered elements in operations that motivates engineering stage meetings.

Meetings

Our final source of risk data is the rate of attendance at routine meetings that enable interdependent design actors to develop a holistic understanding of a complex interdependent system. We can think of meetings as opportunities for a group as a whole to collaboratively identify problems, where the chance of identifying each potential problem equals the fraction of invitees who attend. Low meeting attendance indicates that engineers are unlikely to have clearly understood and responded to the relationships among groups of related work items.

Interdependent Multi-System Failures

A multi-system is a set of components and interfaces that interoperate tightly enough to be recognized as having the potential to interact in complex and unforeseen ways. This integration point estimates the probability that the interactions among interdependent systems will result in a functional failure, independently of whether each component and interface operates properly in isolation.

Integration Points

During design and development, the engineers working on components within a multi-system are typically assigned to attend meetings that focus on understanding and managing these interactions. We model each multi-system in PRA using functional blocks that appear in series with each included component, so that a multi-system failure is functionally equivalent to a failure of all of these systems. In VDT, we recognize regular coordination meetings as the opportunity to mitigate some multi-system failure risks.

Aerospace Examples

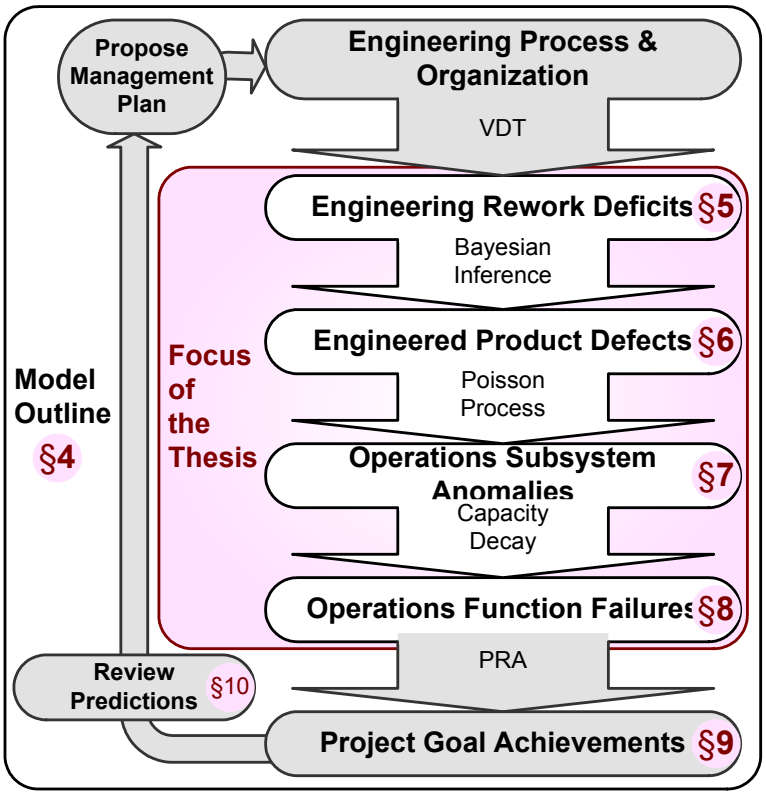
The Mars Polar Lander [JPL 2000] provides an example of a multi-system failure. The lander's extension of landing gear (landing gear component) caused a vibration in the craft (structures component). This was interpreted as contact with the surface (software), resulting in the premature termination of braking activities (propulsion) and loss of the mission. We claim that in this case, more attention to the need for systems integration coordination might have led to engineering decisions that could have prevented the original error.

Chapter 4 - Model Overview

4.1 INTRODUCTION

This thesis provides a mathematical definition of our method of relating project failure risk to engineering process measures using VDT and PRA. The first chapter formulates the top-level problem of calculating the expected utility for a set of management choices, and serves to introduce our overall method and reasoning. We assume here that the rational manager will choose the option that maximizes his/her expected utility. Each of the remaining chapters provides a method for calculating one of the steps in our project failure method, estimating in turn: engineers' rework-related actions, product conformance, anomalies during operations, the loss of operating capacity, functional failures, and project failures.

After a brief, intuitive overview, each chapter lists, categorizes, and describes variables that are introduced to the model. They then explain the theoretical foundations of each step in intuitive as well as formal terms, and finally offer a consistent space mission example. Figure 7 shows the links between VDT input and PRA output.



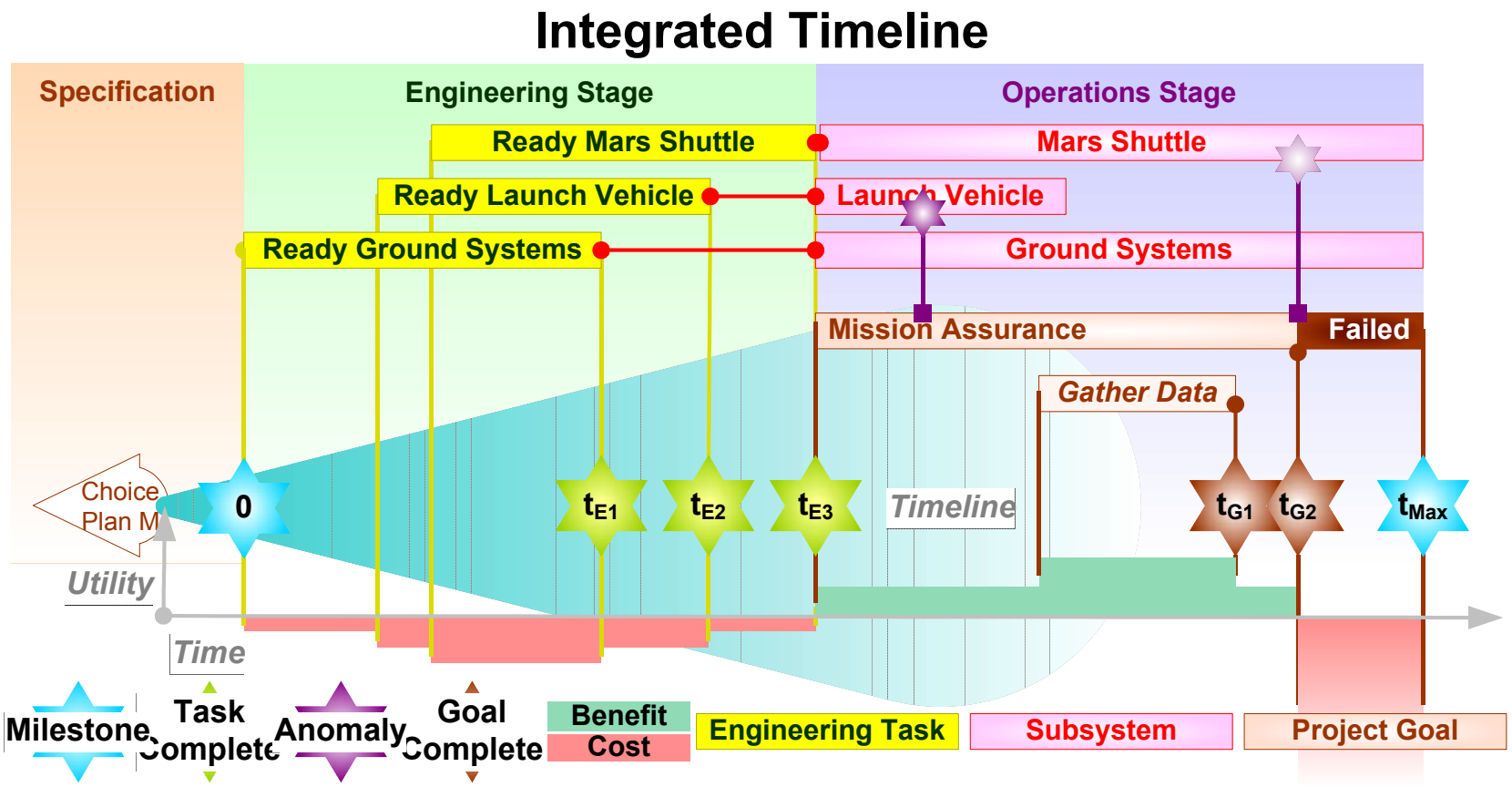


Figure 7 Integrated Timeline of Specification, Engineering, and Operations
This schematic illustrates a Hypothetical case in which a mars shuttle mission experiences an anomaly in the launch vehicle and a second in the shuttle itself. Although the scientific data was gathered according to plan, the anomalies combined to cause the loss of the shuttle itself, at tremendous economic and political cost.

Foundations in the SAM Model

We substitute and expand our nomenclature within the SAM tradition (Murphy and Paté-Cornell 1996). An important point of difference between SAM and the proposed model is that the latter limits itself, and goes into greater depth, into those decisions and actions that influence the rework deficit created during the engineering phase. This close relationship between our proposed model and the SAM formulation lends confidence in its elegance. However, we organize this paper around a more detailed formulation of Eq. 4.2 that clarifies the relationships among additional factors. A more detailed comparison between the proposed method and SAM is in Chapter 2 Section 4.

Notation

Table 5.1.1 defines the various formats that we use to describe random variables, constants, and other data types. We make these conventions explicit because of a conflict between traditions in matrix algebra (which uses capitals for matrices and lower case for scalars) and probability (which uses upper case for random variables and lower case for constants).

Table 4.1.1: Format of Data Representations

Format	Sample	Data Represented
Upper Case	X	Random variable
Lower Case	x	Realization of a random variable X
Upper Case Bold	$\mathbf{X}$	Matrix, vector, or set of random variables
Upper Case Subscripted	X_{ij}	Random variable; appears in **X**
Lower Case Bold	$\mathbf{x}$	Matrix, vector, or set containing one realization x_{ij} for each random variable X_{ij} in **X**
Lower Case Subscripted	x_{ij}	Realization of a random variable X_{ij}; appears in **x**
Greek Upper Case	Δ	Matrix, vector, or set of constants
Greek Lower Case	δ	Constant
Upper Case Italics	$G(x)$	Matrix, vector, or set of functions
Lower Case Italics	$g(x)$	Function
Upper Case Subscript	$_X$	Index identifying a particular variable or structure
Lower Case Subscript	$_x$	Index identifying an arbitrary variable or structure

In order to clarify the equations, we group related random variables into vectors, matrices, or sets denoted with bold, upper case lettering. We use lower case versions of the same letters to indicate realizations of those random variables, and lower case bold to indicate joint realizations of all the variables in a vector, matrix, or set. When we are to execute a function g, for example, on all of the x_{ij} realizations in a matrix **x** that are sampled from random variables X_{ij} in **X**, we have

$$g(\mathbf{x}) = g(x_{11}, x_{12}, \ldots x_{1c}, x_{21}, \ldots x_{r1}, \ldots x_{rc}) \quad \text{Eq. 1}$$

We define $P(\mathbf{x})$ as the joint probability distribution on all the random variables X_{ij}:

$$P(\mathbf{x}) = P(\mathbf{X} = \mathbf{x}) \quad \text{Eq. 2}$$

$$= P(X_{11} = x_{11}, \ldots X_{ij} = x_{ij}, \ldots X_{rc} = x_{rc}) \quad \text{Eq. 3}$$

$$= P(x_{11}, \ldots x_{ij}, \ldots x_{rc}) \quad \text{Eq. 4}$$

Because the random variables are not necessarily distributed independently, this equals the product of the probability of each variable taking on a particular value, conditional on all previously computed probabilities:

$$= \prod_{ij} P(x_{ij} \mid x_{11}, x_{12}, \ldots x_{1c}, x_{21}, \ldots x_{i1}, \ldots x_{i(j-1)}) \quad \text{Eq. 5}$$

Similarly, $P(\mathbf{x} \mid \mathbf{y})$ is a *joint probability density* on the realizations x_{ij}, each conditional on all of the realizations y_{ij}.

$$P(\mathbf{x} \mid \mathbf{y}) = \prod_{ij} P(x_{ij} \mid x_{11}, x_{12}, \ldots x_{1c}, x_{21}, \ldots x_{i1}, \ldots x_{i(j-1)}, y_{11}, \ldots y_{rc}) \quad \text{Eq. 6}$$

For example, where we are to calculate the average value of $g()$ over the joint distributions of all random variables X_{ij} in **X**, we write:

$$\int_{\mathbf{x}} g(\mathbf{x}) f(\mathbf{x}) d\mathbf{x} = \int_{x_{11}} \ldots \int_{x_{ij}} \ldots \int_{x_{rc}} g(x_{11}, \ldots x_{ij}, \ldots x_{rc}) f(x_{11}, \ldots x_{ij}, \ldots x_{rc}) dx_{11} \ldots dx_{ij} \ldots dx_{rc} \quad \text{Eq. 7}$$

Table 4.1.2: Notation Introduced for Model Overview

Term	Name	Data	Description	Example
m	management choices	Set of management choices and assessed values	Assessed values: project plan consisting of many simultaneous choices and facts of importance in predicting the impacts of those choices. We provide **m** as input to the proposed algorithm and calculate the implications of them, and recommend using the set of choices that produce the greatest expected utility.	
$\mathbf{T_e}$	Distribution of all engineering task completion Times	Set of continuous random variables T_{ei}	Output from many VDT runs: joint distribution on durations of all the project's engineering tasks (design, development, or testing), rework links, and meetings. Elements T_{ei} for each engineering task i are uncertain because exception handling and coordination sometimes extend the target schedule.	$\mathbf{T_e} = (T_{e1}, T_{e2}, T_{e3 \ldots})$ indicates that there are three engineering tasks whose completion times t_{e1}, t_{e2} and t_{e3} are distributed as random variables T_{e1}, T_{e2} and T_{e3}.
i	Engineering task or engineered element	Discrete Index	Identifies an engineering task (design, development, or testing), rework link, or meeting that we simulate using VDT, and that has a single corresponding engineered element (component, interface, or multi-system respectively).	Index $_i = 1$ for example identifies the "Ready Ground Systems" task, while $_i = 4$ identifies the "Ground – Launch Interface" rework link.
T_{ei}	Distribution of engineering task completion Times	Continuous random variable	Output from many VDT runs: distribution on duration of an engineering task (design, development, or testing), rework link, or meeting. T_{ei} for each engineering task i are uncertain because exception handling and coordination sometimes extend the target schedule.	$T_{e1} \sim expo(1)$ indicates that the time between the beginning and the end of engineering task 1, "Ready Ground Systems", is distributed exponentially with expected duration 365 days.

Term	Name	Data	Description	Example
$\mathbf{t_e}$	Sample of all engineering task completion times	Set with one realization for each variable in $\mathbf{T_e}$	Output from one VDT run: sampled durations of all the project's engineering tasks (design, development, or testing), rework links, and meetings. Values indicate the extent to which exception handling and coordination will actually extend the target schedule.	$\mathbf{t_e} = (t_{e1}, t_{e2}, t_{e3} \ldots t_{e7})$ indicates that each of the seven engineering tasks i took an amount of time equal to t_{ei}. $\mathbf{t_e} = (1.0, 1.0, 1.5 \ldots)$ indicates that the first two engineering tasks completed after one year, while the third took a year and a half.
t_{ei}	time of engineering task completion	Real-valued Realization of T_{ei}; element of $\mathbf{t_e}$	Output from one VDT run: actual duration of an engineering task (design, development, or testing), rework link, or meeting i, as determined by emergent exception handling and coordination behavior.	$t_{e6} = 0.75$ indicates that the sixth engineering task completed after nine months.
$\mathbf{T_g}$	Distribution of all goal completion Times	Set of continuous random variables T_{gg}	Output from proposed algorithm: joint distribution on failure times of all the project's goals. Elements T_{gg} for each project goal g are uncertain because subsystem anomalies arise probabilistically over time.	$\mathbf{T_g} = (T_{g1}, T_{g2})$ indicates that there are two project goals whose completion times t_{g1} and t_{g2} are distributed as random variables T_{g1} and T_{g2}.
$_g$	Project goal	Discrete Index	$_g$ is the index on project objectives and indexes all of their requirements, anomalies, and failures during operations	Goal g = 1 identifies values related to completion of the trip to and from Mars, while $_g = 2$ indicates that a variable is relevant to success in achieving a scientific data gathering goal.
$\mathbf{t_g}$	times of all goal completions	Set with one realization for each variable in $\mathbf{T_g}$	One set of possible failure times for all the project's goals (i.e., time between commencement of operations and the first occurrence of a failure mode of each goal). Elements t_{gg} for each project goal g take into account a set of subsystem anomalies that has arisen over time.	$\mathbf{t_g} = (t_{g1}, t_{g2})$ indicates that each of the two project goals g will fail after a time equal to t_{gg}. $\mathbf{t_g} = (1.0, 0.5)$ indicates that the first goal will fail if operated for longer than one year, while the second is only sustained for six months.

Term	Name	Data	Description	Example
T_{gg}	Distribution of goal completion Times	Continuous random variable	Output from proposed algorithm: distribution on failure times for a project's goal g. Failure times are uncertain because subsystem anomalies arise probabilistically over time.	$T_{e1} \sim expo(1)$ indicates that the time between the beginning and the end of engineering task 1, "Ready Ground Systems", is distributed exponentially with expected duration 365 days.
t_{gg}	time of goal completion	Real-valued realization of T_{gi}; element of $\mathbf{t_g}$	Particular failure time for a project's goal g, taking into account the specific subsystem anomalies that have arisen over time (i.e., time between commencement of operations and the first occurrence of a failure mode of goal g)	$t_{g2} = 0.5$ indicates that the second project goal (scientific data gathering) will fail if operated for longer than six months.
$f(x)$	Probability Density Function	Function	For a random variable X with realization x and a set B, $f(x)$ is shorthand for $f_X(x)$, the function such that $P(X \in B) = \int_B f(x)dx$. "All probability statements about the random variable can (in principle) be computed from $f_X(x)$" [Law and Kelton 2000].	Where X is the uniform distribution, $f(x) = 1$ for all $x \in [0,1]$, $f(x) = 0$ for all $x \notin [0,1]$
R	Distribution of all rework related engineering outcomes	Set of discrete random variables R_{hi}	Output from many VDT runs: joint distribution on exception handling and meeting attendance for all of the project's engineering tasks (design, development, or testing), rework links, and meetings. Elements R_{hi} for each type of handling h by each engineering task i are random variables because exception handling and coordination activities vary and cannot be predicted with certainty.	$\mathbf{R} = (R_{11}, R_{12}, R_{13 \ldots})$ indicates that there are distributions on the number of times that engineering task 1 handles its limited number of subtasks in each of three ways. The numbers of times subtasks are handled in these ways are distributed as random variables R_{11}, R_{12} and R_{13}.

Term	Name	Data	Description	Example
h	Exception handling	Discrete Index	Identifies an exception handling or meeting outcome from engineering subtasks (small portions of design, development, or testing work) that we simulate using VDT.	For tasks and rework links, values for h are: 1 = verified, no exception; 2 = reworked, an exception and decision to redo the full subtask; 3 = quick-fixed, an exception and decision to do partial rework; 4 = ignored, an exception and decision to do no rework; 5 = defaulted, an exception that was never addressed by a decision of whether or not to conduct rework. For meetings, values for h are 1 = attended and 5 = not attended.
r	Amount of all rework related engineering outcomes	Set with one realization for each variable in **R**	Output from one VDT run: sampled exception handling and meeting outcomes from engineering subtasks (small portions of design, development, or testing work). Values indicate the extent of exception handling, rework, and meeting activities.	**r** = (r_{11} , r_{21}, $r_{31\ ...}$) = (12 , 3 , 4 …) indicates that engineering task 1, "Ready Ground Systems", resulted in r_{11} = 12 verified subtasks, r_{21} = 3 reworked subtasks, r_{31} = 4 quick-fixed subtasks, etc.
R_{hi}	Distribution of all rework related engineering outcomes	Discrete random variable	Output from many VDT runs: distribution on a type of exception handling or meeting attendance h for an engineering task (design, development, or testing), rework link, or meeting i. Quantity is uncertain because exception handling and coordination activities vary and cannot be predicted with certainty.	$R_{57} \sim bin(20, 0.5)$ indicates that the number of absentees at the systems integration meeting is distributed in the same way as the number of heads among 20 coin flips. In our model, VDT generates the distribution according to a complex model of organizational and procedural factors.

Term	Name	Data	Description	Example
r_{hi}	Amount of rework related engineering outcomes	Integer realization of R_{hi}, element of **r**	Output from one VDT run: sampled number of exception handling or meeting h outcomes from an engineering subtask (small portions of design, development, or testing work) i. Value indicate the number of instances of a particular exception handling, rework, or meeting activity.	$r_{42} = 3$ indicates that the second engineering task "Prepare Launch Vehicle" involved three ignored exceptions.
$\mathbf{m_e}$	engineering management choice	Set of decision variables	VDT Input: Engineering management plan, including a complete and consistent set of organizational and process design choices. We simulate $\mathbf{m_e}$ using VDT and calculate possible implications for later operations.	The set includes a diverse range of organization and process choices, including the definition of all engineering teams within a hierarchy, the tasks within a precedence network, and policies such as centralization and meetings. Chapter 5 and Jin and Levitt 1999 provide more detail on these values.
D	distributions of possible defects in all product elements	Set of continuous random variables D_{gi}	Set of distributions on the conformance levels for all engineered elements and goals, derived from the distributions of engineering rework deficits.	A matrix of all ones indicates that all of the engineering products satisfied all of the requirements at each goal.
D_{gi}	distribution on one product element's possible Defects	Continuous random variable; Element of **D**	Random variable distributed according to the fraction of functionality that is defective in engineered product element i with regard to the requirements of project goal g.	$D_{13} \sim bin(100, 0.5) / 100$ indicates that for goal 1 "Mars Round Trip" requirements placed on product component 2 "Ground Systems", the percentage chance of a conformance level equals the chance of that number of heads among 100 coin flips. Our method estimates the actual distribution using a Bayesian analysis of engineering rework deficits.

Term	Name	Data	Description	Example
d	All product elements' defects	Set with one realization for each variable in **D**	Set of conformance levels for all engineered elements i and all project goals g. The elements are d_{gi}, which measure the fraction of requirements that a product meets.	**d** = (d_{11} , d_{21}, …) = (0.9, 0.8 …) indicates that engineering task 1, "Ready Ground Systems", resulted in d_{11} = 90% conforming product as far as goal 1, "Mars Round Trip", but only 80% as far as goal 2 "Collect Science Data" requirements. This means that if both goals rely equally on ground systems during operations, the science data goal is more likely to be jeopardized by the ground systems than the mars round trip is.
d_{gi}	One product element's defects	Real-valued realization of D_{gi}; Element of **d**	Conformance of an engineered element i, represented as a random variable that serves as the probability that a randomly selected requirement for project goal g will be met in operations.	A value of d_{12} = 0.5 indicates that the launch vehicle component, the product of engineering task 2 "Prepare Launch Vehicle", only satisfies half of the relevant goal 1 (critical) requirements.
$\mathbf{m_v}$	Verification	Set of constant *(*expert-assessed*)* probabilities v_{ghi}	Provides the posterior probabilities of conformance for each goal specification, given exception-handling behavior. We develop this value based on expert opinions of the specificity and sensitivity of engineering teams' internal verification procedures. Cell values are v_{ghi}	If $\mathbf{m_v}$ has the value 0.95 in all of its cells, this indicates that product conformance will average 0.95, no matter what the engineering exception handling behavior was.
v_{ghi}	verification	Constant *(*expert-assessed*)* probability; Element of $\mathbf{m_v}$	Probability that a part of task i's product conforms to requirements of goal indexed by g, given that the organization produced exception handling h when processing that subtask.	A value of v_{123} = 0.98 indicates that any work for task 3 that produced exception handling 2 *(*rework*)* has a 98% probability of meeting any given mars return trip *(*goal 1*)* requirement during operations
$\mathbf{m_z}$	Testing coverage	Set of percentages z_{gin}	Provides the fraction of each task's product that is tested or reviewed by testing tasks, at each goal. Cell values are z_{gin}.	If all of the values for $\mathbf{m_z}$ are zero, no engineering tasks are explicitly dedicated to evaluating the correctness of other work.

Term	Name	Data	Description	Example
z_{gin}	Testing coverage	Percentage; element of matrix G	Test coverage—the fraction of task i's goal g – relevant requirements that are evaluated by testing task n.	z_{139} = 0.8 indicates that task 9 "Electrical Testing" will evaluate the correctness of eighty percent of the work produced by task 3, "Electrical Design", and will handle any perceived problems with goal 1 requirements using VDT-modeled exceptions and rework.
$_n$	Testing Task in Engineering	Discrete Index	Identifies a task in the engineering phase that is dedicated to evaluating the products of engineered elements (components, interfaces, or multi-systems)	Index $_n$ = 9 for example could identify the "Electrical Testing" task.
A	Distribution on all possible operations Anomalies	Set of continuous-time stochastic processes $A_{gj}(t)$	Distribution on the total number of various system anomalies that will manifest over time during operations, and that are able to influence each function. Results from the distribution of defects among the engineered elements and the pace and sensitivity of operations. Fractional values represent anomalies of lesser significance. Elements are $A_{gj}(t)$	**A** = ($A_{11}(t)$, $A_{12}(t)$, $A_{13}(t)_{...}$) = (t, t, t …) indicates that the number of anomalies is known, identical in all subsystems, and increases linearly over time.
$_j$		Discrete Index	$_j$ identifies an engineered subsystem of components, interfaces, and multi-systems that is active during operations and capable of producing anomalies	A variable identified with $_j$ = 1 would refer to system 1, "Electrical", that is required to perform certain functions

Term	Name	Data	Description	Example
$A_{gj}(t)$	Distribution of total anomalies in a subsystem at a time during operations	Continuous-time stochastic process; Element of **A**	Defines the distribution of total subsystem j anomalies that have manifested by time t during operations, and that are able to influence the achievement of goal g. Results from the distribution of defects among the engineered elements and the pace and sensitivity of operations. Fractional values represent anomalies of lesser significance.	$A_{11}(t) \sim$ Poisson (1) indicates that the first subsystem, "Ground Systems", develops around one anomaly per year in a Markovian fashion.
a	Total anomalies in all subsystems over time	Set of functions of time to real numbers	The number of anomalies that have manifested with the ability to affect each goal, in each subsystem, at each time.	**a** = ($a_{11}(t)$, $a_{12}(t)$, $a_{13}(t)$ …) = (0, 0, 0, …) is the ideal case- in it, no anomalies ever occur!
$a_{gj}(t)$	Total anomalies in a subsystem over time	Function of time to real numbers; Realization of $A_{gj}(t)$	The number of goal g-related anomalies that have manifested in system j by time t.	A value of $a_{13}(2) = 5$ indicates that five goal 1 (mars round trip) anomalies have manifested by year 2.
$\mathbf{m_w}(t)$	Operations Pace	Set containing w, w_{ijg}, or $w_{gij}(t)$	Rate of operations – the rate at which function j calls upon features provided in task i's goal g specification, at time t.	
$\mathbf{m_s}(t)$	Operations Sensitivity	Set containing s, s_{ijg}, or $s_{gij}(t)$	Sensitivity of operations – the probability of significant change in function j when encountering requirement violations of goal g from dependent task i, at a time t, given management choices m	

Term	Name	Data	Description	Example
w	Homogeneous operations pace	Real-valued constant	Global rate of operations – the rate at which a dedicated function calls upon features provided in each task specifications.	w = 0.9 indicates that the product of an engineering task is relied upon to meet each goal of requirements in each system during most time units, throughout operations.
s	Homogeneous operations sensitivity	Real-valued constant	Global sensitivity of operations – the probability of significant change in a dedicated function when encountering requirement violations from a given task	s = 0.9 indicates that when a non-conforming portion of the product of an engineering task is relied upon to meet requirements in another system, it usually results in an anomaly.
w_{ijg}	Static operations pace	Real-valued constant	Static rate of operations – the rate at which function j calls upon features provided in task i's goal g specification.	w_{123} = 0.9 indicates that the product of engineering task 2 is relied upon by subsystem 3 to meet goal 1 requirements during most time units throughout operations.
s_{ijg}	Static operations sensitivity	Real-valued constant	Static sensitivity of operations – the probability of significant change in function j when encountering requirement violations of goal g from dependent task i.	s_{123} = 0.9 indicates that when a non-conforming portion of the product of engineering task 2 is relied upon to meet goal 1 requirements in system 3, it usually results in an anomaly.
$w_{gij}(t)$	Dynamic operations pace	Function that maps a real number to a real number	Rate of operations – the rate at which function j calls upon features provided in task i's goal g specification, at time t.	$w_{123}(t) = 0.9^t$ indicates at first, nearly every time unit during operations engineered subsystem 3 relies for on the conformance of engineered element 2 to satisfy a new goal 1 requirement for the first time. However, operations test fewer new requirements as time goes on.

Term	Name	Data	Description	Example
$s_{gij}(t)$	Dynamic operations sensitivity	Function that maps a real number to a probability	Sensitivity of operations – the probability of significant change in function j when encountering requirement violations of goal g from dependent task i, at a time t, given management choices m	$s_{123}(t) = 0.9^t$ indicates that under management plan 4, at first a goal 1 anomaly occurs during operations in subsystem 3 nearly every time that a non-conforming requirement from task 2 is tested. However, as time goes on, operations become more robust to these events.
f	Time of Functional failure	Matrix of random variables f_{gk}	Cell values are f_{gk}. Represents the time at which failure or completion of each function k occurs as far as goal g is concerned.	A matrix with all ones in column one, and all zeros in column two, indicates that all of the critical functions succeed, but none of the goal g two (non-critical) functions do.
$\mathbf{t}_f(t)$	Functional failure	Matrix of boolean random variables f_{kv}	Represents the success or failure of all functions all goals. Cell values are f_{kv}	A matrix with all ones in column one, and all zeros in column tow, indicates that all of the critical functions succeed, but none of the goal g two (non-critical) functions do.
f_{gk}	Functional failure	Continuous Random Variable	Random variable that represents the time at which function k ceases to support goal g.	A value of $f_{12}(t) = 1$ indicates that operations function 2 succeeds, when evaluated against mars return trip (goal 1) functional criteria.
$_k$	Function Index	Discrete Index	$_k$ identifies a function in operations that is important to the determination of whether failure occurs	A variable identified with $_k = 1$ would refer to function 1, "Launch", that is required for project success
$\mathbf{m}_y$	Functions' capacity decay rates	Set of continuous random variables y_{mkv}	Rate of capacity decay that anomalies cause for functions during operations. Cell values are u_{mkv}	When U_m contains all ones, any engineered subsystem anomaly will cause a total loss of capacity in dependent functions.

Term	Name	Data	Description	Example
y_{kg}	Function capacity decay rate	Continuous random variable; Element of matrix $\mathbf{m_y}$	Rate of capacity decay that anomalies cause for a function k in preventing goal g failure (for management choices **m**).	A value of $u_{321} = .5$ indicates that under management choice set 3, function 2 responds to each critical anomaly *(*goal 1*)* by losing 50% of capacity.
$\mathbf{m_i}$	Impact of subsystems on functions	Set of continuous random variables i_{gjk}	Defines the reliance of functions upon engineered subsystems during operations. Cell values are u_{gjk}	When $\mathbf{m_u}$ is set to the identify matrix, this indicates an isomorphism between engineered subsystems and operations functions, so that each anomaly in an engineered subsystem will impact exactly one function.
i_{gjk}	Use of subsystem by function	Continuous random variable; Element of $\mathbf{m_i}$	Degree of dependence upon an engineered subsystem j by an operations function k in preventing a failure of goal g.	A value of $i_{123} = 0$ indicates that engineered subsystem 2 does not impact function 3 for the purposes of estimating mars return trip *(*goal 1*)* failure
$\mathbf{m_l}$	Load	Matrix of continuous random variables l_{mkv}	Characterizes load- the pressure placed on all functions at all criticalities during operations	Where $\mathbf{m_l}$ has all ones, each function will be placed under the maximum load, so that any function that loses capacity to anomalies will fail.
l_{gk}	load	Continuous random variable; Element of matrix L_m	Characterizes load- the pressure placed on function k during operations to suffer a failure of goal g (for management choices **m**).	A value of $l_{12} = 0.8$ indicates that under the third set of management choices, function two undergoes a heavy load—as defined by the 80^{th} percentile.
$\mathbf{m_g}$	Goals	Set of Boolean expressions	Defines the reliance of each goal upon the successful operation of a function k.	$\mathbf{m_g} = (g_1, g_1)$ indicates that there are two project goals.
g_g	goal	Boolean expression	Defines the reliance of goal g upon various functions.	$g_1 = 1 + (2*3)$ indicates that goal 1 succeeds as long as either function 1 or both functions 2 and 3 succeed.

Term	Name	Data	Description	Example
$\mathbf{m}_*$	Best operations management choice index	Discrete index	Identifies a set of management choices, including engineering and testing plans, and the use of engineered products during operations in systems that support project objectives.	Index $_*$ = 1 could identify a conservative, serial engineering approach followed by extended operations in a challenging environment, if the alternative (index 2) identifies an aggressive parallel engineering approach followed by shorter operations, when the environmental hazards outweigh the benefits of engineering acceleration.
$U(X)$	Utility	Function (returns a real value)	Measure of the value to a decision maker that a state of the world represented as X represents	If U(X) > U(Y), then we know that a decision maker prefers state X to Y.
$E(x)$	Expected Value	Function (returns a real value)	Average value of a random variable x, integrated (or summed) over all x of the probability that x = y times the value y.	$E(Bernoulli(0.5)) = 0.5$, meaning that the average number of heads in a fair coin toss is one half

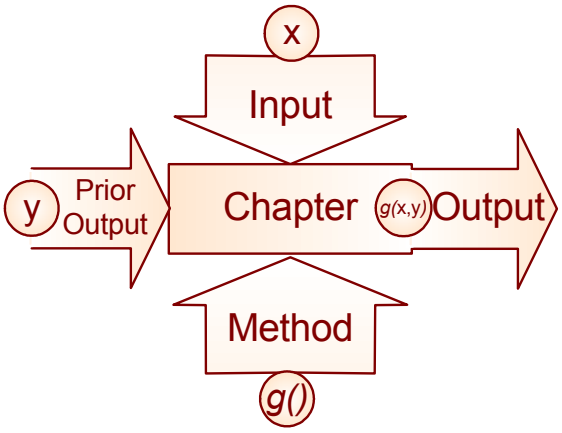
4.2 PROBLEM DECOMPOSITION USING PINCH POINTS

Formulating the Research Question Mathematically

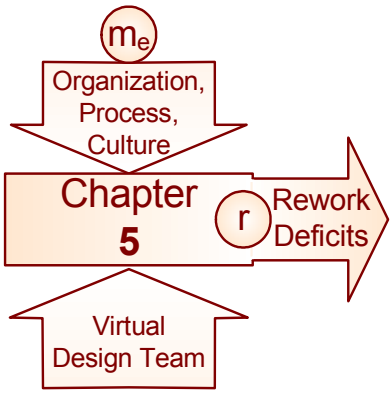
The proposed algorithm is intended to help a decision maker to maximize his or her expected utility from a set of consistent management choices **m**. Our method analyzes decisions by considering two types of data to be of direct importance, in addition to the management choices themselves. The first is the set of random variables $\mathbf{T_e}$, which contains distributions T_{ei} on the predicted durations t_{ei} of each engineering task i. We also impute primary importance to $\mathbf{T_g}$, which contains distributions T_{gg} whose realizations t_{gg} indicate predicted times of failure or completion for each project goal g.

Given these distinctions, our primary research goal is to provide a method for estimating the mean time to failure, mean engineering phase length, and related measures. We achieve this by producing an algorithm to calculate $f(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m})$ (shorthand for $f_{\mathbf{T_e}, \mathbf{T_g} \mid \mathbf{m}}(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m})$), the joint probability density function on engineering task completion and goal achievement/premature failure times that result from a set of management decisions **m**.

This chapter divides this expression into several more individually manageable portions using "pinch points". The illustration at right shows how each portion is the subject of a specific chapter, follows a distinctive method, requires management choices or project constraint assessments from management, and produces output that brings us a step closer to our final goal. In addition, most parts of the algorithm incorporate results from the previous section. Each of the subsections below introduces a pinch point and explains how we use it to divide our complex probability distribution into two more manageable portions.



Rework Deficits



In this section, we introduce our first pinch point, dividing our original probability density expression $f(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m})$. The first pinch point is based on rework deficit—the amount of rework that may well be appropriate, but that was not conducted, among various tasks performed during the engineering stage. We describe rework deficit using a set **R** of random variables R_{hi} whose realizations r_{hi} in **r** signify the number of rework handling choices h for each task i. We use this new structure to develop the first term in Equation 8, which refers to engineering stage behavior:

$$f(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m}) = \int_{\mathbf{r}} f(\mathbf{t_e}, \mathbf{r}, \mathbf{t_g} \mid \mathbf{m})\, d\mathbf{r} \qquad \text{Eq. 8}$$

$$f(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m}) = \int_{\mathbf{r}} f(\mathbf{t_e}, \mathbf{r} \mid \mathbf{m}) * f(\mathbf{t_g} \mid \mathbf{t_e}, \mathbf{r}, \mathbf{m}))\, d\mathbf{r} \qquad \text{Eq. 9}$$

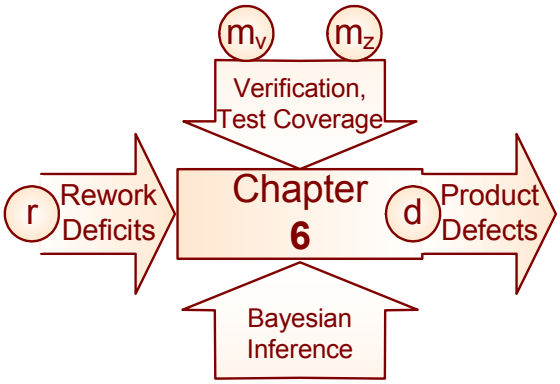
Next, we **assume** that the probability of achieving project goals is influenced by rework deficit, but is not directly linked to engineering stage schedules. Mathematically, this means that product goal completion times in $\mathbf{t_g}$ are independent of engineering task durations in $\mathbf{t_e}$, when given the rework deficit set **r**. We can thus simplify:

$$f(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m}) = \int_{\mathbf{r}} (f(\mathbf{t_e}, \mathbf{r} \mid \mathbf{m}) * f(\mathbf{t_g} \mid \mathbf{r}, \mathbf{m}))\, d\mathbf{r} \qquad \text{Eq. 10}$$

Finally, we note that the only engineering management decisions that influence our estimates of rework deficit and engineering task durations are those that the VDT simulation uses to define an organization, product, and culture. We define these using a set $\mathbf{m_e}$ (here $_e$ is mnemonic for engineering phase structure).

$$f(\mathbf{t_e}, \mathbf{d} \mid \mathbf{m}) = \int_{\mathbf{r}} (f(\mathbf{t_e}, \mathbf{r} \mid \mathbf{m_e}) * f(\mathbf{t_g} \mid \mathbf{r}, \mathbf{m}))\, d\mathbf{r} \qquad \text{Eq. 11}$$

Product Defects



As described in the Frame and Foundation, we **assume** that all the information from engineering that bears on operations behavior rests in the product. We are uncertain about the numbers of defects in each element of the engineered product, and we describe them using a set **D** of random variables D_{gi}. **d** provides realizations d_{gi} that represent the fraction of specifications that are relevant to goal g that an element i the final engineered product meets.

In the next step, we assume independence of operations on engineering given defects, and independence on engineering duration and defects from operations durations and choices, given engineering management choices:

Intuitively, this is the probability of a certain behavior during the engineering phase, times the probability of a certain resulting operations stage behavior given that engineering behavior, times the utility that results from this combination of behaviors.

We develop the second term from Equation 8 thus:

$$f(\mathbf{t_g} \mid \mathbf{r}, \mathbf{m}) = \int_{\mathbf{d}} f(\mathbf{d}, \mathbf{t_g} \mid \mathbf{r}, \mathbf{m})\, d\mathbf{d} \qquad \text{Eq. 12}$$

$$f(\mathbf{t_g} \mid \mathbf{r}, \mathbf{m}) = \int_{\mathbf{d}} (f(\mathbf{d} \mid \mathbf{r}, \mathbf{m}) * f(\mathbf{t_g} \mid \mathbf{d}, \mathbf{r}, \mathbf{m}))\, d\mathbf{d} \qquad \text{Eq. 13}$$

Furthermore, just two sets of engineering management decisions that influence our estimate of defects, given a rework deficit. The first is the reliability of the internal verification procedures that instigate rework, which we define in the matrix **mv**. The second set, **mv**, defines which engineering tasks are dedicated to the verification and testing of prior task's product. Therefore, we can simplify further:

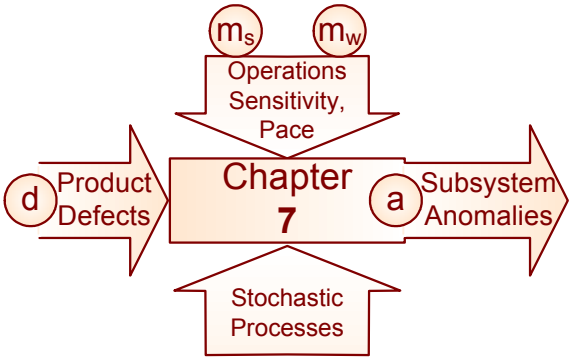
$$f(\mathbf{t_g} \mid \mathbf{r}, \mathbf{m}) = \int_{\mathbf{d}} (f(\mathbf{d} \mid \mathbf{r}, \mathbf{m_z}, \mathbf{m_v}) * f(\mathbf{t_g} \mid \mathbf{d}, \mathbf{r}, \mathbf{m}))\, d\mathbf{d} \qquad \text{Eq. 14}$$

We also **assume** that given a set of management decisions and product defects, the time to failure during operations of project objectives is independent of rework deficit. This simplifies our second term:

$$f(\mathbf{t_g} \mid \mathbf{r}, \mathbf{m}) = \int_{\mathbf{d}} (f(\mathbf{d} \mid \mathbf{r}, \mathbf{m_z}, \mathbf{m_v}) * f(\mathbf{t_g} \mid \mathbf{d}, \mathbf{m}))\, d\mathbf{d} \qquad \text{Eq. 15}$$

Operations Anomalies

The next pinch point is based on the concept of an anomaly—a behavior that occurs during operations that deviates from the specifications. Our model only considers anomalies that result from product defects, and we also assume that given a set of anomalies distributed in time, no prior behavior influences the failure of functions and achievement of project goals.



We develop the second term of equation 11 thus:

$$f(\mathbf{t_g} \mid \mathbf{d}, \mathbf{m}) = \int_{\mathbf{a}} f(\mathbf{t_g}, \mathbf{a} \mid \mathbf{d}, \mathbf{m})\, d\mathbf{a} \qquad \text{Eq. 16}$$

$$f(\mathbf{t_g} \mid \mathbf{d}, \mathbf{m}) = \int_{\mathbf{a}} (f(\mathbf{a} \mid \mathbf{d}, \mathbf{m}) * f(\mathbf{t_g} \mid \mathbf{a}, \mathbf{d}, \mathbf{m}))\, d\mathbf{a} \qquad \text{Eq. 17}$$

We **assume** that the duration of satisfaction of each goal is independent of product defects, given a set of anomalies. Conceptually, the number of defects in a product is not what determines success or failure—instead it is the behavior during operations that we are concerned with. This assumption offers a simplification:

$$f(\mathbf{t_g} \mid \mathbf{d}, \mathbf{m}) = \int_{\mathbf{a}} (f(\mathbf{a} \mid \mathbf{d}, \mathbf{m}) * f(\mathbf{t_g} \mid \mathbf{a}, \mathbf{m}))\, d\mathbf{a} \qquad \text{Eq. 18}$$

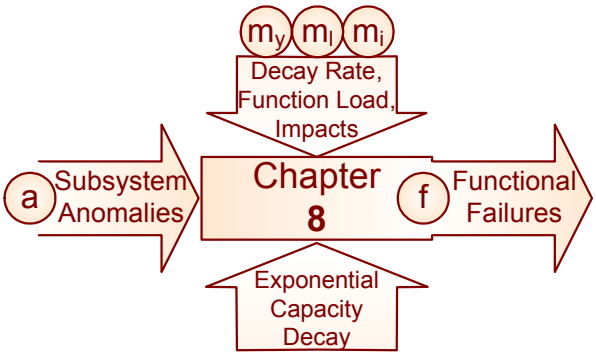
At this point, we can be more specific about which management choices influence the emergence of anomalies. We define the *operations pace* $\mathbf{m_w}$ as the average time between new requirements being placed on each engineered subsystem, and *operations sensitivity*

$\mathbf{m_s}$ as the chance that a requirement that is not met will cause an anomaly. Using these terms makes our equation more specific:

$$f(\mathbf{t_g} \mid \mathbf{d}, \mathbf{m}) = \int_{\mathbf{a}} (f(\mathbf{a} \mid \mathbf{d}, \mathbf{m_w}, \mathbf{m_s}) * f(\mathbf{t_g} \mid \mathbf{a}, \mathbf{m}))\, d\mathbf{a} \qquad \text{Eq. 19}$$

Functional Failures

We base our last pinch point on the concept of functional success and failure. The concept of a functional block that succeeds or fails during operations is fundamental in PRA and systems engineering. In our model, we introduce a set of random variables F whose realizations f identify the time at which each component ceases to perform in its role viz. each project goal.



We introduce this to the second term of equation 15 to obtain:

$$f(\mathbf{t_g} \mid \mathbf{a}, \mathbf{m}) = \int_{\mathbf{f}} f(\mathbf{t_g}, \mathbf{f} \mid \mathbf{a}, \mathbf{m})\, d\mathbf{f} \qquad \text{Eq. 20}$$

$$f(\mathbf{t_g} \mid \mathbf{a}, \mathbf{m}) = \int_{\mathbf{f}} f(\mathbf{f} \mid \mathbf{a}, \mathbf{m}) * f(\mathbf{t_g} \mid \mathbf{a}, \mathbf{m}, \mathbf{f})\, d\mathbf{f} \qquad \text{Eq. 21}$$

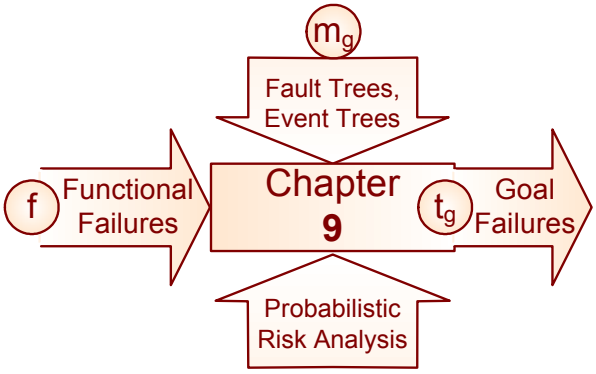
Next, we break up the integral based on the assumption that the behavioral aberrations we call anomalies cause functional failure, but that given a set of functional failure times, the success in reaching project goals is independent of the emergence of anomalies. Mathematically, this means:

In our model, three management factors influence the times of functional failure. Capacity begins at a high nominal level but exponentially decays according to $\mathbf{m_y}$ whenever an anomaly occurs that impacts the function. This latter event results from combining **a**, with an understanding (codified in $\mathbf{m_i}$ of how each anomaly can impact a function. In our model, loads distributed according to $\mathbf{m_l}$ are placed on each function, and failure occurs when this exceeds the function's capacity.

$$f(\mathbf{t_g} \mid \mathbf{a}, \mathbf{m}) = \int_{\mathbf{f}} f(\mathbf{f} \mid \mathbf{a}, \mathbf{m_y}, \mathbf{m_i}, \mathbf{m_l}) * f(\mathbf{t_g} \mid \mathbf{m}, \mathbf{f})\, d\mathbf{f} \quad \text{Eq. 22}$$

Goal Failures

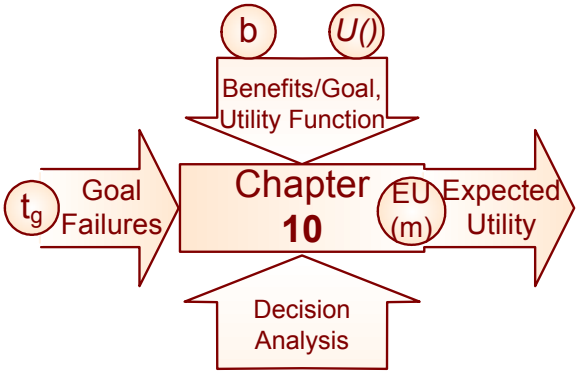
Finally, $\mathbf{m_g}$ is the subset of management decisions $\mathbf{m}$ that defines the structural relationships among functional blocks (expressed as minimal cut sets) when considering each goal. $\mathbf{m_g}$ identifies, for example, when a function is "single string", meaning that its failure means not reaching a given goal.



$$f(\mathbf{t_g} \mid \mathbf{m}, \mathbf{f}) = f(\mathbf{t_g} \mid \mathbf{m_g}, \mathbf{f}) \quad \text{Eq. 23}$$

Expected Utility

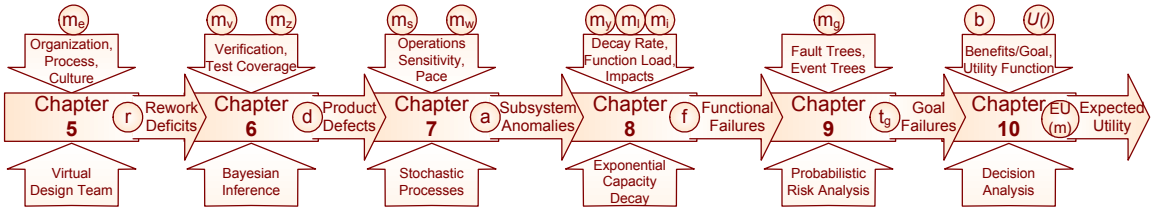
Deciding on the best choice $\mathbf{m}_*$ for a decision maker requires his or her utility function $U()$. Our method presumes that the only values of importance to the decision maker are the original choices $\mathbf{m}$, the times of engineering task completion $\mathbf{t_e}$, and the times of goal achievement or failure $\mathbf{t_g}$ (indicated by premature conclusion). The expected utility of the joint management choices $\mathbf{m}$ that constitute a project plan is:



$$U(\mathbf{m}) \sim U(\mathbf{T_e}, \mathbf{T_g}, \mathbf{m}) \quad \text{Eq. 24}$$

$$EU(\mathbf{m}) = \int_{\mathbf{t_e}} \int_{\mathbf{t_g}} (f(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m}) * U(\mathbf{t_e}, \mathbf{t_g}, \mathbf{m}))\, d\mathbf{t_e}\, d\mathbf{t_e} \quad \text{Eq. 25}$$

4.3 SOLVING THE INTEGRATED FORMULA



In our model, introducing pinch points (From Equations 15, 19, 22, and 28) into the top level decision formula (Equation 4) for the distribution of engineering and operations behaviors gives us a mathematical "big picture":

$$EU(\mathbf{m}) = \int_{\mathbf{t_e}} \int_{\mathbf{t_g}} U(\mathbf{t_e}, \mathbf{t_g}, \mathbf{m}) * f(\mathbf{t_e}, \mathbf{t_g} \mid \mathbf{m})\, d\mathbf{t_e}\, d\mathbf{t_g} \quad \text{Eq. 26}$$

$$= EU(\mathbf{m_e}, \mathbf{m_v}, \mathbf{m_z}, \mathbf{m_w}, \mathbf{m_s}, \mathbf{m_u}, \mathbf{m_y}, \mathbf{m_l}, \mathbf{m_g})$$

$$= \int_{te}\int_{tg} U(\mathbf{t_e}, \mathbf{t_g}, \mathbf{m_e}, \mathbf{m_v}, \mathbf{m_z}, \mathbf{m_w}, \mathbf{m_s}, \mathbf{m_u}, \mathbf{m_y}, \mathbf{m_i}, \mathbf{m_l}, \mathbf{m_g}) *$$

$$\int_r (f(\mathbf{t_e}, \mathbf{r} \mid \mathbf{m_e}) * \int_d f(\mathbf{d} \mid \mathbf{r}, \mathbf{m_z}, \mathbf{m_v}) * \int_a f(\mathbf{a} \mid \mathbf{d}, \mathbf{m_w}, \mathbf{m_s}) * \qquad \text{Eq. 27}$$

$$\int_f f(\mathbf{f} \mid \mathbf{a}, \mathbf{m_y}, \mathbf{m_i}, \mathbf{m_l}) * f(\mathbf{t_g} \mid \mathbf{m_g}, \mathbf{t_f})\, d\mathbf{t_e}\, d\mathbf{t_g}\, d\mathbf{r}\, d\mathbf{d}\, d\mathbf{a}\, d\mathbf{f}$$

4.4 ILLUSTRATION

Introduction

This chapter illustrates our proposed method using a POPE model of a hypothetical mars shuttle flight. Figure 1 *(*on page 10*)* is a schematic of the objects and relationships for this case, and Tables 3.1.1 and 3.1.2 provide a set of sample data that reflect an analysis using one trial simulation. As we conclude each chapter below, we illustrate the introduced algorithm steps on this example problem.

Model Structure

We provide an intuitive formulation that does not reflect any actual aerospace project because our main purpose is to convey an intuition of the proposed method. We model each of the following factors *(*at nominal values unless stated otherwise*)*:

Product The operational product components are a mars shuttle, ground systems, and launch vehicle, with interfaces between each pair. There is also a potential for complex interactions involving all three systems. Failure created by any of these elements can cause a project failure, so we relate the product to a single string functional block diagram.

Process Tracing these components upstream, we see that the engineering process involves readying the mars shuttle, ground systems, and launch vehicle. These three tasks overlap significantly, and the work is interdependent among them. We model this in VDT using "start-to-start with lag" precedence links, and rework links between all pairs.

Organization Linking the process back to the organization, we identify the

engineering team responsible for each of these tasks, and find that a single manager oversees them. We model this in VDT using a two-tier hierarchy of three subteams and a project manager, operating with high centralization and formalization.

Every week, the groups assemble weekly for one-hour meeting to discuss possible interactions among the three components. We model this in VDT as a meeting object with all actors invited.

Environment NASA's own PRA estimates indicate that by far the largest source of operations phase risk is the .005% probability that a shuttle will be destroyed as a result of an impact from micrometeoroids or space junk [NASA 2003]. This is an example of a functional failure cause that designers and developers cannot affect, and it can be assessed independently of our knowledge of the project structure. We model this in PRA as a weather factor with a single-string influence over total failure probability.

For ease of reference, at this point we consolidate the input, intermediate, and output data for our sample illustration. At the conclusion of each chapter, we provide the data that we operate upon as well as an explanation of each step in the sample case.

Table 4.4.1: Consolidated Data for Illustrative Case

r This example evaluates only one distribution (one simulation run) of engineering stage actions

	Routine	Rework	QuickFix	Ignore	Default
Ready Shuttle	17	2	0	1	0
Ready Launch	16	3	0	1	0
Ready Ground	15	3	1	0	1
Shuttle-Launch Rework Link	13	1	0	1	2
Shuttle-Ground Rework Link	17	2	1	0	0
Launch-Ground Rework Link	16	3	1	0	0
System Integration Meetings	15	0	0	0	5

b

Subtasks
20
20
20
20
20
20
20

m_v The posterior probability of product conformance given different engginering stage actions

	Critical					Non-Critical				
	Routine	Rework	QuickFix	Ignore	Default	Routine	Rework	QuickFix	Ignore	Default
Ready Shuttle	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Ready Launch	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Ready Ground	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Shuttle-Launch Rework Link	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Shuttle-Ground Rework Link	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Launch-Ground Rework Link	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
System Integration Meetings	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%

d The conformance (lack of defects) of each engineered element

	Critical	Non-Critical
Shuttle Component	71%	71%
Launch Component	67%	67%
Ground Component	73%	70%
Shuttle-Launch Interface	61%	61%
Shuttle-Ground Interface	74%	73%
Launch-Ground Interface	74%	71%
Integrated Multi-System	75%	75%

m_z The coverage of each engineering stage task's testing activities on output from other tasks

	Critical						Non-Critical					
	Shuttle Component	mponent	mponent	Interface	nd Interface	ılti-System	omponent	omponent	omponent h	Interface	d Interface	Multi-System
Shuttle Component	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Launch Component	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Ground Component	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Shuttle-Launch Interface	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Shuttle-Ground Interface	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Launch-Ground Interface	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Integrated Multi-System	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

$d' = d$ Testing has no impact on conformance in this example

m_w The pace at which subsystems employ engineered elements during operations

	Critical					Non-Critical				
	Weather	Ground	Launch Ve	Shuttle	Science Go	Weather	Ground	Launch Ve	ice Shuttle	Science Goals
Shuttle Component	0%	0%	0%	100%	0%	0%	0%	0%	100%	0%
Launch Component	0%	0%	100%	0%	0%	0%	0%	100%	0%	0%
Ground Component	0%	100%	0%	0%	0%	0%	100%	0%	0%	0%
Shuttle-Launch Interface	0%	0%	50%	50%	0%	0%	0%	50%	50%	0%
Shuttle-Ground Interface	0%	50%	0%	50%	0%	0%	50%	0%	50%	0%
Launch-Ground Interface	0%	50%	50%	0%	0%	0%	50%	50%	0%	0%
Integrated Multi-System	0%	33%	33%	33%	0%	0%	33%	33%	33%	0%

m_s The sensitivity of subsystems to defective behavior by engineered elements during operations

	Critical					Non-Critical				
	Weather	Ground	Launch Ve	Shuttle	Science Go	Weather	Ground	Launch Ve	ice Shuttle	Science Goals
Shuttle Component	0%	0%	0%	100%	0%	0%	0%	0%	100%	0%
Launch Component	0%	0%	100%	0%	0%	0%	0%	100%	0%	0%
Ground Component	0%	100%	0%	0%	0%	0%	100%	0%	0%	0%
Shuttle-Launch Interface	0%	0%	50%	50%	0%	0%	0%	50%	50%	0%
Shuttle-Ground Interface	0%	50%	0%	50%	0%	0%	50%	0%	50%	0%
Launch-Ground Interface	0%	50%	50%	0%	0%	0%	50%	50%	0%	0%
Integrated Multi-System	0%	33%	33%	33%	0%	0%	33%	33%	33%	0%

b The rate at which engineering-induced anomalies manifest in operations

	Weather	Ground	Launch Ve	Shuttle	Science Goals
Critical	0.0%	8.6%	8.0%	8.2%	0.0%
Non-Critical	0.0%	8.4%	7.9%	8.1%	0.0%

a(2) The total number of anomalies that occur in each system by time 2 during operations (point estimate)

	Weather	Ground	Launch Ve	Shuttle	Science Goals
Critical	-	0.17	0.16	0.16	-
Non-Critical	-	0.17	0.16	0.16	-

m_i Influence that subsystem behavior has over the success or failure of functions

	Critical					Non-Critical				
	Weather	Ground	Launch Ve	Shuttle	Science Go	Weather	Ground	Launch Vehicle	Shuttle	Science Goals
Launch	100%	100%	100%	100%	0%	100%	100%	100%	100%	0%
Orbit	0%	100%	0%	100%	0%	0%	100%	0%	100%	0%
Land	100%	100%	0%	100%	0%	100%	100%	0%	100%	0%
Collect Science Data	0%	0%	0%	0%	0%	100%	0%	0%	100%	100%

$\sum_j (a_{gj}(2) * i_{gjk})$ The number of impacting errors on each system at time 2

	Critical	Non-Critical
Launch	0.49	0.49
Orbit	0.34	0.33
Land	0.34	0.33
Collect Science Data	-	0.16

m_y Decay in capacity for each function resulting from each operations anomaly

	Critical	Non-Critical
Launch	50%	50%
Orbit	50%	50%
Land	50%	50%
Collect Science Data	50%	50%

c(2) Operating Capacities at time 2 for each operations function

	Critical	Non-Critical
Launch	71%	71%
Orbit	79%	80%
Land	79%	80%
Collect Science Data	100%	89%

$P(F_{gk} < 2)$ (Independent) probabiliy of functional failure by time 2

	Critical	Non-Critical
Launch	29%	29%
Orbit	21%	20%
Land	21%	20%
Collect Science Data	0%	11%

$P(T_{gg} < 2)$ (Individual) probability of project goal failures

	Critical	Non-Criti	Probability
Failure	45%	40%	
Success	55%	60%	

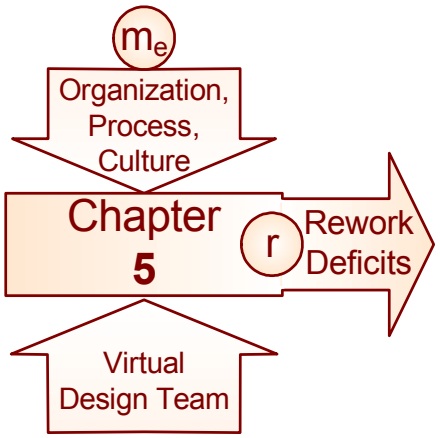
$P(T_g < 2)$ Joint probabiliy distribution project goal failures

Critical	Non-Critic	Probability
Success	Success	33%
Failure	Success	27%
Success	Failure	22%
Failure	Failure	18%

Chapter 5 - Engineering Rework Deficits

Estimating $f(\mathbf{t_e}, \mathbf{r} \mid \mathbf{m_e})$ Using the VDT Simulator

5.1 ENGINEERING REWORK-RELATED ACTIONS



The Virtual Design Team (VDT) is an object-oriented discrete event simulation that models a wide range of distinctive real-world phenomena. In mathematical terms, VDT uses a Generalized Semi-Markov Process (GSMP) to estimate a function that has no closed form representation. The output of this function is a joint distribution over a range of phenomena (including direct work, rework, waiting, and communications) that occur in continuous time.

We have defined $\mathbf{m_e}$ as a consistent set of management choices that VDT can evaluate, including an organizational hierarchy, task network, and operating culture. There are many diverse management choices that can influence these results, which are summarized in Pugnetti [1997] (in addition to previously cited VDT publications). The number of management strategies that we will evaluate using this basic algorithm will generally be small, however, because each one requires a manual VDT simulation and considerable post-processing. For example, one strategy might include a flat hierarchy, parallel tasks, and centralized culture. An alternative strategy could include a deep hierarchy, sequential tasks, and decentralization.

Table 5.1.1: Notation Introduced for Rework Deficits Analysis

Term	Name	Data	Description	Example
ξ	number of simulation iterations	Integer	Number of VDT simulation trials, each sampling different possible engineering phase actions based on management decisions	b=100 indicates that we will estimate engineering behavior based on 100 samples from the distribution of possible outcomes.
ψ_s	Random Number Seed	Sequence of Integers	Random number seeds by each VDT simulation trial run $s \in \{1, 2, \ldots \xi\}$ to sample a complex distribution of possible outcomes	If $s(1)$= 74, the first VDT simulation for a case will offer a particular sample of emergent behavior that differs, for example, from a second run with $s(1)$= 5641.
*Vdt(***m**, ψ_s*)*	*Virtual design team*	Function	Sample of possible engineering project behavior (A), including emergent schedules, rework, and meeting attendance, as predicted by Virtual Design Team simulator based on project design **m** using random number seed s.	A simulation *Vdt(*2, 34*)* could indicate that executing the second possible management plan would create a 140 day schedule overrun, with lots of ignored exceptions. *Vdt(*2, 432*)* might indicate more schedule overrun, while *Vdt(*2, 542*)* could indicate less.
*Vfp (***m**, ψ_s, i, t*)*	*Verification failure probability*	Function *(*returns a probability*)*	Probability that an actor will fail to verify the conformance of work (and raise an exception) based on task or rework link i, at time t, during an engineering simulation run of **m**, using seed ψ_s.	*Vfp (*M_1, 543, 2, 0.5*)* = .1 indicates that for simulating the first possible management plan one time using seed 543 indicates that after a half year, around one in ten subtasks from task 2 "Design Electrical System" creates an exception

Term	Name	Data	Description	Example
vfp^*	*Verification failure probability-nominal*	Probability	Expert-assessed percent of verifications that **perceive** non-conformance, for an average actor addressing an average task	$vfp^* = 0.1$ means that under nominal conditions, ten percent of all completed subtasks generate exceptions.
AP(k_i, p_i, t_i*)*	*Aptitude*	Function	Aptitude for an assigned actor's skill s_{mi} and experience e_{mi} in addressing the task i of complexity c_{mi} under management choices **m**, expressed as a real number using an industry – calibrated lookup table in VDT	*AP(*"High", "High", "Low"*)* = 0.67 means that an actor with high skill and experience generates only two thirds of the normal number of exceptions when working on a low complexity task *(*assuming no beneficial or detrimental events*)*.
k_i	skill	Real	Relevant skill of the actor assigned to task i under management choices **m**, expressed as High, Medium, or Low by experts and defined using industry – calibrated values in VDT	s_{23} = "High" indicates that in the second management plan being considered, the third actor "Electrical Engineering Team" is exceptionally skilled *(*as defined by VDT*)*.
p_i	experience	Real	Experience of the actor assigned to task i under management choices **m**, expressed as High, Medium, or Low by experts and defined using industry – calibrated values in VDT	c_{12} = "Medium" indicates that in the first management plan being considered, the second actor "Mechanical Engineering Team" has a nominal level of experience *(*as defined by VDT*)*.
t_i	complexity	Real	Complexity of task i under management choices **m**, expressed as High, Medium, or Low by experts and defined using industry – calibrated values in VDT	c_{12} = "Medium" indicates that in the first management plan being considered, the second task "Mechanical Engineering Design" involves a nominal level of difficulty.

Term	Name	Data	Description	Example
$V_+(\mathbf{m}, s, i, t)$	*Verification positive events*	Function	Number of beneficial events, such as successful communications, that occur before the time t of task i verification, during a VDT simulation of management choices **m**, with seed s. These decrease vfp.	$V_+(M_4, 3, 2, 1) = 6$ if after simulating management plan 4 for 1 time period using random seed 3, engineering task 2 has had 6 positive events occur, such as three completed communication attempts and two fully attended meetings.
$V_-(\mathbf{m}, s, i, t)$	*Verification negative events*	Function	Number of detrimental events, such as failed communications, that occur before the time of task I verification t, during a VDT simulation of management choices **m**, with seed s. These increase vfp.	$V_-(M_4, 3, 2, 1) = 5$ if after simulating management plan 4 for 1 time period using random seed 3, engineering task 2 has had 5 negative events occur, such as three failed communication attempts and two ignored exceptions.
γ	verification change	Real	VDT calibration constant that indicates the amount of impact that each positive or negative event has upon Vfp	$v_\Delta = 0.1$ means that each time an exception is ignored, the probability of generating exceptions for that task goes up by ten percent.
$X(\mathbf{m}, s, i)$	*eXceptions*	Function	The total number of exceptions for a simulation of management choices **m** using seed s on task i	$X(M_4, 3, 2) = 5$ if when we simulate the engineering project of management plan 4 using random seed 3, task 2 generates a total of 5 exceptions.
Bernoulli(x)	*Bernoulli*	Function (returns a Boolean random variable)	Produces a random variable that takes on value 1 with probability x, and value 0 otherwise	*Bernoulli*(0.5) models a fair coin toss, where a result of 1 indicates heads and 0 indicates tails.
n_i	number of subtasks	Integer	Number of subtasks, or atomic work items, that constitute task i.	$b_4 = 20$ indicates that the fourth task, "Electrical – Mechanical Interface Design" is broken into twenty work items.

Term	Name	Data	Description	Example
*av(***m**, s, i, t, g*)*	*availability*	Function	Availability of an actor at hierarchy level g to attend to exceptions, in a reporting line for the actor responsible for task i, during a simulation of plan **m**, with seed s(s), at time t. Output is ∈ {0, 1} where 1 represents availability, 0 represents overloading.	*av(*M_5, 4, 3, 2, 1*)* = 0 indicates that when we simulate management plan 5 using random seed 4, at time 2 the subteam leader *(*level 1*)* who the actor assigned to task 3 reports to is too busy to make decisions about whether or not to perform rework. Exceptions that are routed to this subteam leader will be defaulted *(*creating no rework*)*.
*Av(***m**, s, i, t*)*	*Availability*	Function	Availability of actors to attend to exceptions, in a reporting line for the actor responsible for task i, during a simulation of plan **m**, with seed s(s), at time t. Vector has with cell values av(r, i, t, g) ∈ {0, 1} where 1 represents availability, 0 represents overloading. g is a level in the hierarchy (Project Manager, Subteam Leader, or Subteam).	*Av(*M_4, 3, 2, 1*)* = *(*1,1,1*)* indicates that when we simulate management plan 4 using random seed 3, at time 1 all of the decision makers who supervise the actor assigned to task 2 are available and will decide on whether to rework, quick-fix, or ignore any exceptions that occur.
Γ	Centralization	Vector	Project-specific values that define the project's centralization, a distribution that determines the probability of routing an exception to each of the different organizational levels	Γ = *(*.20, .40, .40*)* indicates that when an exception occurs under management plan 3, a subteam will make its own decision of whether or not to perform rework one-fifth of the time, while routing to a supervising subteam leader or project manager are each twice as likely.
Λ	Quality focus	Matrix	Industry-specific calibration constants indicating quality focus, specifically the tendency of actors at different organizational levels to rework exceptions (to keep project failure risk low) or ignore them (to minimize project duration and cost)	A matrix with equal values *(*.333*)* in each cell means that subteams, subteam leaders, and project managers make rework decisions the same way- are distribute their decisions evenly among rework, quick-fix, and ignore exception handlings.

5.2 MATHEMATICAL BASIS OF SIMULATION

In this section, we explain mathematically how VDT derives a joint distribution on actions of interest to us, based on a set of management choices. These actions of interest are the exception handling behavior for each task and rework link, and meeting attendance.

Each **r** is a set of engineering actions, such as schedule slippage, communications failures, and rework that can result from the management choices $\mathbf{m_e}$. Many possible results can occur for a given set of management choices m. For example, R_1 might be a case in which schedule and communications are kept, but a lot of rework gets ignored. R_2 might show mild schedule slippage, but shortcomings in both communications and in the amount of rework. The amount of variance between simulations of a particular set of management choices depends on those management choices, and is not straightforward to estimate a priori.

The remainder of this subsection reviews the mathematical basis of simulation so that our definition of the probabilities is complete. VDT does not calculate a continuous probability distribution on the possible actions that will occur during engineering. Instead, as a simulation it estimates that distribution by generating a set of n random samples that together approach the full distribution asymptotically [Law and Kelton, 2000]. The simulator uses random numbers to sample different possible results from distributions of possible engineering micro-behaviors, thereby producing a set of aggregate outcomes that are presumed equally likely to reflect real emergent engineering behavior.

Although the set of A possibilities is very large *(*meaning that it has many continuous dimensions*)*, our algorithm only requires that we evaluate it at a limited set of points. Thus, we estimate that each set of actions A occurs with a probability equal to the long run fraction of simulations that predict those actions. Formally, our formula takes the limit on ξ simulations with random number seeds ψ_s used to generate independent, identically distributed results:

$$P(\mathbf{t_e}, \mathbf{r} \mid \mathbf{m_e}) = \lim_{n \to \infty} |\{ \psi_s \mid Vdt(\mathbf{m_e}, \psi_s) = (\mathbf{t_e}, \mathbf{r}) , s \in \{ 1, 2, \ldots \xi \}\}| / \xi \qquad \text{Eq. 1}$$

With enough trials, we can achieve an arbitrary degree of confidence that the joint distribution of sampled simulator outcomes characterizes the modeled theory. For a given degree of desired precision, we can also identify the required number of samples n [Law and Kelton, 2000]. Although we do not assume this in the remaining algorithm, it is worth noting that in most cases, we can estimate the distribution of actions well enough to support a decision with a small number of simulations, on the order of 100 to 1000. In this case, the probability of sampling the same result twice is negligible, and we are likely to have n equally probable possibilities to evaluate:

$$P(\mathbf{t_e}, \mathbf{r} \mid \mathbf{m_e}) = 1/\xi \text{ when } (\mathbf{t_e}, \mathbf{r}) = Vdt\,(\mathbf{m_e}, \psi_s) \text{ for } s \in \{1, 2, \ldots \xi\}$$
$$0 \text{ otherwise} \qquad \text{Eq. 2}$$

5.3 IMPORTANCE OF EXCEPTION HANDLING BEHAVIOR

Our analysis focuses on emergent exception handling and meeting attendance behavior because when engineering causes an operations failure, we can generally point to work that should have been done over. Often, but not always, engineering displayed some warning signs, such as the escalation of engineering concerns that were subsequently ignored.

The VDT mechanics support this model by deriging exception-handling behavior from a wide range of factors that have been linked to downstream operations failure risk. In VDT, the number of exceptions is a function of:

- Actor skill and experience
- Task complexity
- Coordination and rework completion rates
- Industry- and project- specific calibration constants

In addition, the exception handling of these exceptions is a function of

- Tendency to perform necessary rework at different organizational levels (determined by individual program vs. project priorities, risk attitude, and corner-cutting)
- Backlog and overloading of decision-makers (latency)
- Centralization

These factors each play a direct role in the VDT simulator's predictions of emergent phenomena that we claim indicate *rework deficit*— an amount of rework that may be necessary, but that the organization has not conducted. This simple concept of *rework deficit* suggests that when rework is necessary, but not performed, risk increases. An important part of our model is to provide a precise estimate of the amount of increase in risk that occurs for a given change in rework deficit.

5.4 VERIFICATION FAILURE PROBABILITY

Although we have no closed form equation to define the actions that VDT predicts, by careful evaluation of VDT mechanics we can shed some light on its direct contributors. The first value of critical importance is the *Verification failure probability*, known in VDT parlance at *Vfp*. *Vfp* is the probability that an actor perceiges just-completed work as not conforming to specifications, and will raise an exception. An exception is a warning sign, suggesting that the work may not meet all the appropriate requirements – the organization may either ignore or respond to an exception in various ways.

We formally define $Vfp(\mathbf{m_e}, \psi_s, i, t)$ as a function of task or rework link i and time t during a simulation of management choices **m** using random seed s. Each task or rework link begins with an initial value $Vfp(\mathbf{m_e}, \psi_s, i, 0)$ based on two industry-specific calibration constants. The first is vfp^*, the nominal fraction of verifications that fail for an average actor addressing an average task. The second parameter is the degree of aptitude $AP(k_i, p_i, t_i)$ of an assigned actor's skill k_i and experience p_i in addressing task i of complexity t_i.

$$VfP(\mathbf{m_e}, \psi_s, i, 0) = vfp^* * AP(k_i, p_i, t_i) \quad \text{Eq. 3}$$

During each simulation run, various events occur that alter the likelihood of an actor raising an exception. Of particular importance, these events include the success or failure of required coordination activities and rework. Where $V_+(\mathbf{m}, \psi_s, i, t)$ is the number of events that occur before the time of verification t during simulation r, that increase *Vfp*, and $V_-(\mathbf{m_e}, \psi_s, i, t)$ is the number that decrease it, we have

$$VfP(\mathbf{m_e}, \psi_s, i, t) = VfP(\mathbf{m_e}, \psi_s, i, 0) * (1+\gamma)^{V_+(\boldsymbol{m}_e, \psi_s, i, t) - V_-(\boldsymbol{m}_e, \psi_s, i, t)} \quad \text{Eq. 4}$$

Here v_Δ is a VDT calibration constant that indicates the amount of impact that each event has upon *vfp*. The total formula for *vfp* is thus

$$VfP(\mathbf{m_e}, \psi_s, i, t) = vfp^* * AP(s_{im}, e_{im}, c_{im}) * (1+\gamma)^{V_+(\boldsymbol{m}_e, \psi_s, i, t) - V_-(\boldsymbol{m}_e, \psi_s, i, t)} \quad \text{Eq. 5}$$

5.5 TOTAL NUMBER OF EXCEPTIONS

We define $X(\mathbf{me}, \psi_s, i)$ as the total number of exceptions for a simulation of management choices **me** using seed ψ_s on task i. If they were independent, these values would simply be based on $VfP(\mathbf{me}, \psi_s, i, t)$ at each of the times $V(\mathbf{me}, \psi_s, i, b)$ at which each subtask b is completed (because verification occurs immediately and automatically). Formally, **assuming independence**, we have

$$P(X(\mathbf{m_e}, \psi_s, i) = y) \stackrel{?}{=} \sum_s Bernoulli(VfP(\mathbf{m_e}, \psi_s, i, V(\mathbf{m_e}, \psi_s, i, b))) \quad \text{Eq. 6}$$

$$Mean(X(\mathbf{m_e}, \psi_s, i)) \stackrel{?}{=} \sum_s VfP(\mathbf{m_e}, \psi_s, i, V(\mathbf{m_e}, \psi_s, i, b)) \quad \text{Eq. 7}$$

These equations convey an intuition for simulated project behavior, but they are **not accurate** because the distributions of verification times, exceptions and vfp values are highly interdependent. For example, raising too many exceptions for an organization to handle causes vfp to increase dramatically, causing a downward performance spiral. Similarly, staying on top of communications and conducting necessary rework at a high rate will decrease vfp and the number of exceptions over time. The combination of uncertainty, discontinuity and sensitivity in organizational behavior is one reason why VDT, as opposed to pure mathematics, is an appropriate tool for predicting project behavior.

5.6 EXCEPTION HANDLING OF EXCEPTIONS

We use the term *exception handling* to describe the result of attempting to verify a subtask (portion of an engineering task or rework link) and handling any resulting exception. VDT models five possible exception handlings, and for subscripting purposes, we index them using integers:

1. **Verified –** Upon completion of the subtask, the actor successfully verified the created product's conformance to specification.

2. **Reworked –** Conformance was not originally verified, however the task was later redone to fix the perceived shortcoming.
3. **Quick-Fixed –** The subtask was not originally verified, but a portion of the subtask was redone in an effort to mitigate the risk of an anomaly
4. **Ignored –** The subtask was not originally verified, however a decision was made not to redo an of the work involved *(*for example, because there was no time for rework, the risk was perceived as low, and/or verification of conformance was made by another actor*)*
5. **Default –** The subtask was not originally verified, and a decision of whether or not to perform rework was not made due to overloaded organization. The subtask was not reworked.

Each subtask that is successfully *verified* requires no further action, so we can define the number of verified subtasks r_{i1} for a task or rework link i with b_i subtasks as follows:

$$r_{i1} = b_i - X(\mathbf{m_e}, \psi_s, i) \quad \text{Eq. 8}$$

When verification fails, and an exception is raised, VDT creates a new *decision*—a work item that is routed to an actor at or above the working actor's level in the hierarchy, depending on the project's level of centralization.

Overloaded actors sometimes fail to address decision items, resulting in a *defaulted* exception handling that causes no rework. No simple mathematical expression tells us when an organizational level g is available to address a subordinate's new exception. However we can describe this availability as a vector $Av(\mathbf{m_e}, \psi_s, i, t)$ with cell values $av(\mathbf{m_e}, \psi_s, i, t, g) \in \{0, 1\}$ where 1 represents availability. Γ is a project-specific vector that defines the engineering organization's centralization, a distribution that defines the probability of routing an exception to each of the different organizational levels. Using this nomenclature, we can estimate r_{i2}, the expected number of subtasks that result in exceptions about which no rework decision is made.

$$a_{i5} = X(\mathbf{m_e}, \psi_s, i) * (1 - \Gamma * Av(\mathbf{m_e}, \psi_s, i, t)) \quad \text{Eq. 9}$$

These results are called delegations by default, and they result in no rework being performed.

When the actor who receives a decision item attends to it, VDT simulates a decision based on the actor's role. This decision determines the subtask's exception handling, whether to *rework*, *quick-fix*, or *ignore* the exception. Where Λ is a matrix of industry-specific calibration constants indicating the tendency of actors at different organizational levels to rework or ignore exceptions, the mean number of subtasks resulting in each exception handling is:

$$(r_{i2}, r_{i3}, r_{i4}) = X(\mathbf{m_e}, \psi_s, i) * \Gamma * Av(\mathbf{m_e}, \psi_s, i, t) * \Lambda \quad \text{Eq. 10}$$

Again, these formulae for expected numbers are only approximations that we use for illustrative purposes.

Our algorithm uses values from the matrix **r** that come from the VDT simulation. Using these values, we form a matrix of actions of interest that VDT predicts from management choices $\mathbf{m_e}$, given random number seed ψ_s. Recall that we estimate the probability of **r** actions actually occurring, given management choices $\mathbf{m_e}$, as

$$P(\mathbf{t_e}, \mathbf{r}) = P((\mathbf{t_e}, \mathbf{r}) = Vdt(\mathbf{m_e}, s)) \quad \text{Eq. 11}$$

$$P(\mathbf{t_e}, \mathbf{r}) = |\{ \psi_s \mid (\mathbf{t_e}, \mathbf{r}) = Vdt(\mathbf{m_e}, \psi_s), i \in \{1, 2, \ldots n\}\} / n \quad \text{Eq. 12}$$

Where the cell values of **r** are r_{hi}, the number of task i's subtasks that result in exception handling h, as predicted for management choice $\mathbf{m_e}$ based on random seed ψ_s.

5.7 ILLUSTRATION

Calculations Using Hypothetical Data

In this section, we illustrate each step of the proposed method using hypothetical data.

Table 5.7.1: Sample Data on Rework Deficits

r This example evaluates only one distribution (one simulation run) of engineering stage actions

	Routine	Rework	QuickFix	Ignore	Default
Ready Shuttle	17	2	0	1	0
Ready Launch	16	3	0	1	0
Ready Ground	15	3	1	0	1
Shuttle-Launch Rework Link	13	1	0	1	2
Shuttle-Ground Rework Link	17	2	1	0	0
Launch-Ground Rework Link	16	3	1	0	0
System Integration Meetings	15	0	0	0	5

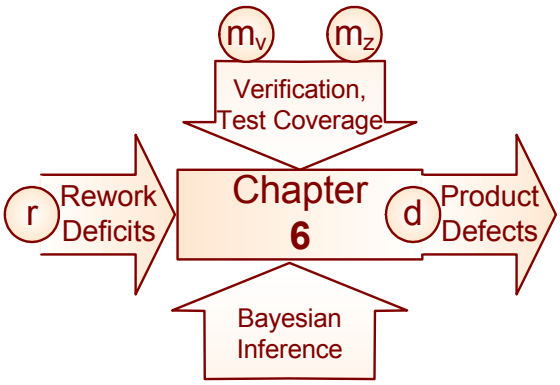
b

Subtasks
20
20
20
20
20
20
20

Chapter 6 - Engineered Product Defects

Estimating f(***d*** | ***r***, $\boldsymbol{m_z}$, $\boldsymbol{m_v}$) *Using Bayesian Inference*

6.1 Engineered Product Elements' Conformance to Specifications



As explained above, the engineering specification lays out a set of requirements that an engineered product must meet. The actions that we observe or predict for an engineering phase are intended to create a product that *conforms* to the given specification. **r** product that conforms perfectly will satisfy all of the specification's requirements when it is operated. In most cases, human limitations prevent the creation of a product that meets all of the requirements under all the specified circumstances.

In this section, we estimate the fraction of each engineered element that does indeed conform to the specifications. For each task, we develop a measure of conformance that ranges from one *(*no significant flaws*)* to zero *(*no requirements are perfectly satisfied*)*.

Because VDT does not have a product model, it does not tell use when a product is likely to conform. Instead, we must use Bayesian inference to estimate the degree of conformance given specific exception handling behaviors. Intuitively, the best task processing behavior that VDT can forecast consists of each subtask being completed routinely, and in this case, we have the highest achievable degree of confidence in the element's conformance. In the other extreme case, each subtask produces an exception,

but the organization is too overloaded to respond to *any* of them. This case is quite unlikely, since in this case the simulation would almost certainly show a complete program failure. Nevertheless, it defines the other end of our scale. Note that in this case, it is very unlikely but still possible that the engineered element conforms to specification. The next section explains our exact mathematical method.

Table 6.1.1: Notation Introduced for Product Defects Analysis

Term	Name	Data	Description	Example
S	Signal	Event	*A given design element* conforms to specification- it goes as far as possible toward project success within the management plan	A given electrical circuit that conforms will not suffer a design error in operations
~~S~~	No Signal	Event	*A given design element* does not conform to specification—it is capable of creating anomalies during operations by violating requirements that are made of it.	A given that does not conform can crash a software system during operations
"S"	signal Perceived	Event	An engineer who has just completed part of *a given design element* believes that he or she has done the work correctly (No exception occurs)	A given programmer who reviews his work may perceive that it fulfills the requirements, and so will not require management attention.
"~~S~~"	signal not Perceived	Event	An actor reviewing a portion of *a given design element* will perceive that it does not conform, and will raise an exception.	A given electrical engineer may feel that the requirements are probably not fully satisfied for the just-completed work item, and will request a decision on whether to do rework.
P("S"|S*)*	Test Sensitivity	Probability	Test Sensitivity: The expert-assessed probability that an engineer will correctly verify *a given design element*, creating no exception	*P(*"C"|C*)* = 1 indicates for example that a given mechanical engineer will always be able to verify that a conforming piece of work is correct.
P("~~S~~"|~~S~~*)*	Test Specificity	Probability	Test Specificity: The expert-assessed probability that an engineer will correctly fail to verify *a given design element*, creating an exception	*P(*"~~C~~"|C*)* = 0.1 indicates for example that ten percent of the time, a given mechanical engineer will believe that a correct component has problems.

Term	Name	Data	Description	Example
$P($"S"$_o)$	Nominal signal perception probability	Probability	Expert-assessed fraction of verifications that succeed under nominal variable settings	$P($"C"$_o)$ = 0.9 indicates for example that nine out of ten times, an average mechanical engineer on an average task will believe that his or her work was right the first time.
$P(S_o)$	Nominal signal probability	Probability	Calculated fraction of subelements that conform under nominal (initial) conditions	$P(C_o)$ = 0.95 indicates for example that nineteen out of twenty times, an average electrical engineer on an average task will produce work that is correct on the first try.
$P($S|"~~S~~"$)$	Probability of Signal given not perceived	Probability	Calculated probability that an unverified portion of *a given design element* conforms to specifications	$P($C|"~~C~~"$)$ = 0.1 indicates that ten percent of the time, work which a given engineer thinks is flawed will actually be correct.
$P($S|"S"$)$	Probability of Signal given perceived	Probability	Calculated probability of conformance for *a given design element* given verification	$P($C|" C"$)$ = 0.95 indicates that ninety five percent of the time, work which a given engineer thinks is correct will actually be correct.
*bin(*a, b*)*	Binomial Distribution	Function returning a distribution	Evaluates to a probability distribution based on the sum of a Bernoulli trials that each has probability of success b.	*bin (*2, 0.5*)* = 0.25. For example, this is the probability of getting heads twice on two coin flips.
d'_{gi}	Product defects *after* testing tasks	Real	Engineered element i's conformance level for goal g requirements—after any changes due to testing. Derived from d_{gi} and z_{gin}.	d'_{14} = 0.9 indicates that after testing, the product of engineering task 4 satisfies nine tenths of the relevant goal 1 *(*critical*)* requirements. If d_{14} = 0.8, this would indicate that before testing the fraction was just eight tenths, so testing improved conformance by one tenth.

6.2 PRIOR CONFORMANCE PROBABILITY AND VERIFICATION QUALITY

Our interpretation of direct engineering work in VDT follows guidelines presented in the SAM framework's "Execution Model" of action [Murphy and Paté-Cornell 1996, pp. 508-509]. VDT divides each work task into atomic portions of activity, known as *subtasks*, that we consider to result in an atomic portion of their engineered output, called a *subproduct*. The *prior conformance probability* for a subproduct is the chance that the requirements underlying the corresponding subtask were met after an actor's first attempt.

S = A given design element conforms to specification

~~S~~ = A given design element does not conform to specification

In this section, we use abbreviated variable forms because they most clearly describe the verification process for a given subtask.

After completing each subtask, actors assess their work's degree of conformance, or achievement of the specification's requirements, and either verify its correctness or raise an exception:

"S" = A given design element's conformance verification succeeds *(*No exception*)*

"~~S~~" = A given design element's conformance verification fails *(*Exception*)*

Our interpretation of exception handling in VDT follows guidelines presented in the SAM framework's "Rule-based Model" of action [Murphy and Paté-Cornell 1996, pp. 507-508]. In this model, where S represents a signal, "S" represents an actor's perception that the signal is there. Verification is not a perfect process, so there is a chance that a verified subproduct does not meet all requirements, and there is a chance that an unverified subproduct actually conforms. In order to capture this phenomenon precisely, we can ask domain experts to assess the following commonly understood measures:

P("S" | S*)* = Sensitivity: The probability that an engineer will correctly verify a given portion of a task or rework link's conforming product, creating no exception

P("~~S~~" | ~~S~~*)* = Specificity: The probability that an engineer will correctly fail to verify a given portion of a task or rework link's non-conforming product, creating an exception

VDT does not require these values because the simulation has no product model. We believe that these values can be derived through expert assessment, guided by the assessment of design review and testing results.

During expert assessment, we should verify that:

$$P(\text{"S"} \mid \text{S}) < P(\text{"S"}) < P(\text{"S"} \mid \sim\text{S}) \quad \text{Eq. 1}$$

This guarantees that the verification provides some indication of the underlying conformance, and some of the following calculations **assume** this.

The next step is to provide a formula for the probability of conformance.

$$P(\text{"S"}) = P(\text{S}, \text{"S"}) + P(\sim\text{S}, \text{"S"}) \quad \text{Eq. 2}$$

$$P(\text{"S"}) = P(\text{S})P(\text{"S"} \mid \text{S}) + P(\sim\text{S})P(\text{"S"} \mid \sim\text{S}) \quad \text{Eq. 3}$$

$$P(\text{"S"}) = P(\text{S})P(\text{"S"} \mid \text{S}) + (1-P(\text{S}))P(\text{"S"} \mid \sim\text{S}) \quad \text{Eq. 4}$$

$$P(\text{"S"}) = P(\text{S})P(\text{"S"} \mid \text{S}) + P(\text{"S"} \mid \sim\text{S}) - P(\text{S})P(\text{"S"} \mid \sim\text{S}) \quad \text{Eq. 5}$$

$$P(\text{"S"}) - P(\text{"S"} \mid \sim\text{S}) = P(\text{S})P(\text{"S"} \mid \text{S}) - P(\text{S})P(\text{"S"} \mid \sim\text{S}) \quad \text{Eq. 6}$$

$$P(\text{"S"}) - P(\text{"S"} \mid \sim\text{S}) = P(\text{S})(P(\text{"S"} \mid \text{S}) - P(\text{"S"} \mid \sim\text{S})) \quad \text{Eq. 7}$$

$$(P(\text{"S"}) - P(\text{"S"} \mid \sim\text{S})) / (P(\text{"S"} \mid \text{S}) - (P(\text{"S"} \mid \sim\text{S})) = P(\text{S}) \quad \text{Eq. 8}$$

In English, the probability of conformance is a function of the probability of verification. More specifically, the nominal probability of conformance equals the ratio between: verification probability minus false negative rate; and the sensitivity minus false negative rate.

VDT input also requires, currently as a stage-wide variable assessed by experts,

P("S"$_o$*)* = Fraction of verifications that succeed under nominal conditions *(*medium skill, experience, task complexity, with no complicating or simplifying events*)*

During expert assessment, we should substitute nominal conditions *(*such as "S"$_o$*)* into Eq. 24 and verify the reasonableness of our predicted fraction of subelements that conform under nominal *(*initial*)* conditions:

$$P(\text{S}_o) = (P(\text{“S”}_o) - P(\text{“S”}|\text{~~S~~})) / (P(\text{“S”}|\text{S}) - (P(\text{“S”}|\text{~~S~~})) \quad \text{Eq. 9}$$

If experts do not feel this value is a reasonable nominal value, we can iterate among these variables (nominal verification, nominal conformance, test Sensitivity, and specificity) to reach a consistent set of assumptions.

6.3 VERIFICATION-POSTERIOR SUBTASK CONFORMANCE PROBABILITY

In this section, we predict the chances of a subtask's conformance, once we know whether an exception was raised or not. These are the probability $P(\text{S}|\text{“S”})$ in the case that the work was verified and deemed correct by the completing actor, and the corresponding probability $P(\text{S}|\text{“~~S~~”})$ that a subtask satisfies the requirements in the specification given that there was an exception.

Our first goal is to calculate $P(\text{S}|\text{“~~S~~”})$, the probability that an unverified portion of a product element conforms to specifications. Applying Bayes' Rule, we have

$$P(\text{“~~S~~”},\text{S}) = P(\text{“~~S~~”}|\text{S})P(\text{S}) = P(\text{S}|\text{“~~S~~”})P(\text{“~~S~~”}) \quad \text{Eq. 10}$$

$$P(\text{S}|\text{“~~S~~”}) = P(\text{“~~S~~”}|\text{S})P(\text{S}) / P(\text{“~~S~~”}) \quad \text{Eq. 11}$$

If we **assume** that $P(\text{S}|\text{“~~S~~”})$ is fixed for a given task, we can calculate its exact value at the initial conditions:

$$P(\text{S}|\text{“~~S~~”}) = P(\text{“~~S~~”}_o|\text{S}_o)P(\text{S}_o) / P(\text{“~~S~~”}_o) \quad \text{Eq. 12}$$

$$P(\text{S}|\text{“~~S~~”}) = (1-P(\text{“S”}_o|\text{S}_o))P(\text{S}_o) / P(\text{“~~S~~”}_o) \quad \text{Eq. 13}$$

That is, the probability of conformance given verification failure equals the chance of a false negative, times the ratio of initial conformances to initial verification failure probability.

Similarly,

$$P(\text{“S”},\text{S}) = P(\text{“S”}|\text{S})P(\text{S}) = P(\text{S}|\text{“S”})P(\text{“S”}) \quad \text{Eq. 14}$$

$$P(\text{S}|\text{“S”}) = P(\text{“S”}|\text{S})P(\text{S}) / P(\text{“S”}) \quad \text{Eq. 15}$$

If we **assume** $P(\text{S}|\text{“S”})$ is fixed for a given task, we can calculate its exact value at the initial conditions to get:

$$P(\text{S}|\text{“S”}) = P(\text{“S”}_o|\text{S}_o)P(\text{S}_o) / P(\text{“S”}_o) \quad \text{Eq. 16}$$

Thus, the probability of conformance given verification equals the test sensitivity times the ratio of initial conformance to initial verification success probability.

6.4 EXCEPTION HANDLING-POSTERIOR SUBTASK CONFORMANCE PROBABILITY

We model two levels of goal among requirements. The first are critical requirements, and failing to meet them can lead to critical anomalies that cause a failure to achieve primary objectives. The second level of goal includes non-critical requirements, which can only lead to a partial failure *(*failure to achieve secondary objectives*)* during operations. The algorithm we present **assumes** that critical and non-critical requirements, anomalies, and failure do not interact, and so they are independent *given* a set of predicted engineering actions. That is, shortcomings in a particular engineering task are likely to cause both critical and non-critical failures, but the two do not directly influence one another.

After verification fails and an exception is handled, our estimate of the final conformance of a subtask's product can change. We conceptualize rework decisions to **assume** that they do not involve an additional verification step, but rather a decision of whether to redevelop the product to meet both critical and non-critical requirements *(*rework*)*, critical requirements only *(*quick-fix*)*, or neither *(*ignore*)*. Using this model, products will tend to conform more to critical than to non-critical requirements.

We can use the verification-posterior conformance probabilities to construct a matrix indicating our degree of belief in the ability of the exception handling process to ensure the conformance of a product. This takes the form of expert-assessed values v_{ghi} that contain the probability that a part of task i's product conforms to requirements for goal g, given that the organization produced coordination handling h when processing that subtask. Table 3 shows a matrix of formulae that we use to assign v_{ghi} based on calculations of $P(\text{S}|\text{“}\cancel{\text{S}}\text{”})$ and $P(\text{S}|\text{“S”})$ for a given task i.

Table 6.4.1: Estimating Product Conformance from Subtask Exception handling

v_{ghi}	**Rework Handling** $(_h)$				
Goal $(_g)$	Verified	Reworked	Quick-Fixed	Ignored	Defaulted
Critical	$P($S|"S"$)$	$P($S|"S"$)$	$P($S|"~~S~~"$)$	$P($S|"~~S~~"$)$	$P($S|"~~S~~"$)$
Non-Critical	$P($S|"S"$)$	$P($S|"S"$)$	$P($S|"S"$)$	$P($S|"~~S~~"$)$	$P($S|"~~S~~"$)$

For example, the table indicates that for all I we have

$$v_{13i} = P(\text{S}|\text{“}\sim\text{S”}) \quad \text{Eq. 17}$$

$$v_{23i} = P(\text{S}|\text{“S”}) \quad \text{Eq. 18}$$

This indicates that a quick-fixed subtask from task i has a probability of meeting critical requirements that is equal to the probability of conformance, given verification success. The probability of meeting secondary (non-critical) requirements equals the probability of conformance given verification failure. This operationalizes our assumption that quick-fixes address critical requirements only.

Thus, we **assume** that rework decisions are based on no insight into conformance beyond that gained during the initial verification. We can however extend our method to develop sophisticated models of decision-making, as illustrated in Figure 1. To solve for v_{ghi} in the general case we use Bayesian inference to flip the event tree in Figure 11. These more sophisticated models can take into account some controllable organizational features that are of interest for risk mitigation, such as technical insight (a "second opinion"), risk aversion, or corner-cutting [Garber]. This requires simply building an event tree based on the consequences of conformance and verification, and then flipping the tree to discern the posterior conformance for a given set of manifest behaviors.

6.5 MEETING ATTENDANCE-POSTERIOR MULTI-SYSTEM CONFORMANCE PROBABILITY

This section provides a method to calculate the degree of engineering conformance to specifications that regard the interactions among interdependent systems, independently of whether each component and interface operates properly in isolation. We **assume** that

meetings are the engineering stage opportunities for a group as a whole to collaboratively identify these problems.

As we have with tasks, we **assume** that interactions may occur among any subset of represented systems, that all possible dependencies are equally probable, and that meeting attendance provides all of the information from the design phase results A that influences the multi-system function's failure probability.

We can estimate the conformance of the multi-systems that correspond to meetings by assigning r_{1i} *(*corresponding to verification*)* to be the number of attendees, and r_{5i} *(*corresponding to default*)* to be the number of absences, summed over all meetings. In our simplified model *(*developed for a large number of equivalent and independent subtasks*)*, this model will **assume** linear interpolation between the best and worst cases. In the best case, when all meetings have perfect attendance, we can expect the greatest possible conformance in the system-of-systems, equivalent to verification of all subtasks. The worst case is no attendance at any meeting, which is equivalent to a delegation by default in all subtasks.

We can calculate the meeting attendance-posterior conformance rates, v_{g1i} and v_{g5i}, in a parallel fashion as we do for tasks, by soliciting the sensitivity and specificity of attendance as an indicator of multi-system conformance. Alternatively, we may be able to **assume** the v_{g1i} equals the average over the tasks i to which each invited actor is assigned.

6.6 PRODUCT CONFORMANCE RATES

In this section, we calculate a set **D** of random variables that describe the defects in different engineering products. In the case where we are able to make some reasonable assumptions, we can use cell values that represent the rates at which anomalies will manifest when an engineered component is operated. In a detailed model, cell values contain the future frequency of rates at which anomalies will manifest when an engineered component is operated.

We define the conformance D_{gi} of an engineered element i as a random variable that represents the probability of a randomly selected subtask conforming to the goal g requirements in the specification.

Recall that to calculate the probability densities $f(d_{gi}) = P(D_{gi} = d_{gi})$ we are given r_{hi} (sampled by VDT from random variable R_{hi}) equal to the number of task i's subtasks that result in exception handling h.

In the previous section, we also assessed $\mathbf{m_v}$, the set of probabilities v_{ghi} that an individual subtask from task i conforms to goal g requirements, given exception handling behavior h.

In many cases, we can **assume** each subtask to either meet all requirements or none, that these trials are independent of one another, and that requirements are equally probable to be sampled from among each of the b_i subtasks. In this case, we can model the conformance of each subtask using a Bernoulli trial. For the total distribution, we have a sum of five Binomial Distributions, each using a number of trials r_{hi} and with a probability parameter from v_{ghi}. This produces the formula that sums over all handling options $h \in \{1,2,3,4,5\}$ (representing that the subtask was verified, reworked, quick-fixed, ignored, default results):

$$D_{gi} \sim \sum_h (1/ b_i) * bin (r_{hi}, v_{ghi}) \quad \text{Eq. 19}$$

Here the binomial distribution (with $x = r_{hi}$, $p = v_{ghi}$) is

$$P[X = k] = \begin{cases} \binom{n}{k} p^k (1-p)^{n-k} & 0 \le k \le n \\ 0 & \text{otherwise} \end{cases} \quad \text{Eq. 20}$$

where

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad \text{Eq. 21}$$

When we **assume** the distributions of defects D_{gi} are independent given rework deficits r_{hi} and verification v_{ghi}, we can easily describe the distribution of joint conformance levels. Unfortunately, for large problems working with the joint distribution **D** on all D_{gi} can be analytically cumbersome.

Fortunately, we often can simplify the analysis greatly. If we additionally **assume** that the number of subtasks m is large, the central limit theorem indicates that the mean will be a highly reliable point estimator:

$$d_{gi} = E(\sum_h (1/ b_i) * bin (r_{hi}, v_{ghi})) \quad \text{Eq. 22}$$

$$d_{gi} = \sum_h (1/ b_i) * E(bin (r_{hi}, v_{ghi})) \quad \text{Eq. 23}$$

$$d_{gi} = \sum_h (1/ b_i) * r_{hi} * v_{ghi} \quad \text{Eq. 24}$$

In general, substituting point estimators must be done with care [Law and Kelton]. However, where the above assumptions hold, we can calculate—with probability approaching one— the values d_{gi} = the probabilities that a randomly sampled subelement from task or rework link i conforms to specification requirements of goal g.

Sometimes these assumptions may not hold, for example, when conformance among subtasks is dependent given a set of actions, or when the number of subtasks is small. In this case, we can attempt to work through the subsequent steps using the analytic form. If this is not feasible, we can sustain VDT's simulation paradigm by randomly sampling the joint defect distribution and performing later analytic steps on each sampled result.

6.7 TESTING-POSTERIOR PRODUCT CONFORMANCE RATES

Many projects include a separate test phase between development and operations. This phase operates the developed product in various ways to determine whether it meets requirements, and if not, calls upon the engineering organization to rework the element. The advantage of an independent test stage is that it provides a level of redundancy in verifying the engineering conformance of a product.

To model a test stage, we create the test stage as an interdependent sub-project in VDT with links to design and development. This project could, for example, include a task for each of the designed elements *(*components, interfaces, and multi-systems*)* that will be tested. By creating the testing task explicitly in VDT, we capture behaviors such as the creation of design and development rework based on test results, and the dependency of those results on the original quality of the design and development.

Test coverage is the fraction of requirements *(*weighted by likelihood in operations*)* that are simulated and assessed. Short and simple test stages generally offer lower test coverage than more detailed efforts. The corresponding tasks' complexities and rework volumes will be greater when their test coverage is large.

We use the estimated degree of test stage conformance to model the impact of a test stage on product conformance. Specifically, our model **assumes** that if the design review process is executed perfectly, it will eliminate a fraction of the design's nonconformances that is equal to the specified design test coverage. If the design review is not performed well, it will have a less positive effect. Design reviews and product testing do not directly reduce design or development conformance under this model. Instead, we characterize the review of prior work using expert-assessed levels z_{gin} of test coverage. Each task n that is dedicated to finding defects in the product of task i that can impact goal g has a degree of coverage z_{gin}. We can use z_{gin} along with the levels of defects in both the original task and in testing itself to update our estimate of the product conformance thus:

$$D'_{gi} \sim D_{gi} + (1 - D_{gi}) * z_{gin} * D_{gn} \qquad \text{Eq. 25}$$

For instance, a value of 0.5 for z_{123} indicates that task 3 tests half of task 2's conformance to goal 1 *(*critical*)* requirements. This model easily accommodates the incidental review of prior work, such the review of designs at the beginning of development tasks, by using low but non-zero values (z_{gin} is zero wherever i = n because internal testing is already calculated into the rework deficit).

When a project employs testing tasks of this kind, we can simply substitute the values d'_{gi} for the original d_{gi} values in **d**. When more than one testing step occurs, we can apply the same method recursively. Finally, we can proceed without any further variation through later analysis steps.

6.8 ILLUSTRATION

Calculations Using Hypothetical Data

In this section, we illustrate each step of the proposed method using hypothetical data.

Table 6.8.1: Sample Data on Product Defects

m_v The posterior probability of product conformance given different engginering stage actions

	Critical					Non-Critical				
	Routine	Rework	QuickFix	Ignore	Default	Routine	Rework	QuickFix	Ignore	Default
Ready Shuttle	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Ready Launch	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Ready Ground	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Shuttle-Launch Rework Link	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Shuttle-Ground Rework Link	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
Launch-Ground Rework Link	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%
System Integration Meetings	80%	80%	80%	60%	60%	80%	80%	60%	60%	60%

d The conformance (lack of defects) of each engineered element

	Critical	Non-Critical
Shuttle Component	71%	71%
Launch Component	67%	67%
Ground Component	73%	70%
Shuttle-Launch Interface	61%	61%
Shuttle-Ground Interface	74%	73%
Launch-Ground Interface	74%	71%
Integrated Multi-System	75%	75%

m_z The coverage of each engineering stage task's testing activities on output from other tasks

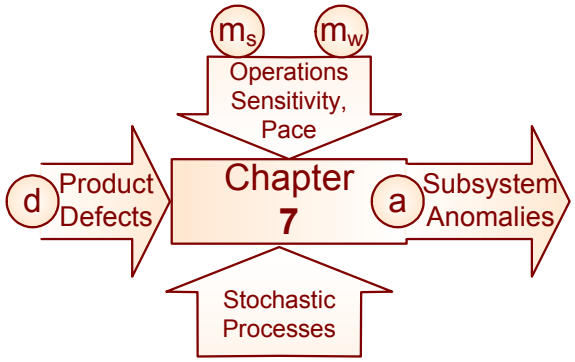
	Critical						Non-Critical					
	Shuttle Component	mponent	mponent	Interface	nd Interface	lti-System	omponent	omponent	omponent	h Interface	d Interface	Multi-System
Shuttle Component	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Launch Component	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Ground Component	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Shuttle-Launch Interface	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Shuttle-Ground Interface	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Launch-Ground Interface	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Integrated Multi-System	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

d' = d Testing has no impact on conformance in this example

Chapter 7 - Operating Subsystem Anomalies

Estimating f(***a*** | ***d***, $\boldsymbol{m_w}$, $\boldsymbol{m_s}$) *Using Stochastic Processes*

7.1 ANOMALIES DURING OPERATIONS



Complex engineering projects' tendency to create products that do not fully conform to specifications often causes unexpected and undesired behavior by engineering systems during operations. An *anomaly* is an initiating event in which an engineered subsystem behaves in a manner that violates the requirements presented in its specification. In this section, we estimate the rate at which anomalies will manifest when an engineered element is operated, and the total number that will manifest within a given timeframe. We can extend the model to include additional non-engineering anomaly sources, such as actions by human operators or by environmental factors, using a parallel method.

Some operations call upon engineered elements to meet the requirements of their specifications. As we have seen, these elements generally do not perfectly conform to the specification, and so they will sometimes fail to meet the requirements demanded of them during operations.

When a requirement is placed that falls outside the engineering specification, the product's degree of conformance to specifications does not influence the satisfaction of this requirement. In this case, we say that the engineered product's behavior *(*probability of anomaly*)* is *engineering-conformance-independent*. We can assess the probability of an anomaly under these circumstances – an *engineering-conformance-independent-anomaly*– through standard means such as statistical inference or expert assessment.

When operations place a product under a requirement that does appear in the specification, we say that the engineered product's failure probability is *engineering-conformance-dependent*. A *design-dependent anomaly* is an anomaly that results from nonconformance in the design product, and a *development-dependent* anomaly results from development shortcomings. In either of these cases, the initially developed product may not be able to meet the operating requirement.

Table 7.1.1: Notation Introduced for Subsystem Anomalies Analysis

Term	Name	Type / Source	Description	Example
b	Homogeneous anomaly rate	Continuous random variable; Element of matrix B	Static rate at which anomalies of all subsystems manifest in regard to all goals, for each engineered product defect.	When β_{21} has value 0.81, this indicates that during operations, anomalies of goal 1 occur in engineered subsystem 2 approximately 0.81 time units apart.
b_{gj}	Static anomaly rate	Real-valued constant	Static rate at which anomalies of goal g manifest in a system j as a result of engineering.	When β_{21} has value 0.81, this indicates that during operations, anomalies of goal 1 occur in engineered subsystem 2 approximately 0.81 time units apart.
$b_{gj}(t)$	Dynamic anomaly rate	Function returning a continuous random variable	Changing rate at which anomalies of goal g manifest at a given point in time, in a system j, for each engineered product defect.	$b_{21}(t) = 0.9^{2t}$ indicates that toward the beginning of operations, system 2 exhibits anomalies of goal 1 during most time units, but they occur less frequently as operations continue.
$B_{gj}(t)$	Stochastic anomaly rate	Continuous-valued stochastic process	Uncertain and changing rate at which anomalies of goal g manifest at a given point in time, in a system j, for each engineered product defect. Realizations are $b_{gj}(t)$	
ζ_{gj}	remedy	Real-valued constant	The rate at which each manifest anomaly is fixed during operations, so that for h flaws manifest, the rate of fixing is h ζ_{gj}	$\zeta_{12} = 0.9$ indicates that a single anomaly that affects subsystem 2's support of goal 1 will be fixed after 0.9 time units on average. If there are two anomalies, they will both be fixed in about the same amount of time.

7.2 PACE AND SENSITIVITY OF SUBSYSTEM OPERATIONS

Over time, operations rely in different ways upon the product of engineering tasks (even though we **assume** that their engineered conformance does not change). We define an *anomaly* to be a significant behavior in an operational system that differs from what was targeted by the specification. The probability of an anomaly per unit time equals the probability that an operational constraint is encountered, times the probability that the specification's corresponding engineering requirement is not satisfactorily met in this situation (that it does not conform).

Three factors determine the probability that an anomaly will occur in a particular system. First, the engineered subsystem must currently be operating at a certain pace, which we define as the rate at which new operating requirements are placed. As an example, a pace of 1.0 means that in one time unit, the function generally calls upon as many requirements as appear in the specification (although some may be tested multiple times, and in different ways). On average, a function's will produce anomalies at half that rate if the pace of operations is 0.5.

The second factor is the probability of that a constraint encountered in operations was engineered for properly. Each time a requirement is tested, there is a probability that the engineered element will not conform, and will violate an operations constraint. The probability of a randomly selected requirement being met by an engineered element is just its conformance, as found in the conformance matrix. In the previous section, we developed this value as d_{gi} – the goal g conformance level of an engineering task i.

The third factor is the probability that a requirement placed upon an unsatisfied engineering requirement results in a deviation from operating expectations. This is the probability of an anomaly, given that the engineering requirements upon which operations depend were *not* met during the engineering project.

Thus, few anomalies will generally emerge during operations when a system is operated infrequently, or if operations are robust to engineering specification violations, or when the engineering tasks met most of the requirements. More anomalies tend to emerge if a

system's behavior is highly dependent upon design and development quality, the system is operated continuously, and/or engineering tasks were not completed effectively.

Since requirements may be tested in different ways, meeting one test of a requirement does not guarantee it will be met later. We model time as continuous, and we **assume** the manifestation of requirements as independent, so the probability of simultaneous manifestations *(*requirements*)* is zero [Law and Kelton]. We **assume** that each time unit of operations calls upon a large number of requirements. Finally, we **assume** that anomalies cannot happen without new requirements being encountered in operations, anomalies only result from non-conforming products, and conformance does not change over time *(*since it was determined during the engineering stages*)*.

To illustrate, the correct operations of a "telecommunications system" might be equally sensitive to the effective conduct of thirteen engineered elements. These would be the design and development tasks for telecommunications hardware, software, and human factors, the three interfaces among them, and finally a multi-system element that is the subject of periodic staff meetings.

By consolidating the influences of these tasks into models of system anomalies, we capture the behavior of root causes—initiating events—that compromise the functions upon which project failure depends.

7.3 MODEL 1: HOMOGENEOUS OPERATION OF ENGINEERED ELEMENTS

$s_{gij}(t)$ is the *sensitivity* of operations – the probability of significant change in function j when encountering requirement violations of goal g from dependent task i, at a time t.

$w_{gij}(t)$ is *pace* of operations – the rate at which function j calls upon features provided in task i's goal g specification. In most cases, $w_{gij}(t)$ is zero for all t because the relationships between engineered elements i and operating systems j is sparse. Formally, these factors combine to provide the following formula:

$$b_{gj}(t) = \sum_i d_{gi} * w_{gij}(t) * s_{gij}(t) \qquad \text{Eq. 1}$$

Intuitively, $b_{gj}(t)$ is the rate, at time t, at which anomalies with the potential to affect goal g manifest in a system j as a result of engineering nonconformance.

In this paper, we **assume** $w_{gij}(t)$ and $s_{gij}(t)$ to be deterministic and known in advance. The next section identifies several simple cases that are mathematically simpler to analyze.

Operations Description

In the simplest models, we define the emergence of anomalies during operations using the simplest of distinctions from the engineering phase. Specifically, we define one operating subsystem for each task that was executed in the engineering phase. We also **assume** that each engineered element is operated by a dedicated operations subsystem at the same, constant pace w and sensitivity s.

As we will see, we can use the simplified model even when functional failure results from *(*for example*)* design anomalies producing a design function failure, *or* development anomalies producing a development failure, because in this case each function can rely upon the behavior of multiple, independent lower-level functions.

Rate of Anomalies

Because anomalies manifest independently of one another, the time between them is distributed exponentially.

In this case, the rate of anomaly events in each subsystem is proportional to the number of defects in its corresponding engineering task:

$$b_{gj} = d_{gj} * w * s \qquad \text{Eq. 2}$$

Distribution of Total Anomalies

The rate at which multiple exponential samples occur equals the sum of their rates, and the number of samples from an exponential distribution that occurs in a given period is mathematically guaranteed to follow a Poisson distribution [Law and Kelton].

$$P(a_{gj}(t) = n \mid b_{gj}) = Poisson(n, b_{gj} * t) \qquad \text{Eq. 3}$$

$$P(a_{gj}(t) = n \mid b_{gj}) = e^{-(d_{gj} * w * s * t)} (d_{gj} * w * s * t)^n / n! \qquad \text{Eq. 4}$$

Point Estimate of Total Anomalies

In some cases, we may simplify our analysis by approximating the distribution on the total number of failures by using the expected value. We estimate the average number of anomalies that have occurred at a time t_* using a basic property of the Poisson process:

$$a'_{gj}(t) = E(a_{jg}(t)) = E(Poisson(b_{gj} * t)) \quad \text{Eq. 5}$$

$$a'_{gj}(t) = b_{gj} * t \quad \text{Eq. 6}$$

$$a'_{gj}(t) = d_{gj} * w * s * t \quad \text{Eq. 7}$$

7.4 Model 2: Static Operation of Compound Subsystems

Operations Description

We cannot use these simplifications, however, and we must use the full matrix form of O, in a number of cases. For example, the homogeneous model cannot accurately predict the distribution of anomalies with multiple root causes when, for example, design anomalies and development anomalies *combine* to reduce the capacity of a single function that succeeds or fails as a unit.

To model this case we first define a *compound subsystem* or simply *subsystem* as a set of engineered elements (each produced by different engineering tasks) that act in concert during operations. In this model, we consider anomalies to emerge within subsystems because of the combined behaviors of the integrated, engineered elements.

The model remains simple however when we **assume** that the pace w_{gij} and sensitivity s_{gij} of operations is fixed over time for a given product element i, system j, and goal g.

Operations Description

Rate of Anomalies

In this "static operations" case, we define b_{gj} as the rate of subsystem j anomalies that can impact goal g:

$$b_{gj} = \sum_i d_{gi} * w_{gij} * s_{gij} \quad \text{Eq. 8}$$

Calibrating the values for the w_{gij} and s_{gij} requires expert assessment and/or statistical inference. Specifically, experts may be able to provide a number of intuitive values that

we can transform into the variables in our equation. These include the rate at which anomalies occur in a system due to problems in engineering, the average time between such anomalies *(*1/pace*)*, and the probability that a nonconforming subelement will create an anomaly during the life of operations *(*rate/mission length*)*. If statistics are available on the rates at which anomalies have occurred in the operations of comparable products, and how these anomalies influenced different functions, we can combine these values with expert assessed figures using standard techniques from PRA.

Distribution of Total Anomalies

Because b_{gj} is static we can estimate the total number of anomalies $a_{gj}(t)$ as a Poisson process:

$$A_{gj}(t) \sim Poisson(b_{gj} * t) \quad \text{Eq. 9}$$

$$p(a_{gj}(t)) = e^{-(b_{gj} * t)} (b_{gj} * t)^{a_{gj}(t)} / a_{gj}(t)! \quad \text{Eq. 10}$$

Joint Distribution of Total Anomalies

Point Estimate of Total Anomalies

In the case where the rate and pace of operations are certain and static (Eq. 20-21), we have:

$$a_{jg}(t_*) = \int_{t=o}^{t_*} \sum_i d_{gi} * w_{ijg} * r_{ijg}\, dt \quad \text{Eq. 11}$$

$$a_{jg}(t_*) = \sum_i d_{gi} * w_{ijg} * r_{ijg} * t_* \quad \text{Eq. 12}$$

7.5 MODEL 3: DYNAMIC SUBSYSTEM OPERATIONS

Operations Description

In this case we define operations as a dynamic process with $w_{gij}(t)$ and $s_{gij}(t)$ as our pace and sensitivity over time.

Rate of Anomalies

We can generalize Eq. 8 to get

$$b_{gj}(t) = \sum_i d_{gi} * w_{gij}(t) * s_{gij}(t) \quad \text{Eq. 13}$$

Distribution of Total Anomalies

Therefore, we can calculate the probability of a given number of anomalies $a_{gj}(t)$ as:

$$P(a_{gj}(t_*) = x \mid b_{gj}()) = Poisson(x, \int_o^{t_*} b_{gj}(t)\, dt) \quad \text{Eq. 14}$$

$$= Poisson(x, \int_o^{t_*} \sum_i d_{gi} * w_{ijg}(t) * s_{ijg}(t)\, dt) \quad \text{Eq. 15}$$

When we **assume** $p_{ijg}(t)$ and $s_{ijg}(t)$ are deterministic functions (meaning that operations follow an essentially known "path"), the values $a_{gj}(t)$ are independent (given d_{gi}). In this case we can easily compute the joint distribution on $A_{gj}(t)$ for all g and j as the product of their individual distributions.

Joint Distribution of Total Anomalies

Point Estimate of Total Anomalies

To develop a point estimate of the number of anomalies that will manifest at time t, we can **assume** that the total number of requirements being tested is large. This is the same assumption that justified a point estimate of defects, and one that is particularly justified if the number of engineering subtasks used in VDT is large.

The anomalies occurrences are independent and identically distributed (iid), so we can use the central limit theorem to approximate the total number of anomalies using the mean [Law and Kelton]. The expected number of anomalies (relevant to goal g) that have manifested by time t_* and affect each function j is:

$$E(a_{jg}(t_*)) = \int_o^{t_*} \sum_i d_{gi} * p_{gij}(t) * s_{gij}(t)\, dt \quad \text{Eq. 16}$$

7.6 MODEL 4: UNCERTAIN SUBSYSTEM OPERATIONS

In this section, we will relax the assumption that the pace and sensitivities of operations are deterministic and known in advance. In this case we define operations as a stochastic process with $w_{gij}(t)$ and $s_{gij}(t)$ as our pace and sensitivity over time.

Operations Description

Rate of Anomalies

Distribution of Total Anomalies

Joint Distribution of Total Anomalies

Point Estimate of Total Anomalies

7.7 Model 5: Long-Run Correction of Anomalies

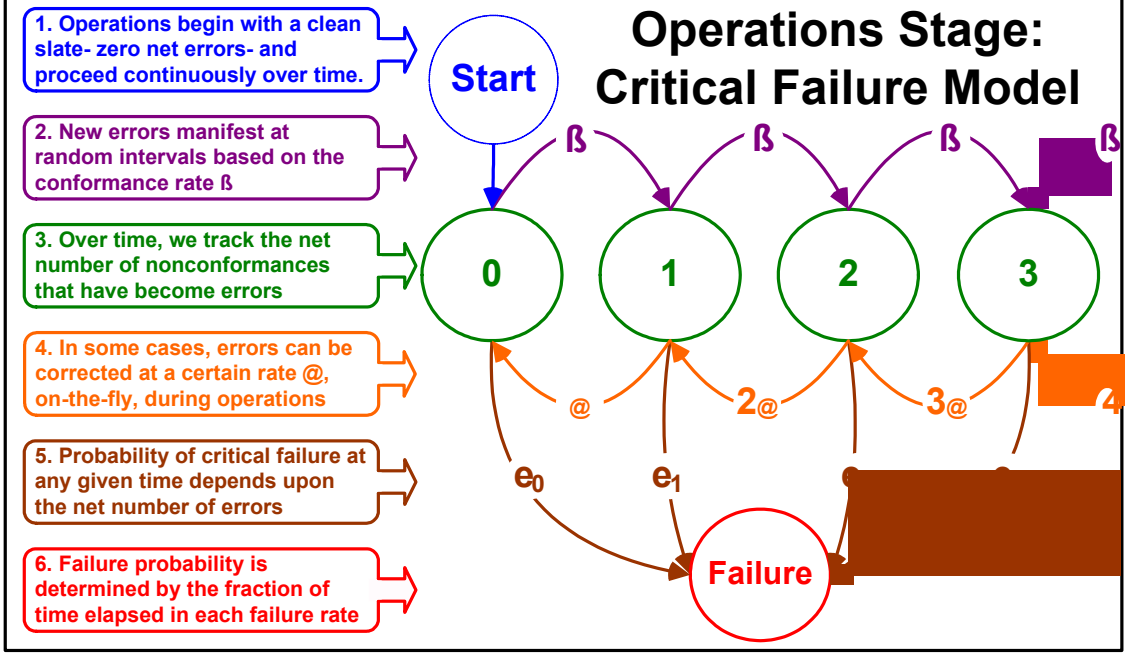


Figure 8 m/m/1 Long-Run Operations Stage Critical Failure Model
This formulation allows us to estimate the steady-state failure probability when anomalies are probabilistically created and fixed over time.

Sometimes operations stretch over a long period and the risk of failure at any given point in time is low. However, in our simplest model, anomalies manifest at the same non-zero rate over time, so the total number of anomalies in the long run is infinite. This measure of long-term system behavior is of limited value.

Under these project circumstances, operators are often able to fix anomalies as well as to create them. For example, including human astronauts in an extended space mission, or coding important elements in remotely adjustable software, makes possible the development of long-term workarounds that bring operations back into conformance.

Figure 10 shows that when at any given time we have a chance of correcting each anomaly, we can model the total number of flaws as an **m**/**m**/1 queue. If the rate at which each manifest anomaly is fixed is ζ *(*so that for h flaws manifest, the rate of fixing is h ζ*)*, then in the long run *(*for the simplest case*)* we have

$$p(a_{gj}) = e^{-(d_{gj} * \zeta)} * (d_{gj} * \zeta)^{a_{gj}} / a_{gj}! \qquad \text{Eq. 17}$$

Note that being able to fix emergent anomalies is distinct from the ability to reduce the rate of flaws emerging, which we can model by reducing the pace and/or sensitivity of operations over time using the dynamic $w_{gij}(t)$ and $s_{gij}(t)$ forms.

7.8 ILLUSTRATION

Calculations Using Hypothetical Data

In this section, we illustrate each step of the proposed method using hypothetical data.

Table 7.8.1: Sample Data on Subsystem Anomalies

m_w The pace at which subsystems employ engineered elements during operations

	Critical					Non-Critical				
	Weather	Ground	Launch Ve	Shuttle	Science Go:	Weather	Ground	Launch Vehce	Shuttle	Science Goals
Shuttle Component	0%	0%	0%	100%	0%	0%	0%	0%	100%	0%
Launch Component	0%	0%	100%	0%	0%	0%	0%	100%	0%	0%
Ground Component	0%	100%	0%	0%	0%	0%	100%	0%	0%	0%
Shuttle-Launch Interface	0%	0%	50%	50%	0%	0%	0%	50%	50%	0%
Shuttle-Ground Interface	0%	50%	0%	50%	0%	0%	50%	0%	50%	0%
Launch-Ground Interface	0%	50%	50%	0%	0%	0%	50%	50%	0%	0%
Integrated Multi-System	0%	33%	33%	33%	0%	0%	33%	33%	33%	0%

m_s The sensitivity of subsystems to defective behavior by engineered elements during operations

	Critical					Non-Critical				
	Weather	Ground	Launch Ve	Shuttle	Science Go:	Weather	Ground	Launch Vehce	Shuttle	Science Goals
Shuttle Component	0%	0%	0%	100%	0%	0%	0%	0%	100%	0%
Launch Component	0%	0%	100%	0%	0%	0%	0%	100%	0%	0%
Ground Component	0%	100%	0%	0%	0%	0%	100%	0%	0%	0%
Shuttle-Launch Interface	0%	0%	50%	50%	0%	0%	0%	50%	50%	0%
Shuttle-Ground Interface	0%	50%	0%	50%	0%	0%	50%	0%	50%	0%
Launch-Ground Interface	0%	50%	50%	0%	0%	0%	50%	50%	0%	0%
Integrated Multi-System	0%	33%	33%	33%	0%	0%	33%	33%	33%	0%

b The rate at which engineering-induced anomalies manifest in operations

	Weather	Ground	Launch Ve	Shuttle	Science Goals
Critical	0.0%	8.6%	8.0%	8.2%	0.0%
Non-Critical	0.0%	8.4%	7.9%	8.1%	0.0%

a(2) The total number of anomalies that occur in each system by time 2 during operations (point estimate)

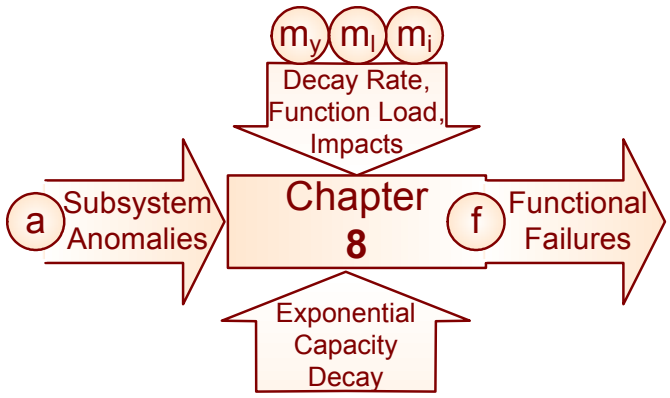
	Weather	Ground	Launch Ve	Shuttle	Science Goals
Critical	-	0.17	0.16	0.16	-
Non-Critical	-	0.17	0.16	0.16	-

Chapter 8 - Operations Function Failures

Estimating f(***f*** | ***a***, $\boldsymbol{m_y}$, $\boldsymbol{m_i}$, $\boldsymbol{m_l}$) *Using Exponential Decay*

8.1 FUNCTIONAL CAPACITIES DURING OPERATIONS

Introduction



In this section, for each functional block, we calculate the probability of failure per time unit during operations. We begin by decomposing the operations into a set of functional blocks using any of the PRA methods that we wish.

The next step is to distinguish between those factors whose failure probabilities are directly influenced by engineered subsystems behavior, and those whose failure probabilities are independent.

In most projects, many factors that influence failure probability are not influenced by engineering-dependent anomalies. We can assess failure probabilities for functions that are independent of engineered subsystems' behavior, such as external events, operations actions, and environmental conditions, without recourse to the VDT output, using techniques described in the PRA literature such as statistical inference or expert assessment.

Once this formula is complete, however, our analysis becomes integral to determining the probability of failure for those functions that rely upon the correct operation of engineered subsystems.

Because we will estimate the probabilities of design- and development- dependent failure, we require that the hardware portions of this decomposition be performed to the level of components, interfaces, and interdependent systems-of-systems.

Table 8.1.1: Notation Introduced for Function Failure Analysis

Term	Name	Type / Source	Description	Example
$c_{kg}(t)$	capacity	Real	Characterizes capacity- the resistance to operational pressures toward function k having a failure of goal g.	$y_{21} = 0.5$ means that due to the manifestation of anomalies during operations, function 2 is only operating at half of nominal capacity with regards to goal 1 project failure risk.
C	Capacity			
$F(x)$	Cumulative Probability Distribution			

Function Capacities

When an anomaly manifests during operations, it can stimulate differences in behavior that result in additional anomalies and eventually catastrophe. In contrast, under some circumstances these problems may not lead to any significant consequence. As a third possibility, a single anomaly that can be endured without failure may interact with another, independently manageable anomaly, and cause failure.

In this section, we **assume** that it is the total number of independent anomalies that have occurred up to a target point in time t that determines the capacity of an operational function.

In most models, there will be factors other than engineered subsystems anomalies that influence capacity, either positively or negatively. For example, we can define a low-probability micrometeoroid strike against an orbiter's hull to reduce the capacity of the thermal protection function by a large percentage.

Our first task is to derive the total number of goal g anomalies that have manifested during operations, *and* that are able to influence each function k. To do this, we multiply the total goal g anomalies $a_{gj}(t)$ manifested in each engineered subsystem j by a coefficient i_{gjk} representing the influence over function k. As part of the project definition, we collect this information i_{gjk} about which functions make use of which subsystems from management.

Our analysis **assumes** that these values are fixed and certain, although generalizing to time-dependence or stochasticity can follow methods like those in the previous chapter. We further **assume** that the impact of anomalies is to reduce functions' capacity exponentially. Intuitively, we have some resilience, but anomalies cause "decay", or operations lose access to orthogonal "degrees of freedom". In this model, each anomaly during operations will have the same impact: cutting a residual capacity by some fraction u_{gk} ($0 <= u_{gk} <= 1$) defined by management in the project plan.

We can use this data to define $c_{gk}(t)$ as the capacity at time t that a function k will provide in support of goal g:

$$c_{gk}(t) = u_{gk}^{\sum_j (a_{gj}(t) * i_{gjk})} \quad \text{Eq. 1}$$

For example, if $u_{gk} = 0.5$, then we will have 0.5 capacity when one anomaly occurs, 0.25 when two occur, 0.125 when three occur, etc.

8.2 FUNCTIONAL FAILURE

Function Loads

We model the load that functions are subjected to as **ml**, a matrix with elements l_{gk} that each characterizes the pressure placed on function k's support of goal g. At this point, we **assume** that each l_{gk} is distributed uniformly over [0,1] and is fixed over time. This provides a clear interpretation of behavior during operations, and simplifies our mathematical analysis, but is not necessary for any other reason.

In many models, it will be appropriate to model non-engineering factors that probabilistically influence load, either positively or negatively. For example, maneuvers undertaken to avert an unanticipated encounter with space junk might place an additional load upon a spacecraft's attitude control and propulsion functions.

Functional Failure

Our next step is to calculate a distribution on the time at which each function ceases to serve the project goals it is responsible for. We define a set of random variables F_{gk} whose realization f_{gk} represents the time at which the function k ceases to support goal g. We **assume** that failure occurs when load exceeds capacity:

$$F_{gk} \sim Min(t \mid C_{gk}(t) < L_{gk}) \quad \text{Eq. 2}$$

$$f_{gk} = Min(t \mid c_{gk}(t) < l_{gk}) \quad \text{Eq. 3}$$

Because capacity $C_{gk}(t)$ ranges from zero to one, if load l_{gk} is one, any anomaly will cause a goal g operations failure in function k. If l_{gk} is zero, no amount of anomalies will cause the function k to fail with regard to goal g. Note that in this model we **assume** that capacity never increases, and so we have only one time f_{gk} at which capacity falls below load.

Equation 3 provides the probability that a function has failed before a given time, also known as a cumulative distribution function on the time to functional failure, $F(f_{gk})$ (shorthand for $F_{F_{gk}}(f_{gk})$).

$$F(f_{gk}) = F_{F_{gk}}(f_{gk}) \quad \text{Eq. 4}$$

$$F(f_{gk}) = P(F_{gk} <= f_{gk}) \quad \text{Eq. 5}$$

$$F(f_{gk}) = P(C_{gk}(f_{gk}) < L_{gk}) \quad \text{Eq. 6}$$

Since we have assumed that L_{gk} is distributed uniformly on [0,1], and $C_{gk}(t)$ is also distributed on [0,1], we can simplify Eq. 3 to get:

$$F(f_{gk}) = P(0 < L_{gk} - C_{gk}(f_{gk})) \quad \text{Eq. 7}$$

$$F(f_{gk}) \sim 1 - C_{gk}(f_{gk}) \quad \text{Eq. 8}$$

We differentiate this simple cumulative form to compute the probability density function $f(f_{gk})$ on a given time to failure, f_{gk}:

$$f(f_{gk}) = F'(f_{gk}) \quad \text{Eq. 9}$$

$$f(f_{gk}) \sim d(1 - C_{gk}(f_{gk}))/df_{gk} \quad \text{Eq. 10}$$

$$f(f_{gk}) \sim d(-C_{gk}(f_{gk}))/ df_{gk} \quad \text{Eq. 11}$$

Substituting from Equation 1 we get

$$f(f_{gk}) \sim d(- u_{gk}^{\sum_j (A_{gj}(f_{gk}) * i_{gjk})})/ df_{gk} \quad \text{Eq. 12}$$

Because $d(-x^{g(t)})/dt = - x^{g(t)} * ln(x) * dg/dt$, we have

$$f(f_{gk}) \sim - C_{gk}(t) * ln(u_{gk}) * d(\sum_j (A_{gj}(f_{gk}) * i_{gjk}))/ df_{gk} \quad \text{Eq. 13}$$

$$f(f_{gk}) \sim -ln(u_{gk}) * C_{gk}(f_{gk}) * (\sum_j i_{gjk} * dA_{gj}(f_{gk})/ df_{gk}) \quad \text{Eq. 14}$$

We have moved the negative sign to the natural log term for two reasons: first to collect our constants together, and second to reduce the chance of misperception that a negative probability could result ($f(f_{gk})$ is always positive because $ln(x)$ is negative for $0<x<1$).

8.3 FULL EXPANSION USING A POINT ESTIMATE ON STATIC OPERATIONS

We illustrate this formula using an expected-value point estimate on static operations. Substituting the formula for $a_{gj}(t)$ from Equation 26 we get:

$$f(f_{gk}) = -ln(u_{gk}) * c_{gk}(f_{gk}) * (\sum_j i_{gjk} * d(\sum_i d_{gi} * p_{gij} * s_{gij} * f_{gk})/ df_{gk}) \quad \text{Eq. 15}$$

$$f(f_{gk}) = -ln(u_{gk}) * c_{gk}(f_{gk}) * (\sum_i \sum_j d_{gi} * p_{gij} * s_{gij} * i_{gjk}) \quad \text{Eq. 16}$$

Intuitively, the probability that failure occurs at a given time f_{gk} equals a constant (based on capacity decay rate), times the capacity at that time, times the (expected) number of defects that can create anomalies that impact the function.

Cases in which these terms vary over time are similarly easy to solve when the functions are differentiable.

Expanding d_{gi} based on Equation 26 traces failure times to engineering rework deficits:

$$f(f_{gk}) = -ln(u_{gk}) * c_{gk}(t) * (\sum_h \sum_i \sum_j (1/b_i) * r_{hi} * v_{ghi} * p_{gij} * s_{gij} * i_{gjk}) \quad \text{Eq. 17}$$

This equation treats r_{hi} as deterministic. In general, however, we are uncertain about r_{hi} because it results from complex engineering stage behaviors. We can incorporate the distribution of engineering phase behaviors created using the VDT simulator, as adapted from Equation 12. Recall that the joint probability distribution on rework behavior equals the fraction of simulation trials that produces that behavior distribution. Since r_{hi} is a random variable based on VDT output **r** we have:

$$f(f_{gk}) = -ln(u_{gk}) * c_{gk}(f_{gk}) * \sum_{R_r} P(\mathbf{r}) * \sum_h \sum_i \sum_j (1/b_i) * r_{hi} * v_{ghi} * p_{gij} * s_{gij} * i_{gjk}) \quad \text{Eq. 18}$$

$$f(f_{gk}) = -ln(u_{gk}) * c_{gk}(f_{gk}) * \sum_{R_r}(|\{\psi_s|(\mathbf{t_e},\mathbf{r})=Vdt(\mathbf{me},\psi_s), i \in \{1,2,\ldots n\}\}/n) * \sum_h \sum_i \sum_j (1/b_i) * r_{hi} * v_{ghi} * p_{gij} * s_{gij} * i_{gjk}) \quad \text{Eq. 19}$$

We can use the same method to include other uncertainties in our models, by summing or integrating over the joint distribution of possible values. For example, we can model

different possible sequence of operations by integrating over the possible values of operations pace $W_{gij}(t)$ and sensitivity $S_{gij}(t)$.

The equation becomes more intuitive if we introduce a random variable Z_{gk} to represent the potential impact per unit of time from engineering stage rework deficits:

$$F_{gk} \sim -ln(u_{gk}) * u_{gk}^{Z_{gk} * f_{gk}} * Z_{gk} \quad \text{Eq. 20}$$

$$Z_{gk} \sim \sum_h \sum_i \sum_j (1/b_i) * R_{hi} * v_{ghi} * p_{gij} * s_{gij} * i_{gjk} \quad \text{Eq. 21}$$

Again using Equations 26 and 1 we have

$$f(f_{gk}) = -ln(u_{gk}) * u_{gk}^{z_{gk} * f_{gk}} * z_{gk} \quad \text{Eq. 22}$$

This intuitively reflects the constant rate of anomalies and resulting exponential degradation of capacity that we have assumed for this illustration.

8.4 JOINT DISTRIBUTION ON FUNCTIONAL FAILURES

The next term we wish to calculate is $f(\mathbf{F} \mid E)$, where each possible realization **f** of **F** is a set of values f_{gk} that represent the failure time of each function k at each goal level g. In cases where we can **assume** that functional failures are independent, given a number of anomalies in each subsystem **a**, we can calculate the probability for a final state very easily:

$$P(\mathbf{f} \mid \mathbf{a}) = P(\mathbf{f} \mid \mathbf{a}) \quad \text{Eq. 23}$$

$$P(F \mid E) = \prod_{kv} ((f_{kv} * P(y_{kv} < l_{mkv})) + ((1 - f_{kv}) * P(y_{kv} >= l_{mkv})) \quad \text{Eq. 24}$$

This is just the product of the failure probabilities over all functions that fail, times the success probability times the probability of success over all functions that succeed. Since l_{mkv} is a uniform distribution on [0,1] we can simplify further:

$$P(F \mid E) = \prod_{kv} ((f_{kv} * y_{kv}) + ((1 - f_{kv}) * (1 - y_{kv})) \quad \text{Eq. 25}$$

8.5 ILLUSTRATION

Calculations Using Hypothetical Data

In this section, we illustrate each step of the proposed method using hypothetical data.

Table 8.5.1: Sample Data on Function Failures

m_i Influence that subsystem behavior has over the success or failure of functions

	Critical					Non-Critical				
	Weather	Ground	Launch Ve	Shuttle	Science Go:	Weather	Ground	Launch Vehice	Shuttle	Science Goals
Launch	100%	100%	100%	100%	0%	100%	100%	100%	100%	0%
Orbit	0%	100%	0%	100%	0%	0%	100%	0%	100%	0%
Land	100%	100%	0%	100%	0%	100%	100%	0%	100%	0%
Collect Science Data	0%	0%	0%	0%	0%	100%	0%	0%	100%	100%

$\sum_j (a_{gj}(2) * i_{gjk})$ The number of impacting errors on each system at time 2

	Critical	Non-Critical
Launch	0.49	0.49
Orbit	0.34	0.33
Land	0.34	0.33
Collect Science Data	-	0.16

m_y Decay in capacity for each function resulting from each operations anomaly

	Critical	Non-Critical
Launch	50%	50%
Orbit	50%	50%
Land	50%	50%
Collect Science Data	50%	50%

c(2) Operating Capacities at time 2 for each operations function

	Critical	Non-Critical
Launch	71%	71%
Orbit	79%	80%
Land	79%	80%
Collect Science Data	100%	89%

$P(F_{gk} < 2)$ (Independent) probabiliy of functional failure by time 2

	Critical	Non-Critical
Launch	29%	29%
Orbit	21%	20%
Land	21%	20%
Collect Science Data	0%	11%

Chapter 9 - Project Goal Achievements

Estimating f(t_g | m_g, t_f) *Using Boolean Algebra*

9.1 PROJECT FAILURE DURING OPERATIONS

We have formally defined a final state, the failure of individual functional blocks, using event F, which is defined by a matrix of values f_{kv} that represent the goal g failure of each of the functional blocks indexed by k. Our model of *P(*F | E*)* provided a joint probability distribution on f_{kv}, based on an upstream engineering management plan m as simulated using random seed s. Our goal in this section is to determine the set of project failures of various severities during operations that occur for a given set of final states F.

To achieve this, we can apply the standard PRA method [Paté-Cornell 2004] to derive a mathematical formula from the final states of each function. In accordance with standard PRA methods, we can illustrate the operations functions intuitively using a block diagram, which we can translate into a fault tree. Using this representation we can distill a simple Boolean formula for *P(*F*)*, then simplify down to "minimal cut sets" that fully characterize the requirements for operational success.

We **assume** that each goal of possible failure v is related to functional blocks defined for that goal, so that:

$$P(w_{wv} | F) = \text{PRA}\ (f_{1v}, f_{2v}, f_{3v}, \ldots) \qquad \text{Eq. 1}$$

Here W is a vector with cells w_{wv} equal to one if a goal g failure will result during operations, and zero otherwise.

If we **assume** a single-string case, in a goal g failure in one function causes total failure for goal g, we have

$$P(w_{wv} = Maximum(\sum_{kv} f_{kv}, 1) | F) = 1 \quad \text{Eq. 2}$$

If instead we **assume** a totally redundant case, in which only a goal g failure in all functions causes total failure for goal g, we have

$$P(w_{wv} = Maximum(\sum_{kv} f_{kv}, 1) | F) = 1 \quad \text{Eq. 3}$$

9.2 ILLUSTRATION

Calculations Using Hypothetical Data

In this section, we illustrate each step of the proposed method using hypothetical data.

Table 9.2.1: Sample Data on Goal Achievements

$P(T_{gg} < 2)$ (Individual) probability of project goal failures

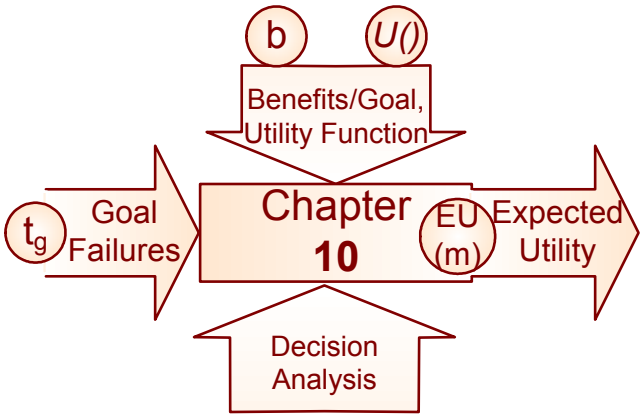
Critical	Critical	Non-Critic	Probability
Failure	45%	40%	
Success	55%	60%	

$P(T_g < 2)$ Joint probabiliy distribution project goal failures

Critical	Non-Critic	Probability
Success	Success	33%
Failure	Success	27%
Success	Failure	22%
Failure	Failure	18%

Chapter 10 - Decision Maker Utility

Calculating U(t_e, t_g, m_e, m_v, m_z, m_w, m_s, m_u, m_y, m_i, m_l, m_g) *and Maximizing* EU(m_e, m_v, m_z, m_w, m_s, m_u, m_y, m_l, m_g)



Our analysis of project planning decisions **assumes** that costs are fixed and spent at the beginning of engineering. We also **assume** that rewards are received at the end of operations, provide benefits independently, and are fixed *(*given success*)* at each goal level. We also **assume**, as suggested by [NASA 2005], that project benefits materialize only when project failure does not occur.

Table 9.2.1: Notation Introduced for Decision Maker Utility Analysis

m_m	Alternate Management Choices	Set of Decision Variables	One possible choice of **m** that we provide as input to the proposed algorithm and calculate the implications of. We then recommend using the plan $M_m = \mathbf{m}_*$ that produces the most desired results.	The set includes a diverse range of organization and process choices, including the definition of all engineering teams within a hierarchy, the tasks within a precedence network, and policies such as centralization and meetings.
$_m$	Management Choice Index	Discrete index	Identifies a set of management choices, including engineering and testing plans, and the use of engineered products during operations in systems that support project objectives.	Index $_m = 1$ for example could identify a conservative, serial engineering approach followed by extended operations in a challenging environment. Index $_m = 2$ could identify an aggressive parallel engineering approach followed by shorter operations in a less hazardous environment.
B	Benefit	Real-valued constant	Amount of benefit received at the end of a fully successful project	A value of 500 million indicating that a total project success produces $500M of value
δ		Real-valued constant	Discount rate	A value of 0.1 indicates that the decision maker views money as of ten percent less worth when it is obtained one year further in the future.
b_0	budget	Real-valued constant	Budget invested at the beginning of a project	A value of 200 million indicates that the total investment in the project is $200M

10.1 FORMAL METHOD

m contains t_{eMax}, which is the maximum length of operations, and b_0, which is the fixed budget of the project. The vector $\mathbf{t_g}$ has cell values t_{gg} that contain the time at which failure occurs for each goal g, and $\mathbf{t_e}$ has cell values t_{ei} that contain the time at which each engineering task i is completed. Thus t_{gg} is always less than or equal to t_{gMax} and t_{ei} is always less than or equal to t_{eMax}:

For all engineering tasks i, $t_{ei} \leq t_{eMax}$ Eq. 1

For all project goals g, $t_{gg} \leq t_{gMax}$ Eq. 2

We also **assume** that the benefits of each objective accrue independently of one another.

$$U(\mathbf{t_e}, \mathbf{t_g}, \mathbf{m}) = U(-b_0 + \sum_g (b_g (t_g / t_{gMax}) * (1/(1+\delta))^{t_{eMax}+t_g})) \quad \text{Eq. 3}$$

The above equation assumes that benefits b_g for each goal g are received at the time of completion (either failure, or end of operations). As an example, if \$100 of benefit accrues for a total success in reaching goal g, with partial success providing benefits equal to the fraction of operations that elapse before failure, times this \$100 benefit, we have:

$$b'_g(x) = \$100 * x \quad \text{Eq. 4}$$

If benefits accrue only at the end of operations, and if no failure occurs, we have:

$$b''_g(x) = \begin{cases} \$100 & \text{if } x = 1 \\ \$0 & \text{otherwise} \end{cases} \quad \text{Eq. 5}$$

When benefits accrue continually we can define a set of functions $b_v(t)$ that provides the (instantaneous) rate at which benefits accrue for goal g at a time t. In this case, we reformulate Equation 1 as:

$$U(\mathbf{t_e}, \mathbf{t_g}, \mathbf{m}) = U(-b_0 + \sum_g (\int_{t=0}^{t_g} (b_g(t) * (1/(1+\delta))^{t_{EMax}+t}) dt)) \quad \text{Eq. 6}$$

We can assist decision makers by using this algorithm to identify which among several possible management choices maximizes this expected utility figure. Mathematically we

can identify discrete sets of prospective management choices $\mathbf{m}_m$, and find a particular $\mathbf{m}_*$ such that

For all $\mathbf{m}_m$, $EU(\mathbf{t}_e, \mathbf{t}_g, \mathbf{m}_m) \leq EU(\mathbf{t}_e, \mathbf{t}_g, \mathbf{m}_*)$ Eq. 7

10.2 ILLUSTRATION

Chapter 11 - Discussion

11.1 CONTRIBUTION

We have responded to our first research question with the claim that *ignoring the need for coordination and rework during engineering phases tends to create product defects that increase the probability of subsequent operational failures.*

Table 4 illustrates the resulting intuition that risk mitigation should involve preventing the overlap of weaknesses among product, organization, process, and environment factors. We first determine the adequacy of an organization to its assigned process, and then the adequacy of the product to endure its operating environment. When a required process will be difficult for a given organization to complete, for example, we should ensure that the product is selected to be highly resilient when compared to its environment (for example by including component redundancy or adopting a slower but more robust operations plan). Similarly, if the product will confront overwhelming hazards, project planners should select an organization that is able to execute the engineering to a very high standard (for example by hiring the best available team, or adding extra test cycles).

In response to our second research question, *we have proposed a model that translates VDT predicted engineering phase choices (exception handling and meeting attendance), into PRA estimates of engineering element failure risks (for components, interfaces, and interdependent multi-systems), which informs the calculation of the operations phase failure risk probability of a given project.*

In general, quantitative decision-making methods are far more effective than qualitative methods. Even in cases where conversation requires qualitative terms, these terms should be grounded precisely so that communication is clear and analysis can be

rigorous. However, one implication of decision makers' bounded rationality is that the best decision making method is determined by the time that is available to conduct analysis. In cases where time is short and data are complex, some organizations adopt qualitative analysis methods to provide some degree of common understanding. Although we feel that this consensus can be illusory, and generally recommend against employing qualitative methods for risk analysis, those who feel otherwise may benefit from our research even if they are unable or unwilling to adopt our analysis method.

In particular, the relationships we have illuminated can provide important guidance even to a high-level qualitative evaluation, such as one based on Failure Mode Effects Analysis [NASA 1995]. However, we have provided a method for the detailed study of project failure risks that employs VDT to model engineering and PRA to model operations. These theory-founded models from the social sciences and engineering can enable project planners to quantitatively assess the complex impacts of changes to any one of the POPE factors, and to assess their interactions.

Table 11.1.1: Qualitative Performance Predictions

Engineering		Process	
		Easy	Difficult
Organization	Strong	Low Risk	Medium Risk
	Weak	Medium Risk	High Risk

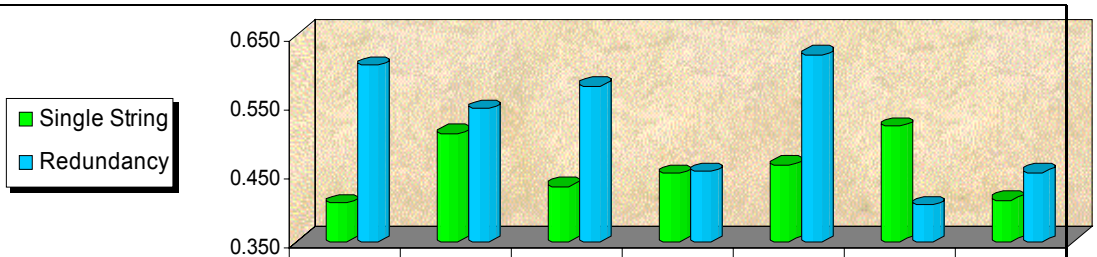
Operations		Product	
		Robust	Fragile
Environment	Safe	Low Risk	Medium Risk
	Hazardous	Medium Risk	High Risk

Total Project Failure Risk		Operations		
		Low Risk	Medium Risk	High Risk
Engineering	Low Risk	Lowest Risk	Lower Risk	Medium Risk
	Medium Risk	Lower Risk	Medium Risk	Higher Risk
	High Risk	Medium Risk	Higher Risk	Highest Risk

At a high level, our model indicates that total failure risk often depends on risks introduced during engineering stages and risks from operations. Engineering risks loosely depend in magnitude on the ability of the organization to execute its assigned process, and operations risks generally result from the robustness of a product relative to its operating environment. Qualitatively, the greatest risks most often (but do not always) occur where weaknesses of organization, process, product, and environment are superimposed. Although these guidelines may be of intuitive value, the proposed method of integrating VDT and PRA provides greater benefits by enabling decision makers to identify and quantify these risks under specific project circumstances.

To understand this contribution's power, consider the decision to employ a redundant telecommunications array on the Huygens probe [JPL 2000]. PRA alone might predict that because the two antennae fail in a probabilistically independent manner, redundancy is would improve the project failure risk. VDT on the other hand, would predict that because the redundant system would require a more complex and uncertain process, greater risk is incurred. Using a qualitative analysis based on table 4, high risk engineering and low risk operations suggest a medium risk total failure probability. We believe that when properly calibrated the proposed integration of VDT and PRA will be able to shed additional light on the tradeoff. Specifically, the method will use VDT and

the integration points to compute the change in antennae failure probability that engineering difficulty produces, and then it will use PRA to calculate the updated project risk based on the component redundancy. The clarity of action this analysis offers is evident in the calculations of Figure 8.



	Payload Component	Payload Interface	Vehicle Component	Vehicle Interface	Ground Component	Ground Interface	Meeting (Systemw i
Single String	0.406	0.507	0.430	0.449	0.461	0.517	0.410
Redundancy	0.605	0.543	0.576	0.451	0.621	0.402	0.450

VDT Risk Measure	SingleString	Redundancy
Payload Component	0.406	0.605
Payload Interface	0.507	0.543
Vehicle Component	0.430	0.576
Vehicle Interface	0.449	0.451
Ground Component	0.461	0.621
Ground Interface	0.517	0.402
Meeting (Systemwide)	0.410	0.450

PRA Reliability	SingleString	Redundancy
Payload Component	94.594%	94.395%
Payload Interface	94.493%	94.457%
Vehicle Component	94.570%	94.424%
Vehicle Interface	94.551%	94.549%
Ground Component	94.539%	94.379%
Ground Interface	94.483%	94.598%
Meeting (Systemwide)	94.590%	94.550%

	SingleString	Redundancy
Communications Loss Rate	0.349	0.365

	SingleString	Redundancy
P(DesignError)	32.5%	32.8%
P(Success)	67.5%	74.9%
Quality	96.5%	96.4%
Product Value	$ 65.17	$ 72.16

Figure 9 Quantitative Failure Risk Calculation for a Payload Redundancy Decision
With a well-calibrated model, the precision of quantitative analysis can support decision-making in the face of difficult tradeoffs. For instance, we produced the chart and table above based on a payload redundancy option that increases product robustness but decreases engineering sureness. Figure 3 provides our PRA fault tree and functional block diagram, while our VDT organization and process are isomorphic to those in Figure 1. In our model, redundancy increases the spacecraft's design complexity, and this raises the probability of engineering flaws causing components to fail. However, the payload's functional redundancy more than compensates, raising the total success probability above that of the "single string" case. Thus the redundant case is more likely to suffer a component failure, but it is also more likely to complete the mission.

11.2 PRACTICAL APPLICATIONS

Risk Analysis

The Columbia Accident Investigation Board's final report includes the following observation:

> *Although schedule deadlines are an important management tool, those deadlines must be regularly evaluated to ensure that any additional risk incurred to meet the schedule is recognized, understood, and acceptable.*

Based on this analysis, the board issued recommendation R62-1:

> *Adopt and maintain a Shuttle flight schedule that is consistent with available resources. Although schedule deadlines are an important management tool, those deadlines must be regularly evaluated to ensure that any additional risk incurred to meet the schedule is recognized, understood, and acceptable.*

The board also asserts that:

> *unless the technical, organizational, and cultural recommendations made in this report are implemented, little will have been accomplished*

The analysis in this paper suggests that existing analysis tools are not sufficient to systematically enable NASA to meet the requirements of NASA 2003 recommendation R62-1. Instead, we believe that NASA will need to implement an integrated planning tool like the one contributed by this research.

The method we propose offers transparent and theory-based support to important project planning trade-offs that previously relied on human intuition. The model of design projects brings together cost, schedule, quality, reliability, and sustainability predictions that can help individual designers to make difficult trade-offs [Chachere 2004.1]. By adjusting and iterating on the design project's POPE factors, we believe it will be possible obtain improved project plans as well as better and more accurate view of their expected performance.

Project Management

Another major concern in complex projects is the prediction of cost, schedule and other types of overruns that will occur in downstream project phases. Benjamin and Paté-Cornell [2004] highlight the need for probabilistic risk analysis in this project setting, and the Jet Propulsion Laboratory's new Risk Station testifies to its perceived importance in project planning [Meshkat and Oberto 2004].

We can use the proposed model to predict the likely accuracy and completeness of upstream efforts to estimate the cost, quality, or schedule of downstream stages, even though these measures are not explicit within the system. As with traditional engineering tasks, project planners require appropriate experts as well as an effective collaborative

process to correctly estimate a project's programmatic risk, costs and schedule. Therefore, we can use the upstream error probabilities to estimate the chance of cost, schedule, and other overruns that result from flawed (unrealistic) designer estimates of downstream behavior.

The proposed method improves the prospects for advancing VDT justification by introducing an explicit product model. Translating VDT results into recommendations is currently difficult because even though VDT predicts schedule improvements, for example, the product often is affected in ways that VDT does not estimate. Providing this ability using the proposed method also improves the joint system's intuitiveness and provides a clear map for enhancements that individual applications require.

The proposed also model compensates for a lack of specificity in some outcome measures that confound efforts to calibrate VDT against real-world projects. The simulator offers a project schedule that estimates the specific days during which each task is executed, and illustrates the number of work items in each team's "inbox" at any given time. However, VDT theory currently describes some of its most important measures of outcome simply as "risks" because they are not explicitly related to a product model. Because the proposed model has a physical interpretation of information processing impacts, in the form of engineered elements' failure probabilities, our contribution lends precision and verifiability to outcome measures, and can therefore improve the ease of precisely calibrating VDT.

11.3 THEORETICAL IMPLICATIONS

POPE Model Proof of Concept

We believe that in project planning, effective risk management requires understanding the *POPE—the* Process, Organization, and Product to be designed, and the Environment in which the product will operate. Project planners assess uncertainties about: how robust their operational *processes* are under different circumstances; the actual performance and properties of the *products* they are building (such as a spacecraft and its support systems); the *organizations* that conduct the mission during its operational phase; and the *environments* in which the product and project operate. Of equal importance, planners must assess the possible interactions among the four factors.

The method we have proposed provides an important existence proof of practical *POPE models*—formal descriptions that span product, organization, process, and environment factors, as well as their interactions. This paper illustrates the feasibility of integrated POPE modeling by emulating the influence that shortcomings in the process and organization that engineer a product have on failure probability in an operating environment. Operations analysis tools can be applied to engineering activities, and engineering analysis tools can be applied to operations, but each is generally much weaker outside its target domain. The proposed method tries to enable several methods to interoperate and to apply their strengths, while compensating as much as possible for one another's weaknesses. Developing models that span all four POPE factors in a justifiable manner is challenging (in technical, theoretical, and practical terms), so proving that a best of breed strategy can produce an effective POPE model is important.

Refinement of Qualitative Theories

The proposed method makes possible a quantitative definition of important theories of risk. VDT's task uncertainty and external rework links capture the "complexity and interdependence" concepts that Normal Accident Theory (NAT) claims are responsible for numerous errors [Perrow]. Similarly, the backlog and exception handling behavior that VDT predicts are important measures of safety culture according to Ciavarelli 2003, and Cooke et al 2003. At the same time that comparing the proposed method's predictions against these theories can help us to formalize their claims, it can also improve VDT's calibration and justification.

Although many of the dynamics this model illustrates have precedent in the social science literature, it is unprecedented for these sometimes-conflicting theories to be operationalized quantitatively.

This improved precision might enable the comparison of competing theories of human and organizational risk management, and the eventual determination of how their dynamics interact under specific project circumstances. For example, the proposed method can cross validate PRA predictions systematically against qualitative risk analysis theories including and High Reliability Organizations Theory (HROT) [Roberts, Weick]. A simple intellective experiment could use idealized or representative POPE models to

illuminate the ways in which NAT's predicted risk sources, complexity and interdependence, balance against HROT's remedies, effective communications and redundancy. We can also use the proposed method to predict how changes in the organization or the product structure could influence failure risks resulting from a rework deficit.

We can also directly change VDT's simulated actor decision-making behavior to approximate changes in safety culture. The contributions that actor decisions make to project risks and rewards are a matter of personal judgment regarding the relative importance of the fundamental outcomes (speed, quality, cost, reliability, and sustainability). In the field, both strategic and tactical mission designers' concern over safety balances against the importance of meeting schedule and cost budgets. A limited analytic focus neglects the complexity of these decisions and can compromise results. For example, a designer in doubt might. Although choosing to rework a component in a simple response to strong safety climate can improve the reliability of a finished product, the increased schedule pressure can lead unexpectedly to stress-induced errors instead [Cooke et al 2003].

Integrated Concurrent Engineering and Latency

Field advances with highly parallel engineering teams demonstrate the potential for orders of magnitude improvements in efficiency [Wall, Mark]. However, existing theories struggle to explain these teams' performance, and computational models have been unable to calculate their costs and benefits accurately [Chachere et al 2004.1, 2004.3]. In particular, it is difficult for managers to determine whether improvements in design schedule are offset by possible increases in project risk [NASA 2004].

Analysis of these teams indicates that a key to understanding them theoretically is the *response latency* metric [Chachere et al 2004.2]. Response latency, the delay between actors' requests and replies, is to information processing what "lead time" is to materials processing in a supply chain.

According to Chachere et al 2004.1 and 2004.3, many factors that are theoretically related to project performance affect organizations only through latency, and this research promises to relate latency to its impacts on project and program risks. Latency is

quantitatively observable metric that offers sufficient precision to compare and calibrate established but currently qualitative theories from the organizational and risk management literature [Chachere et al 2004.2, Chachere et al 2004.3]. In particular, very high latency conditions that arise in VDT simulations cause decision opportunities to be ignored, resulting in a dramatic increase in default dispositions. By linking these defaults in exception handling through conformance measures and errors to project failure risk, the proposed model explains how latency can increase project risk as well as program risk.

11.4 JUSTIFICATION

Justification Process

Both human experts and computer models can add value at any stage of justification, but they are most valuable after they earn stakeholders' confidence [Feigenbaum 1988]. This section lays out a procedural roadmap for our proposed model's justification and compares it with PRA and VDT justifications.

We first review common PRA and VDT validation methods because the integrated model relies crucially on them. After indicating where concerns lie most prominently among academics and practitioners, we indicate where the proposed method contributes or detracts from PRA and VDT validation processes. Finally, we review the state of our proposed model's justification and indicate what next steps may be most appropriate.

It is possible for even flagrantly inaccurate computational models to provide value by suggesting possibilities and alternatives that were not previously considered, and that can be independently explored. Formal models like ours target applications when human experts are unable to predict behavior with sufficient precision. An important milestone for project modelers is to offer performance predictions that significantly improve upon those made by human decision makers. Models are sometimes indispensable, and sometimes offer both advantages and some disadvantages over experts. The proposed model is a step in the development of project models systems that may one day enable projects to be designed using similarly scientific procedures and confidence as engineers enjoy in today's bridge building projects [Levitt and Kunz].

To contribute its full potential toward this aim, however, the proposed model requires a sustained investment in model refinement throughout which the system becomes increasingly useful. To identify the stage that our models have reached, we adopt a validation framework explained in Levitt and Burton [], and in Thomson et al [1999], through which we can build confidence and adapt the models while progressing through a series of increasingly refined stages:

1. **Toy Problems** provide intuitive demonstrations of complex models
2. **Intellective** studies explore theoretical phenomena on cognitively tractable problems
3. **Gedanken** experiments compare expert and model predictions for a real-world project
4. **Retrospective** studies calibrate model predictions against historic documentation
5. **Natural History** predicts outcomes, then compares them against emergent reality
6. **Prospective** studies predict, recommend, and intervene in order to benefit a project

Justification of Foundations

Because our model's broad span relies upon a large number of foundational assumptions, its power stems primarily from integrating a set of existing theoretical results. Our model subjects the base models to additional interpretive rigor, and so in this section we summarize the base models' independent justifications.

PRA Justification

Intuitively conveying PRA's theoretical justification is confounded by the need to recommend with confidence while simultaneously conserving the uncertainty in predictions. PRA has nevertheless achieved a level of popular validation through strikingly accurate predictions such as the Columbia space shuttle's thermal protection system failure [Paté-Cornell and Fischbeck 1993.1, 1993.2]. In some industries, PRA has already achieved a strong tradition of application and operates in the final, "prospective" study phase.

VDT Justification

There is a considerable body of published documentation of VDT modeling efforts, including an important but limited body of predictive application. VDT has had some striking success, notably an accurate prediction of schedule delay, and critical

organizational faults in the cabling contractor for a Lockheed satellite [Levitt et al]. VDT is intended for use on a specific range of routine project activities, and is said to model the "physics" of rational organizational behavior, but not the "chemistry" of, for example, ambiguity [March]. Because calibration also remains an important concern, we view the core VDT model as operating at the "retrospective" level of justification.

Justification of Proposed Method

In addition to calling upon VDT and PRA as base models, the proposed integration also requires several novel theoretical assumptions, and these assumptions' clarity, verifiability and adaptability are essential to our contribution's credibility and practicality. After finding this integrated model intuitively more justified for many applications than either model alone, we provide a roadmap for further refinement and calibration.

I base the integrated model's validation on prior VDT and PRA justification, taking the information processing model and the PRA risk decomposition methods for granted. Although this can accelerate the process, this research must start at the beginning of the validation sequence because its predictions are critically influenced by fundamentally new components. At its current level of justification, the proposed model relies in part on intuition and external observation to validate its claims.

From a theoretical perspective, the "prior" probabilities and updating procedures that PRA develops in conjunction with domain experts are often the most debated. Therefore, the most troublesome subjects for PRA validation are those domains that are difficult for experts to assess. Many large projects, such as in aerospace or construction, are exceedingly complex, largely unprecedented, and face a legacy of mixed risk management results. The proposed method enhances PRA's justification in this domain by bolstering engineering failure probabilities with VDT's organization-theoretic foundation.

Changing the VDT formulation, calibration variables or code implementation makes comparison with other theoretical results from VDT less viable, reducing the value of results to the academic community. However, certain of the VDT calibration measures may not be accurate for a particular application. In other applications, substantial

extension may be appropriate, for example (provide a list of extension from prior theses). To contribute to this research, we must strive to document changes clearly and make the matrices available. This should include a clear description of the intuitive impact- the real-world interpretation.

To establish face validity we will integrate a simple illustrative example problem into the initial algorithm's definition. This example project illustration will satisfy a first step in our formal justification process [Burton and Levitt], by offering the simplest "Toy Problem" that is able to convey the algorithm's complete essence. This example project will include the five stages we have found to be the most common: specification, design, development, testing, and operations (as illustrated and described occasionally in the existing paper). In practice, many projects involve additional steps, such as decommissioning (after operations). However, as long as the foundational VDT assumptions hold for each stage, the algorithm we propose captures other project forms in a straightforward manner that requires neither mathematical nor theoretical extensions.

The next step in our method's justification will be to conduct a more complex intellective experiment in a later study. One possible application is a comparison between ordinary shuttle missions and the final Columbia and Challenger launches:

> *The four flights scheduled in the five months from October 2003, to February 2004, would have required a processing effort comparable to the effort immediately before the* Challenger *accident. -NASA 2003*

This follow-on study will compare our calculated implications of micro-theories against established macro-level predictions, such as those of NAT or HROT. Finding agreement among our model's components, a broad range of social science results, and the empirical evidence that has sustained those theories, will improve our confidence in our model's soundness.

11.5 EXTENSIONS

Automated Search for Optimal Plans

The proposed model requires that we separately calculate the expected utility for each alternative we wish to consider, and manually compare these results to select the best. As a simulator, the core VDT system offers numerous performance metrics, but has no

model of an integrative objective function. Instead, users who wish to optimize the schedule must conceptualize alternatives and test them iteratively. As noted by Pugnetti [1997, p. 143], there are methods that may help us to overcome this hurdle.

Figure 1 shows how we can build on this approach model using a synthesis of VDT and PRA. This predictive model of a given project's POPE factors and their interactions has the potential to recommend decisions that affect both program and project risks. We claim that a fully developed configuration of VDT, PRA, and an optimization can integrate many important objectives such as design schedule, cost, product quality (variable amount of benefit that is achieved upon success), employee burnout, and team coherence [Chachere 2004.1, Chachere 2004.2].

Figure 10: Project Decision Analysis Method
This schematic diagram illustrates how a project decision analysis model could help project planners to design and optimize project plans, while taking account of the behavior of each project stage. As in the basic VDT-PRA model, users of this system would build design choices into an integrated model of the project's POPE building blocks (Product, Organization, Process, and Environment). The assessment "engine" would predict the impacts of the design choices (the range of diverse and essential data suggested by bullet points in the top boxes) by simulating the project from design through development. For a particular project, the method would predict task schedules, costs and risks, as well as organizational and procedural compliance measures. These output data would inform project managers of theory-founded predictions and enables them to iteratively improve project plans. Finally, the decision analytic element would compare planner preferences with the model predictions for each case, and recommend an optimal choice among POPE factors.

Rather than just predicting outcomes, we can incorporate one of several possible optimization modules to recommend action using a decision maker's preferences. To use decision analysis, we can group compatible POPE configurations into discrete alternatives, and maximize utility over the VDT and PRA predicted outcome

distributions. Alternatively, or in combination with decision analysis, we can use nonlinear optimization methods to recommend continuous-valued choices, as pioneered in APRAM. It is also possible for genetic algorithms [Koza 1992, Holland 1975]to effectively evaluate the large set of highly interdependent organization an process variables, as suggested by Pugnetti 1997, and as demonstrated by Bijan and Levitt () and Bijan, Levitt, and Koza () to be capable of beating human experts.

Modeling Additional Error Sources

There are theoretical predictors of design error rates that this method does not assess. Examples include *conformity*, which decreases the likelihood that individuals will contradict peers' public, erroneous statements [Festinger 1954]. *Groupthink* reduces the likelihood of thorough, critical evaluation of alternatives in a group setting [Janis 1982.2]. Finally, the *risky shift* phenomenon leads groups to select choices that are more risky than those that participants would individually choose [Bem et al 1965]. These additional factors can make it very difficult to isolate the exception-handling impact on risk, and to calibrate the model.

We can capture some of these phenomena using new integration points between PRA and VDT. For example, workers under stress tend to make mistakes [Cooke et al 2003], and so it is reasonable to expect a relationship between the VDT-predicted backlog of engineers, and the number of gross errors that enter operations. To further refine this model, we consider that as work falls behind, a conflict of interest between a personal need to meet schedule and a project's need for design robustness forms, and this moral hazard can produce harmful corner-cutting behavior [Garber].

Predicting Gate and Other Mid-Project Decisions

VDT and PRA are theoretically and methodologically compatible, and pragmatically complementary in several important ways. The model detailed in this paper is just one of three possibly useful ways of integrating PRA, VDT, and Decision Analysis that Chachere 2004.1 presents. Although each synthesized model has distinctive characteristics, they are not mutually exclusive. The same paper also outlines the methods and capabilities of an algorithm that applies all three integration methods simultaneously.

An example shortcoming that further integration points may be able to resolve is that VDT cannot model important mid-project decisions that are made during project execution. Gate decisions critically influence project behavior, for example, and can adjust future stages' POPE structure to account for observed engineering behavior. Other mid-project decisions influence low level features that VDT models. For example, a project's quarterly re-evaluation of staffing levels might increase or reduce full-time-equivalent levels for a particular position. Although VDT does not currently model these mid-course corrections, we can employ a decision analysis engine to estimate the decisions that managers will make at these junctures. In fact, we go further to model actors' rework decisions using the general rational framework [March 1994], and interpret exceptions as a perceived shortcoming in the decision basis that prevents "Clarity of action" [Chachere et al 2004.1, 2004.2, Howard 1992, Howard and Matheson 1993].

Chapter 12 - Conclusion

This research addressed the questions:

1. What are some mechanisms by which engineering design activities create product flaws that later increase the probability of downstream operational failure?
2. What is a method that quantifies the degree to which specific engineering (design and development) phase choices change the operational phase failure probability of a given project?

These issues are important because in spite of prior efforts, engineering-stage errors repeatedly result in the operations-stage loss of space missions, pharmaceutical drugs, facilities, and other large investments. This research puts forward the following claims:

1. Ignoring the need for coordination and rework during engineering phases tends to create product flaws that increase the probability of subsequent operational failures.
2. We have proposed a model that translates VDT predicted engineering phase choices (exception handling and meeting attendance), into PRA estimates of engineering element failure risks (for components, interfaces, and interdependent multi-systems), which informs the calculation of the operations phase failure risk probability of a given project.

These results offer both intuition and precise analytic techniques that are valuable to theorists and practitioners alike. We have identified several exciting implications, such as the potential to analyze the interactions among existing risk theories that are currently defined only in qualitative terms. We have also identified promising next steps, such as the assessment of additional risk factors, the development of project optimization methods, and the advancement of justification for the proposed method.

Chapter 13 - Bibliography

Asch, S. E. (1987, original work published 1952). *Social Psychology*. New York: Oxford University Press.

Bem, D., M. Wallach, and N. Kogan (1965). "Group Decision Under Risk of Aversive Consequences" Journal of Personality and Social Psychology, 1(5), 453-460.

Benjamin, J. and M. E. Paté-Cornell (2004). "Risk Chair for Concurrent Design Engineering: Satellite Swarm Illustration" Journal of Spacecraft and Rockets Vol. 41 No. 1 January-February 2004.

Bergner, D. (2005). Personal communication by an employee of the NASA Ames Research Center's Human and Organizational Risk Management research program.

Brooks, F. (1995). *The Mythical Man-Month* (20th Anniversary Edition), Addison-Wesley.

Carley, K. (1996). "Validating Computational Models" Working paper prepared at Carnegie Mellon University.

Chachere, J. (2004.1) "Methods and Benefits of Integrating The Virtual Design Team with Probabilistic Risk Analysis for Design Project and Program Planning" Working Paper developed during a Management Science and Engineering Ph.D. Tutorial for Elisabeth Paté-Cornell, Stanford University, Stanford, CA

Chachere, J. (2004.2) "Design Project Optimization" Working Paper developed during a Management Science and Engineering Ph.D. Tutorial for Elisabeth Paté-Cornell, Stanford University, Stanford, CA

Chachere, J., J. Kunz, and R. Levitt (2004.1). "Can You Accelerate Your Project Using Extreme Collaboration? A Model Based Analysis" 2004 International Symposium on Collaborative Technologies and Systems; Also available as Center for Integrated Facility Engineering Technical Report T152, Stanford University, Stanford, CA.

Chachere, J., J. Kunz, and R. Levitt (2004.2). "Observation, Theory, and Simulation of Integrated Concurrent Engineering: 1. Grounded Theoretical Factors that Enable Radical Project Acceleration" available as Center for Integrated Facility Engineering Working Paper WP087, Stanford University, Stanford, CA.

Chachere, J., J. Kunz, and R. Levitt (2004.3). "Observation, Theory, and Simulation of Integrated Concurrent Engineering: 2. Risk Analysis Using Formal Models of Radical Project Acceleration" available as Center for Integrated Facility Engineering Working Paper WP088, Stanford University, Stanford, CA.

Chachere, J., Kunz, J., and Levitt, R. (2004.4). "Observation, Theory, and Simulation of Integrated Concurrent Engineering: Grounded Theoretical Factors and Risk Analysis Using Formal Models" forthcoming in Project Risk Management Principles and Practices, Institute of Chartered Financial Analysts of India, Banjara Hills, India.

Ciavarelli, A. (2003). "Organizational Risk Assessment: The Role of Safety Culture" Unpublished manuscript prepared at the Naval Postgraduate School for NASA-Ames Research Center

Cooke, N., J. Gorman, and H. Pedersen (2002). "Toward a Model of Organizational Risk:

Critical Factors at the Team Level" Unpublished manuscript prepared at New Mexico State University and Arizona State University.

Cyert, R., E. Feigenbaum, and J. March (1959) "Models in a Behavioral Theory of the Firm" *Behavioral Science*, Vol. 4, No. 2

Dillon, R. L., and M. E. Paté-Cornell (2001). "APRAM: an advanced programmatic risk analysis method," *International Journal of Technology, Policy, and Management*, Vol. 1, No. 1, pp.47-65

Dillon, R. L., M. E. Paté-Cornell, and S. D. Guikema (2003) "Programmatic Risk Analysis for Critical Engineering Systems Under Tight Resource Constraints," *Operations Research*, May/June.

Feigenbaum, E. (1988). The Rise of the Expert Company, Times Books, New York and Macmillan, London.

Galbraith, J. (1977). *Organization Design*. Reading, MA: Addison-Wesley.

Holland, J. (1975). *Evolution in Natural and Artificial Systems* University of Michigan Press, Ann Arbor, Michigan.

Howard, R. (1992). "Heathens, Heretics and Cults: The Religious Spectrum of Decision Aiding", *Interfaces*, 22, pp 15-27.

Howard, R. and J. Matheson (eds.) (1983). *Readings on the Principles and Applications of Decision Analysis*, Decision Analysis, Strategic Decisions Group, Menlo Park, CA.

Janis, I. (1982.1). *Stress Attitudes, and Decisions: Selected Papers* New York, Praeger Publishers.

Janis, I. (1982.2). *Groupthink: Psychological Studies of Policy Decisions and Fiascoes*, Houghton Mifflin Company, 01 June.

JPL (2004). Personal communication by an anonymous employee of the Jet Propulsion Laboratory's Advance Studies (mission design) program.

JPL Special Review Board (2000) *Report on the Loss of the Mars Polar Lander and Deep Space 2 Missions*, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA.

Jin, Y., R. Levitt, T. Christiansen, and J. Kunz. (1995). "The Virtual Design Team: Modeling Organizational Behavior of Concurrent Design Teams," *International Journal of Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol.9, No.2, (April) 145-158.

Jin, Y. and R. Levitt (1996). "The Virtual Design Team: A Computational Model of Project Organizations" *Computational and Mathematical Organization Theory*, 2(3): 171- 196.

Koza, J., (1992). *On the Programming of Computers by the Means of Natural Selection* MIT Press, Boston, MA

Kranz, G. (2000). *Failure is not an Option: Mission Control from Mercury to Apollo 13 and Beyond* New York: Simon and Schuster

Kunz, J., T. Christiansen, G. Cohen, Y. Jin, and R. Levitt (1998). "The Virtual Design Team: A Computational Simulation Model of Project Organizations," *Communications of the Association for Computing Machinery*, (November) pp.84-92.

Law and Kelton (2000). *Simulation Modeling and Analysis* 3rd Edition, New York: McGraw-Hill.

Lave, C. and J. March (1975). *An Introduction to Models in the Social Sciences*. New York: Harpers and Row.

Levitt, R., J. Thomsen, T. Christiansen, J. Kunz, Y. Jin, and C. Nass (1999). "Simulating Project Work Processes and Organizations: Toward a Micro-Contingency Theory of Organizational Design," *Management Science 45 (11),* November, pp1479-1495.

Luce, R. D. and H. Raiffa (1957). *Games and Decisions: Introduction and Critical Survey*. New York: Wiley. (Reprinted New York: Dover, 1989).

Luce, R. and H. Raiffa (1990). "Utility Theory" In Moser, P.K. (Ed.) *Rationality in Action:*

Contemporary Approaches (pp. 19-40) Cambridge University Press: New York, NY.

Marais, K., N. Dulac, and N. Leveson (2004). "Beyond Normal Accidents and High Reliability Organizations: The Need for an Alternative Approach to Safety in Complex Systems" Working paper prepared at the Massachusetts Institute of Technology.

March, J. and Simon, H. (1958). *Organizations*. New York, John Wiley & Sons, Inc.

March, J. (1994). *A Primer on Decision Making: How Decisions Happen*. New York: Free Press.

Maule, A. and A. Edland (1997). "The Effects of Time Pressure on Human Judgment and Decision Making" in R. Ranyard, W. Crozier, and I. Svenson (Eds.) *Decision Making: Cognitive Models and Explanations* (pp. 189-204) New York: Routledge.

Meshkat, L., and R. Oberto (2004). "Towards a Systems Approach to Risk Considerations for Concurrent Design" Working paper prepared at the Jet Propulsion Laboratory, California Institute of Technology.

Milgrom, P, and J. Roberts (1992). *Economics, Organization & Management* Englewood Cliffs, N.J.: Prentice Hall.

Moder, J. and C. Phillips (1983). *Project Management with CPM, PERT and Precedence Programming* 2nd Edition.

Murphy, D. and M. E. Paté-Cornell (1996). "The SAM Framework: A Systems Analysis Approach to Modeling the Effects of Management on Human Behavior in Risk Analysis", *Risk Analysis*, Vol. 16, No. 4, pp.501-515.

NASA (1995) *NASA Systems Engineering Handbook*, National Aeronautics and Space Administration, Washington, D.C.

NASA Columbia Accident Investigation Board (2003). *Report Volume 1* Government Printing Office, Washington, D.C.

Paté-Cornell, M.E. (1984). "Fault Trees vs. Event Trees in Reliability Analysis", *Risk Analysis*, Vol. 4, No. 3 pp. 177-186

Paté-Cornell, M.E. (1990). "Organizational Aspects of Engineering System Safety: The Case of Offshore Platforms" *Science*, Vol. 250, November 1990, pp. 1210-1217.

Paté-Cornell, M.E. (2004). "Engineering Risk Analysis: Volume 1" Course Reader, Stanford University, Stanford, CA.

Paté-Cornell, M.E., and P. Fischbeck (1993.1). "Probabilistic risk analysis and risk-based priority scale for the tiles of the space shuttle" *Reliability Engineering and System Safety*, Vol. 40, pp. 221-238.

Paté-Cornell, M.E., and P. Fischbeck (1993.2). "PRA as a management tool: organizational factors and risk-based priorities for the maintenance of the tiles of the space shuttle orbiter" *Reliability Engineering and System Safety*, Vol. 40, pp. 239-257.

Paté-Cornell, M.E., D. Murphy, L. Lakats, and D. Gaba (1996). "Patient risk in anaesthesia: Probabilistic risk analysis and management improvements" *Annals of Operations Research*, Vol. 67, pp. 211-233.

Perrow, C. (1984). *Normal Accidents: Living with High-Risk Technologies* New York: Basic Books.

Perrow, C. (1994). "The Limits of Safety: The Enhancement of a Theory of Accidents" *Journal of Contingencies and Crisis Management*, 2, (4), 212-220.

Powell, W. and P. DiMaggio (Eds.) (1991). *The New Institutionalism in Organizational Analysis* Chicago: University of Chicago Press.

Roberts, K. (1990). "Managing High-Reliability Organizations" *California Management Review*, 32, (4), 101-113.

Scott, W. R. (1998). *Organizations: Rational, Natural, and Open Systems* 4th Edition New Jersey: Prentice-Hall.

Sherif, M., O. J. Harvey, B. J. White, W. Hood, and C. Sherif (1961). *Intergroup conflict and cooperation: the robbers cave experiment* Norman, OK University Book Exchange.

Simon, H. (1977). *The New Science of Management Decision* 3rd revised edition (1st edition 1960) Prentice-Hall, Englewood Cliffs, NJ.

Thompson, J. (1967). "Organizations in Action: Social Science Bases in Administrative Theory", McGraw-Hill, New York.

Tversky, A. and D. Kahneman (1974). "Judgment Under Uncertainty: Heuristics & Biases" in D. Kahneman, P. Slovic, &A. Tversky (Eds.) *Judgment Under Uncertainty: Heuristics and Biases*. Cambridge: Cambridge University Press.

Vaughan, D. (1996). *The Challenger Launch Decision* The University of Chicago Press, Chicago, IL.