Testing in NASA Human-Rated Spacecraft Programs: How much is just enough?

by

Keith J. Britton

M.S. Systems Management, Florida Institute of Technology, 1992
B.S. Mechanical Engineering, University of Central Florida, 1983

and

Dawn M. Schaible

M.S. Space Systems Operations, Florida Institute of Technology, 1992
B.S. Mechanical Engineering, Bradley University, 1987

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

Master of Science in Engineering and Management

at the

Massachusetts Institute of Technology

February 2003

Signature of Author _____
Keith J. Britton
System Design and Management Program
February 2003

Signature of Author _____
Dawn M. Schaible
System Design and Management Program
February 2003

Certified by _____
Nancy Leveson
Thesis Supervisor
Professor of Aeronautics and Astronautics

Accepted by _____
Steven D. Eppinger
Co-Director, LFM/SDM
GM LFM Professor of Management Science and Engineering Systems

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

Testing in NASA Human-Rated Spacecraft Programs:
How much is just enough?

by

Keith J. Britton

and

Dawn M. Schaible

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

Master of Science in Engineering and Management

ABSTRACT

Testing has long been recognized as a critical component of spacecraft development activities at the National Aeronautics and Space Administration (NASA). Determining the appropriate amount of testing, however, is a very difficult task. The objectives of this thesis are to document the test approaches and philosophies used within NASA and other organizations, to determine if these factors can be applied to all human-rated spacecraft programs, and to provide lessons learned for future projects. Through a series of expert interviews and review of current literature, a number of themes, findings and recommendations emerged.

Some of the major themes the resulted from expert interviews include:

Subjectivity of test requirements development: Typically, the actual test requirement decision process is not documented as a formal process but relies heavily on the judgment of the decision makers, adding to the overall subjectivity. Due to this subjectivity, testing practices are not consistent across the aerospace industry.

Paradoxical nature of testing: Testing alone does not make a project successful, but it does raise confidence in the likelihood of success. Testing is often regarded as a drain on project resources, rather than a valuable undertaking.

Vulnerability to changes and cutbacks: Most testing occurs late in the development phase. Since the budget and schedule pressures are most keenly felt at the end of a development program, testing becomes vulnerable to changes.

Inadequate attention to testing: Testing, in general, does not receive as much attention as it should. Testing is often overlooked during the early planning phases of a project, in current literature, training programs and academia and in organizational status.

Some of the major findings include:

For some projects, software is considered a stand-alone system: Today's systems are becoming increasingly more reliant on software to operate successfully. Yet for many projects software is not being treated as an integral part of the overall system but as a separate stand-alone system.

While testing practices vary, decision factors do not: Contrary to common belief, there is consistency in the decision factors used by the various decision makers. These common decision factors can be divided into technical (safety, risk and confidence building) and non-technical (resource availability, process, individual decision-making behavior and political/cultural influences). Decision makers must consider each of the sources of uncertainty, especially the effects of reuse, an inexperienced team and potential unexpected emergent behavior.

Current methods of tracking testing costs are not sufficient: Actual cost figures for spacecraft testing programs are very difficult to determine because they are not typically tracked as a discrete line item in spacecraft programs. Life-cycle cost estimating is another area that should be enhanced and used in making test decisions.

Upfront planning is a key to success, but be prepared for change: Early planning in both systems engineering and testing is necessary to manage the complexity of human-rated spacecraft programs. The early involvement of the test organization is also essential in establishing a set of firm test requirements that will be less vulnerable to changes later in the program.

Testing is more of an art than a science: Experience and mentoring are more important than formal training in developing test engineering expertise. In order to maintain the knowledge base and core capabilities, test engineering should be treated as a valid profession with the same prestige level as other engineering disciplines.

A few of the recommendations proposed include:

- Form an agency-wide team to seek out best practices in the test engineering field.
- Enhance the current risk management practices to include an assessment of all decisions making factors that contribute to the overall level of likelihood.
- Establish improved training and mentoring programs for test engineers, perhaps through the creation of a test engineering corps.
- Assign both responsibility and accountability for software to system engineering.

Thesis Supervisor: Nancy Leveson
Title: Professor of Aeronautics and Astronautics

Acknowledgements

We would like to acknowledge the following individuals for their support in making our time at MIT an experience of a lifetime.

First, we would like to thank our families for their patience, encouragement and support - especially Gina, Spencer and Sarah Britton, Carol Schaible and Kimberly Squire. It was their sacrifices, rather than ours, that made it possible for us to be active participants in the System Design and Management Program.

We also want to thank our NASA supervisors, managers, mentors and co-workers. Their understanding, encouragement and cooperation made it possible for us to take full advantage of our time at MIT. In particular, we would like to thank Ken Payne, Tip Talone and Randy Galloway for their mentorship, and Mr. Roy Bridges for his support of the PMDP-ALO Program.

We would also like to acknowledge the expert interviewees that shared their time, experience and knowledge with us. Without their contributions this thesis would not have been possible.

Finally, we would like to thank our advisor, Professor Nancy Leveson. Her patience, guidance and expertise made the development of this thesis a valuable and rich learning experience. In addition, Nancy's support of us, SDM and NASA has served as an inspiration for our future endeavors and is greatly appreciated.

Table Of Contents

List of Figures

List of Tables

Acronyms

DOD	Department of Defense
FAA	Federal Aviation Administration
FPGA	Field Programmable Gate Array
IPT	Integrated Product Team
ISS	International Space Station
IV&V	Independent Verification and Validation
MCO	Mars Climate Orbiter
MEIT	Multi-Element Integration Test
MIT	Massachusetts Institute of Technology
MPIAT	Mars Program Independent Assessment Team
NASA	National Aeronautics and Space Administration
PRA	Probability Risk Assessment
RAC	Risk Assessment Code
RMP	Risk Management Plan
SDM	Systems Design and Management
V&V	Verification and Validation
WIRE	Wide Field Infrared Explorer

Chapter 1: Introduction

1.1 Motivation

"A thorough test and verification program is essential for mission success"

NASA Mars Program Independent Assessment Team Summary Report [1]

The above statement was cited as a lesson learned by the National Aeronautics and Space Administration (NASA) Mars Program Independent Assessment Team (MPIAT) in their March 2000 report. The MPIAT was chartered to review and analyze three successful and three unsuccessful Mars and deep space missions in order to provide recommendations for future NASA projects. This is just one example of a review committee finding that thorough testing is needed to ensure a successful mission. Another example includes the Huygens Probe. While on its way toward Saturn's largest moon Titan in 2000, the Huygens Probe experienced anomalies in its communication system. An enquiry board was assembled to investigate the problems and included in their recommendations that "an end-to-end test should have been performed on the complete system as a final test". [2] Even with these and numerous additional examples that point to the benefit of testing, many programs still cut back on the amount of testing that is performed, only to meet with less than optimal results.

NASA is not the only agency, and aerospace is not the only industry, that has experienced major failures that perhaps could have been prevented if more testing would have been conducted. Just one of a plethora of examples was in the floating rescue capsule aboard the ill-fated Russian submarine Kursk. According to published reports "some of the 118 Russian sailors who died…could have been [saved] if rescue gear aboard the ship had ever been tested". [3] It was also reported that the capsule testing was deleted from the testing program because construction of the ship was falling behind schedule. [4]

If testing is recognized as such a benefit, then why is more testing not performed? Given unlimited time and unlimited budget, more testing would be performed. Since it is unfeasible for programs to perform exhaustive testing, a balance must be struck between too little testing and too much testing. Not enough testing adds risk to a program, while testing too much can be very costly and may add unnecessary run-time on the equipment. Determining exactly how much testing is just enough is an extremely difficult question for many program managers and other decision makers. The decision process often appears arbitrary and has received little attention in the past.

The role of testing in a successful program is to allow an opportunity for errors to be discovered and corrected before the system is put into operation. Testing is performed for risk mitigation; it serves as an insurance policy against failed missions. Even though the importance of testing is generally recognized, relatively little attention has been paid to this area in either the NASA Systems Engineering Handbook [5] or the NASA Program and Project Management Process and Requirements Policy Guide. [6] While the structure of a

test and verification program is addressed in these documents, actual implementation and guidelines on how much testing should be performed are not covered in detail.

Because testing is performed in almost every industry and in all NASA programs, a more narrow focus is required. This thesis will begin to address the challenges of testing NASA's human-rated spacecraft programs. The focus on human-rated spacecraft was chosen due to the inherent criticality of protecting against loss of life. Also of interest is the influence that expendable and human-rated spacecraft philosophies have on each other, and the relationship between the two types of programs. This thesis will address the following concerns that served as motivation for this research:

- Difficulty in striking a balance between too much and too little testing
- Perceived arbitrary nature of the decision-making process
- The vulnerability of testing to cutbacks as a program progresses
- Apparent lack of attention to testing in established NASA program and managerial processes

1.2 Hypotheses and Objectives

This thesis poses several hypotheses. The first is that while most decision makers recognize the importance of testing, they rely on highly subjective decision factors and do not use a holistic approach to determining how much testing should be performed. The next hypothesis is that cost and budget factors tend to be the major drivers in determining the level of testing, but actual life-cycle costs are not adequately addressed in the decision-making process. When life-cycle costs are considered, they are out-weighed by more pressing issues of the moment. Also, when actual life-cycle estimates are obtained, it would be reasonable to believe that testing on the ground would a more cost-effective option than finding and correcting the problem on-orbit. The third hypothesis is that the decision-making process plays a significant role in determining what testing will be performed. Engineers and managers are typically unaware how this process will affect their decisions. The final hypothesis is that organizational factors are important to a test program's success. However this relationship may not be well understood or appreciated by program managers. A summary of the hypotheses is as follows:

- Many decisions makers do not utilize a holistic approach in addressing testing requirements
- Life-cycle costs are not adequately addressed in the decision-making process
- Decision makers are not fully aware of the influences inherent in the decision-making process
- The influence of organizational factors is not fully appreciated

The first objective of this thesis is to document the role of testing, along with the approach and philosophy used within the agency's test programs and across similar industries. From

this research, decision criteria and the critical factors involved in making testing decisions will be determined. A second objective is to determine whether a common set of decision factors can be formed from the various approaches studied. Using this set of factors, this thesis will propose recommendations for architecting future test programs and will address the third objective of providing lessons learned for future projects.

A fourth objective is to understand how various test managers and systems engineers deal with this issue of unexpected emergent behavior in systems. When the overall system's behavior depends upon the interaction between two or more sub-systems, this is referred to as emergent behavior. This emergent behavior is usually anticipated and is necessary for the system to operate successfully. However, interactions between sub-systems can result in unexpected, and often undesirable, system responses. This unexpected emergent behavior can usually be explained after it reveals itself but is not anticipated ahead of time. As Eric Bonabeau (2002) noted, "Because of their very nature, emergent phenomena have been devilishly difficult to analyze, let alone predict". [7] This thesis will attempt to capture current methods of driving out unexpected emergent behavior and provide recommendations for the future.

A summary of the thesis objectives is as follows:

- Document testing approaches and philosophies used and determine decision criteria and critical factors
- Determine if these factors can be universally applied to all programs and propose a framework for architecting test programs
- Provide lessons learned for future projects
- Understand how various programs address the issue of unexpected emergent behavior in systems

1.3 Thesis Approach and Structure

The research methodology used for this thesis includes a review of current literature on the subject of test and verification from system engineering, decision theory, and organizational behavior sources. Since previous research in this area is limited, a series of expert interviews was conducted with NASA, contractor and industry personnel. From these interviews and the literature review, common themes and findings were established, along with recommendations for architecting future test programs. This research will be described in this thesis as follows:

Chapter 2 provides a foundation for understanding the NASA development process and the role of testing in this process. In addition, this chapter explains the basic terminology of testing as described in the literature. This is necessary since each program and industry tend to use the terms for slightly different applications. A common understanding of the terminology is established for the remainder of this thesis. This chapter also briefly

discusses the purpose of testing, test implementation and challenges faced in a test program.

Chapter 3 summarizes the expert interview process. The survey questions are described along with the approach used in conducting the interviews. The data analysis methodology is explained. General themes are identified for detailed review in Chapter 4.

Chapter 4 outlines the common themes that were observed from the interview process and its applicability to the literature review.

Chapter 5 presents the findings derived from the themes described in Chapter 4.

Chapter 6 offers recommendations for architecting future test programs. These recommendations were developed based on the work outlined in the previous chapters and addresses both the technical and non-technical factors that influence the testing decision-making process.

Chapter 7 provides a brief summary of this thesis, makes recommendations for future NASA human-rated spacecraft programs and proposes future work that should be conducted in the area.

Chapter 2: Testing Classifications and Methodologies

"The purpose of testing and evaluation is to determine the true system characteristics and ensure that the system will successfully fulfill its intended mission."

- Blanchard [8]

This chapter discusses the NASA spacecraft development process and the role testing plays toward spacecraft integration and verification. Because testing terminology is not used consistently across different programs and industries, a set of common definitions is established for use in this thesis. The discussion also includes a description of the NASA project cycle including the technical aspects. All of this provides the background for further discussions regarding the stages of the verification process and methods used for verification. Finally, the role of testing is discussed in terms of its contribution to the overall verification process.

2.1 The NASA Development Process

The NASA Systems Engineering Handbook (SP-6105) addresses the program and project life-cycle process and how the technical aspects are incorporated into each phase. In general, the NASA development process is "requirements driven" and uses hierarchical terminology for decomposing a system from its highest level to successively finer layers, down to individual parts. [9] A general hierarchical structure of a system is depicted in Figure 1.

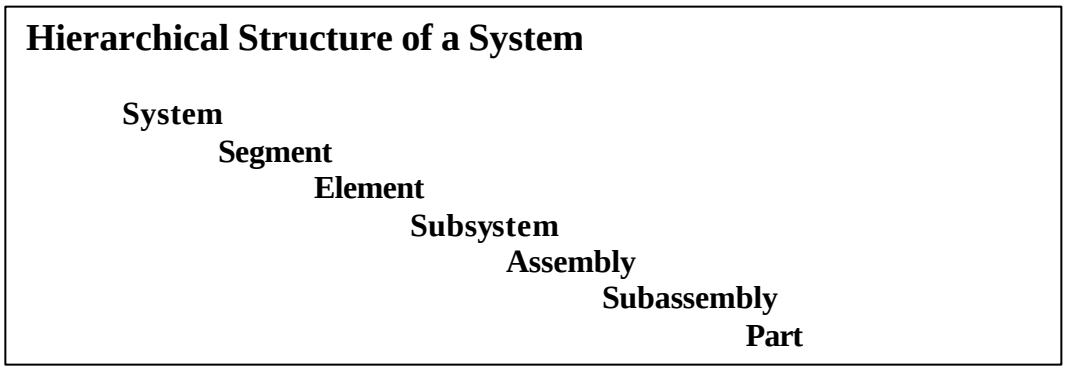

Figure 1 - Hierarchical Structure of a Generic System (Derived from [10])

This structure serves as the framework for systems requirements development that in turn provides the functional requirements that must be verified. Testing is used to accomplish this verification process.

The systems engineering approach encompasses the entire technical effort required to develop, deploy, operate and dispose (decommission) a spacecraft. The objective of the

systems engineering process is to ensure that the system is designed, built, and operated in accordance with its intended purpose and in the most effective way possible, considering cost, schedule and risk.

The NASA program and project life-cycle process identifies various activities for the preliminary design, detailed design, fabrication & integration and preparation for deployment phases. NASA has adapted the generic Forsberg and Mooz "Vee" chart (Figure 2) that describes the technical aspects of a project cycle in terms of the decomposition and definition sequence (referred to as definition) and the integration and verification sequence (referred to as verification). [11] The definition sequence (left side of Vee) defines the process for understanding user requirements, developing system requirements, expanding them to "design-to" specifications and finally, evolving to "build-to" documentation and inspection plans. During each step of the definition sequence a set of verification requirements should be developed. These requirements (sometimes referred to as verification and validation requirements) serve as the framework for ensuring the system will perform as desired in its intended operating environment. The verification sequence (right side of Vee) of the NASA project cycle provides the requirements for inspection of the "build-to" hardware, verification of the "design-to" documentation, integration of the system performance, and demonstration/validation of the complete systems in accordance with user requirements.

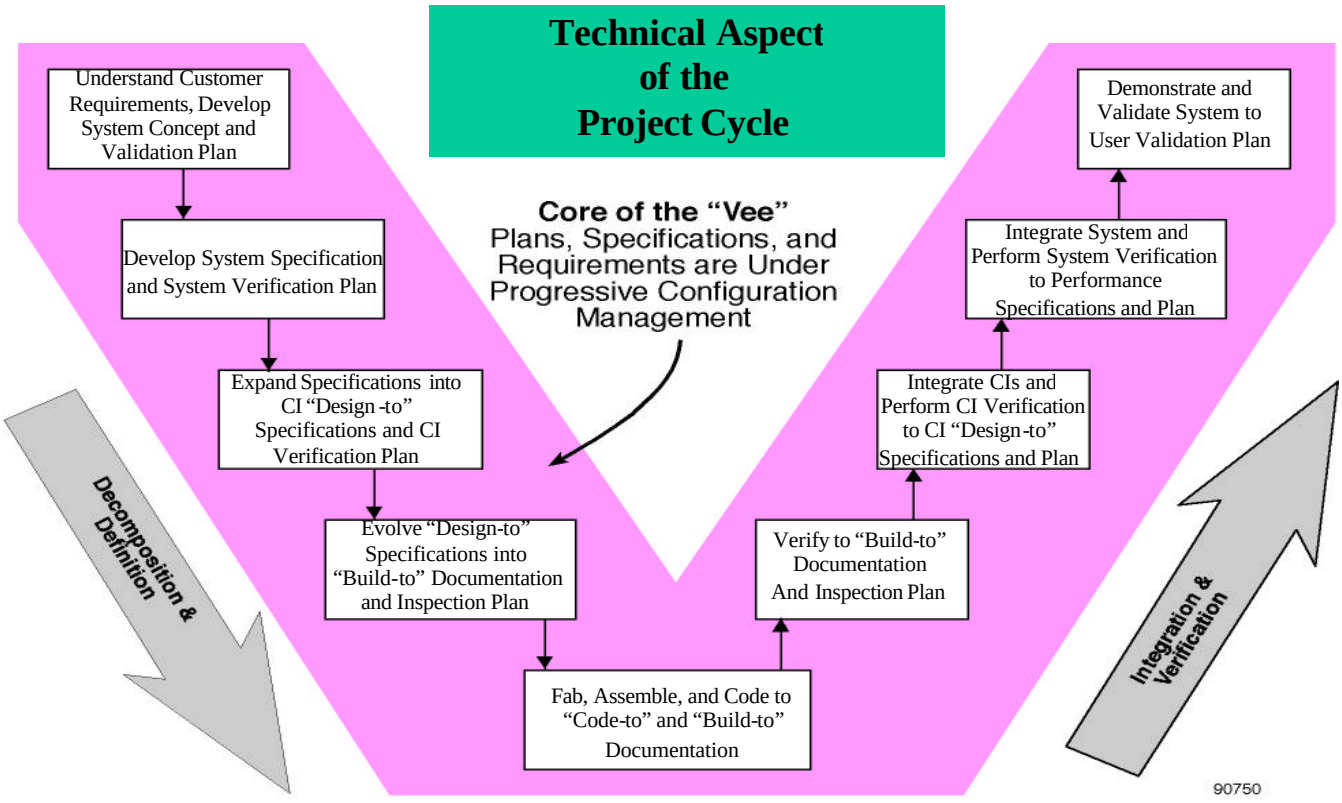

Figure 2 - Generic Overview of the NASA Project Cycle [12]

2.2 Methodologies Used in the Verification Process

There are various methods of accomplishing the objectives of the verification sequence including analysis, similarity, inspection, simulation and testing. These methods provide a level of certainty and knowledge regarding the system behavior and performance, and they verify the activities of the definition sequence. The following paragraphs provide a brief description of the verification alternatives and how those methods meet specific verification requirements.

Analysis uses specific techniques, such as statistics, modeling, or qualitative methods to verify compliance to specification/requirements. Analysis is used in lieu of testing when the methods can be proven to provide rigorous and accurate results and when testing is not cost-effective.

Similarity is a verification technique that uses comparison of similar hardware components. The acceptance data or hardware configuration and application for a particular piece of hardware is compared to one that is identical in design and manufacturing process and has been previously qualified to equivalent or more stringent specifications.

Inspection techniques are used to physically verify design features. This implies a physical inspection but can also refer to documentation. Construction features, workmanship, dimensional features and cleanliness are all applications for inspection.

Simulation is a technique for verification that uses hardware or software other than flight items, but built to behave in an identical manner as the flight systems.

Testing is a method of verification in which the actual equipment is allowed to operate under conditions that demonstrate its ability to perform as specified by the design requirements. Various stages of testing may be required, as the system is configured, in order to verify functionality at each stage of assembly.

2.3 Stages of Verification Process

The verification sequence of the NASA spacecraft development process, as shown in the verification sequence of Figure 2, can be further divided into stages that correspond to defined periods in which different verification goals are met. [13] The six generic verification stages are:

- **Development**
- **Qualification**
- **Acceptance**
- **Pre-launch (Integration)**
- **Operational**
- **Disposal**

For human-rated spacecraft, the role of testing is prevalent in the first four stages and is used less in the operational environment and lesser still in the disposal stage. Therefore, the following paragraphs will focus on those stages that most relate to the testing activities (the development, qualification, acceptance and pre-launch).

2.3.1 Development Stage

Development or component level testing is intended to verify that the lowest level components of a system actually perform their intended functions in accordance with the specified requirement(s). A component can be considered a single "part" or a series of parts configured to perform a function. Component testing occurs at an early stage of systems development in order to provide the building blocks that will comprise the overall system. The data collected during this phase of testing serves as the foundation for system reliability, maintainability and cost effectiveness. Reliability data is usually compiled on each system component according to the mathematical relationships of continuously operated systems.

These reliability equations help determine infant and wear-out failure rates and thus determine the remaining component failure probabilities in a range containing random occurrences. The period of operation in which only random failures occur is considered the useful operating life of the component. This is valuable information because it determines how long a particular component may be used before its replacement becomes necessary. It also defines the amount of higher level testing to which the component can be subjected before the wear-out period is entered. A graphic depiction of this distribution is referred to as a "Bathtub Curve" in which the hazard function is divided into the three regions described in Figure 3.

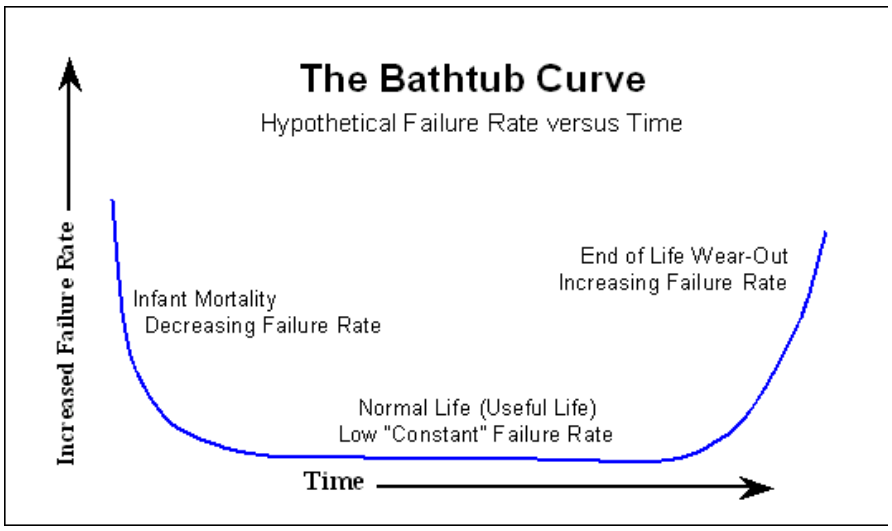

Figure 3 - Typical Depiction of a Weibull Distribution (Bathtub Curve) [14]

2.3.2 Qualification Stage

The qualification stage of spacecraft development is intended to verify that fight hardware meets functional, performance and design requirements. [15] Specific requirements evolve from performance measures that are traceable back to system level requirements and to customer requirements. Verification requirements in this stage are more severe than the conditions expected during operation in order to establish that the hardware will perform satisfactorily in the flight environment with sufficient margins. [16]

Blanchard provides a further description of qualification testing as addressing the need to prioritize requirements according to criticality, in order to allow an integrated test plan to be developed. This places the proper emphasis on the most critical requirements and eliminates redundancies that would increase costs. Ideally, qualification tests should be scheduled as a series of individual tests performed in an integrated manner as one overall test. [17] However, for highly complex spacecraft systems being developed in stages, at geographically separate facilities, or when contractually fragmented, such integration testing is not always possible.

Specific testing objectives that are typically defined in qualification testing include overall performance, structural, environmental (thermal), and vibration testing. Software validation should also be performed during qualification testing but is often performed with simulation or development software that does not represent the final released versions. Validation of the software ensures that the system behaves per the user requirements. Actual verification that the software meets the stated requirements can only be completed after the final flight version is released.

The performance margins established in this stage will determine the amount of system flexibility that can be utilized downstream. Even though each individual component may be within specified tolerance, the aggregate of all the components may result in an out-of-tolerance condition for the system. These system tolerances are a key factor to be considered at this point because neglecting them may result in unexpected emergent properties during system integration or worse yet, discovering an out-of-tolerance condition of the system late in the development phase.

2.3.3 Acceptance Stage

The acceptance stage defines the period in which the delivered spacecraft (end-item) is shown to meet the functional, performance and design requirements as specified by the mission requirements. [18] This stage often concludes with the shipment of the accepted item to the launch site.

Acceptance activities typically occur after the initial qualification testing and before final integration testing of the spacecraft and are intended to evaluate system performance using the actual production components that will constitute the final

spacecraft. Acceptance testing also serves to verify the "workmanship" of the assembled products.

2.3.4 Pre-Launch Stage (Integration)

The pre-launch stage typically begins with the arrival of flight hardware and software at the launch site and ends with the successful launch of the spacecraft. This stage is intended to verify requirements of the integrated spacecraft as well as integration between the spacecraft and launch vehicle or launch facilities. The integration phase allows interfaces between different portions of the spacecraft to be brought together and tested for fit as well as function, thus providing the opportunity to assemble all the spacecraft components in their final configuration. If necessary, the entire system can be run under actual operating conditions. For human-rated spacecraft such as the International Space Station (ISS), integration testing with flight hardware is not feasible because the various ISS elements are in different stages of development. Therefore, integration testing is often comprised of a mixture of actual flight hardware and software with emulators of other parts of the overall system.

For unmanned spacecraft that have no on-orbit repair capability, problems found and corrected in the integration test may be the final determinant for mission success. Although not the intended purpose, integration testing does provide the final opportunity to verify the system meets the intended user requirements and to identify any system-level performance shortfalls. In a perfect world, the integration phase would test interfaces only and would not identify any lower level failures (assuming the previous test phases were performed successfully). In reality, integration testing is often the first system-level checkout of flight hardware and software interfaces. This thesis will focus on the integration test phases.

2.4 The Role of Testing

In general, the purpose of test and evaluation activities is to determine the true system characteristics and to ensure that the system will successfully fulfill its intended mission. [19] Testing serves a dual purpose of verifying fulfillment of requirements while also driving out uncertainty in system behavior due to imperfect requirements (which will always exist).

As discussed, testing is considered one method of meeting the system verification and validation (V&V) goals and usually is performed to address risk and uncertainty that cannot be satisfactorily mitigated by other, less expensive, means (i.e. analysis, simulation, inspection). Spacecraft testing is sometimes considered synonymous with system V&V; however, this view may be misleading. V&V is the process by which systems are evaluated and has a twofold objective: to determine if the system meets the design (verification) and to verify that design performs the intended tasks (validation) and meets the customer's needs. Stated differently, verification is the process of verifying that hardware and software operate according to the specified requirements (i.e. it does what it is asked to do) while validation is the process of determining whether the end product

meets the design and user intentions (i.e. it does what it is suppose to do). As such, system validation activities would be performed at a higher level in the system hierarchy. It should also be noted that while verification establishes that the equipment being tested meets the specified requirements, it does not evaluate the validity of the requirement. Therefore, special emphasis should be placed in the development of the requirements to ensure they represent the actual intended system operation.

Chapter 3: Research Methodology and Data Analysis

"The beginning of knowledge is the discovery of something we do not understand."

- Frank Herbert [20]

3.1 Data Collection through Interviews

In order to capture expert knowledge regarding integration testing of human-rated spacecraft, a series of interviews with recommended experts in the testing discipline was conducted. The interview process was intended to solicit the tacit knowledge of the interviewee and was not structured to obtained data for statistical analysis. A total of fourteen experts were interviewed. Nine of these experts were from NASA while five represented either government contractors or the aerospace industry. The number of interviewees represented a broad spectrum of the testing community (from test engineers to test program managers, both from government agencies and contractors), but was not deemed large enough to constitute a valid statistical data population. From the interview process additional data was sought that would provide a statistical foundation for verification of the findings and recommendations presented in the testing framework. In fact, the statistical data was not forthcoming in sufficient quality or quantity to be useful. The lack of statistical data, however, does not deter from the qualitative value of the knowledge obtained through the interviews and subsequent response analysis.

3.2 Interview Methodology

The interview methodology consisted of private sessions with each individual (or in some cases two individuals). Interviewees were selected from several sources, recommendations from Massachusetts Institute of Technology (MIT) professors and advisors, reputation, personal knowledge of the researchers, associations with the MIT Systems Design and Management (SDM) Program and recommendations from the actual expert interviewees. The sessions were conducted in person wherever possible or otherwise via telephone and typically lasted approximately one hour.

In general, the interviews consisted of the two researchers asking questions regarding one of four focus areas of testing. The interviewees were provided an opportunity to describe their backgrounds and basic philosophies regarding system testing and systems engineering in general. This introduction period allowed the researchers to tailor the questions toward a specific area of interest or expertise the interviewee may have possessed. For instance, a test engineer may have described their expertise in environmental (thermal) testing of hardware and how certain barriers existed that made conducting these tests difficult. In this case the engineer would be allowed to elaborate on that specific aspect of testing (component and qualification level). The questions were focused on their experiences, perceptions and challenges, based on their background. Often the interview included a discussion on philosophies regarding spacecraft testing programs and then flowed into a discussion on the strengths and weaknesses of the approach in question. The interviewee was provided ample time to respond and then,

through follow-up questions, asked to elaborate on the previous response. Confidentially was assured so that the expert interviewees could provide frank responses without consideration of consequences and to protect any proprietary knowledge regarding testing processes and procedures or financial information.

3.3 Interview Categories

The interview questionnaire consisted of five sections, an introduction and biography section for the interviewee, and four sections on specific focus areas regarding spacecraft testing. These areas include the role of testing, decision factors that influence how test programs are implemented, cost factors that influence test programs and organizational issues relating to test programs. The actual interview questionnaire is contained in Appendix A.

Section I, a biographical section, was required to allow the researchers and interviewees to become familiar with each other. This section provided contact information, prior experience and current responsibilities. The information was also used for demographic analysis regarding coverage of the spacecraft-testing spectrum.

Section II of the questionnaire focused on the role of testing and requirements definition. This section identified the type of testing (i.e. qualification, acceptance, integration) that the interviewee was most familiar with and how that particular testing phase adds value to the overall spacecraft development process. Questions regarding the criteria and processes used to define testing requirements and how these are coordinated with the actual hardware and software development processes were also discussed. Finally, several questions regarding decision and cost factors were included to serve as transition points to follow-on sections.

Section III of the questionnaire focused on factors that are considered in deciding what tests to perform on a spacecraft. Internal and external factors were sought as well as how much weight specific factors carry in influencing the final test requirements. This section also introduced questions regarding the integration of hardware and software in human-rated spacecraft test programs and the implications of treating software differently from hardware and in delaying the integration of the two. Factors that influence decisions to modify test programs were also sought. In this context, the researchers explored subjective influences such as the political environment and the effects they play on maintaining or changing planned testing programs. The final objective of this section was to explore the value of identifying unexpected emergent properties in human-rated spacecraft through testing. This was an important issue to the researchers because of the perceived tendency for test programs to verify known requirements and not seek to identify unexpected behaviors prior to deployment.

Section IV, budget and cost considerations, provided a transition from the decision-making factors because these considerations serve as a separate influence on the level of testing performed on human-rated spacecraft. The focus of the cost-related questions was to determine how well test program costs are planned, budgeted, monitored and reviewed

upon completion of the program. Rules of thumb and other budgetary guidelines were sought for use in planning and implementing various phases of a human-rated spacecraft test program. In addition, the influence of cost factors on the decision process was discussed.

The final section, Section V, of the questionnaire focused on organizational factors that influenced the success of test programs. Comparisons were made between actual test program organizational structures against those that the experts deemed most effective. Questions also addressed organizational considerations given to software versus hardware testing and the integration of the two. This section also sought out insights on how well knowledge is captured, retained and transferred to future programs. Finally, the effectiveness of documenting lessons learned and transferring them within the current organization, and to other programs, was discussed.

After all the questionnaire topics were covered, a period of open dialog was provided in which the expert could elaborate further on any topic previously discussed and offer more insights on other aspects of testing not included in the questionnaire. These discussions were captured under a generic category called "Other" and served as the conclusion of the interview process.

3.4 Interview Questions

As previously mentioned, the questions used in the interview process were designed to draw out the tacit knowledge of the individuals as opposed to a verification of their formal knowledge regarding human-rated spacecraft testing or systems testing in general. The sequence of the questions was intended to begin with basic concepts and progress into more abstract areas that are not widely covered in academia or formally considered in systems testing documentation.

The interview questionnaire contained a total of fifty-two questions (Appendix A). Some questions contained follow-up queries while others sought informational data only. Forty-five separate questions related to expert knowledge of the individual while another seven were for informational purposes. Although all questions were covered in the interview process, not all were sufficiently answered by the respondents. This reflects the wide spectrum of experience sought in the research. Finally, several questions contained overlapping themes that allowed the interviewee to answer them simultaneously, if they so choose. For instance, question (20) asks about cost factors for determining the extent in which testing is performed. As the response was delivered and elaborated upon, it was often possible to segue to question (31), which addressed testing budgets. Therefore the questionnaire was used as a guide rather than a formal process.

3.5 Goal of Interviews

Each of the four major sections of the questionnaire correlated with specific areas of focus regarding human-rated spacecraft. The interviews were intended to capture knowledge of experts regarding current test practices within NASA programs as well as the aerospace

industry in general. In concert with capturing knowledge regarding current testing practices, recommendations for improving on those practices, within the four focus areas, were also sought. The body of knowledge compiled from the interviews served as the foundation for further comparison to system engineering principles. A second goal of the interview process was to gather enough responses from a diverse spectrum of testing experts to allow identification of common themes that apply to system testing in general and human-rated spacecraft testing specifically. Finally, the interviews attempted to capture best practices, as well as deficiencies, in organizing and implementing programs. This goal is not specifically oriented toward test programs but toward an overall spacecraft program organization that provides the best chance of success.

3.6 Overview of Interviews and Method of Data Analysis

The interview process was conducted with both researchers present. Responses were recorded individually by the researchers and later compared for consistency. The responses were then compiled electronically into a larger data set. The data set was built according to the focus sections in the questionnaire. All responses were analyzed and common themes were drawn from each section through a process of grouping like responses from different interviewees together.

A system of maintaining anonymity was established that masked the responses but allowed traceability back to the specific expert interviewees. The final data set of responses consisted of over 500 separate items that were used in the analysis. A distribution of responses to expert interviewee function is depicted in Table 1.

	INTERVIEW SECTIONS/FOCUS AREAS				
Function	**II - Role**	**III - Factors**	**IV - Cost**	**V - Org/KM**	**VI - Other**
Test Manager	75	71	21	60	10
Test Engineer	22	17	3	23	7
Systems Manager	16	20	1	4	0
Systems Engineer	39	39	4	16	3
Independent Verification & Validation	9	15	4	7	4
End User	10	3	0	12	0
Totals	171	165	33	122	24

Table 1 - Mapping of Expert Interviewee Responses

The process for evaluating the data consisted of categorizing each response individually according to the focus section being reviewed. After all four sections were completed, a second phase of grouping the responses according to common themes was conducted. This phase also identified themes that applied across each section. Those crossover themes were also compiled and addressed in the final analysis.

Development of the common themes from the interviews was accomplished from the perspectives of the researchers' testing experiences. An internal perspective was developed from the researcher with extensive experience in human-rated spacecraft testing while an external perspective was developed from the researcher with limited testing experience. These differing approaches to the interview responses provided a deeper understanding of the knowledge captured and resulted in a broad set of recommendations for future test programs. Finally, after the interview data analysis was completed, a collection of overall themes and insights emerged. These along with the literature information were used to develop the common themes and findings described in Chapters 4 and 5.

Chapter 4: Themes Resulting from Expert Interviews

"It All Depends"

- Expert Interviewee [21]

4.1 Introduction

Upon completion of the interview process, as described in the previous chapter, a set of general themes was developed from the compilation of the expert responses. These themes represent the most commonly held opinions and perceptions among the expert interviewees based on their testing experience. This chapter will address five primary themes that were the most prevalent topics from the aggregate of the expert responses. General insights and findings derived from the interviews are presented in the next chapter.

The primary themes that emerged from the interviews are:

- Subjectivity of test requirement development
- Paradoxical nature of testing
- Vulnerability to changes and cutbacks
- Inadequate attention to testing
- Organizational influence on the testing process

Each of these is now discussed further.

4.2 Primary Themes

The primary themes represent a common set of knowledge derived from the expert interviews. These themes, along with literature research, serve as a foundation for the development of findings and recommendations presented in later chapters.

4.2.1 Subjectivity of Test Requirement Development

Although not specifically asked in the interview process, the subjective nature of testing was mentioned by almost all of the expert interviewees. The essence of this subjectivity comes partly from the risk identification and mitigation process and partly from the systems requirements process. As mentioned in Chapter 2, testing serves as risk mitigation. Risk is traditionally discussed in terms of both the likelihood of a specific outcome occurring and the consequences that would result. [22] Because determining risk levels is highly subjective, especially for new designs and architectures, testing will in turn be highly subjective. It is also impossible to perfectly

relate risk to testing since so many additional factors apply, such as the ability to recreate the operating environment on the ground. [23]

In system requirements development, the system architect must answer the question "is this what the customer wants"? Once answered, the requirements for the system can be established. Test requirements flow from these system requirements and attempt to answer the question " how do I prove it"? [24] Because each system is different and has different system requirements, the test requirements will be different as well. These differences in system and test requirements make it difficult to generalize testing.

One additional reason that testing is very subjective is the nature in which test requirements decisions are made. There currently are no concrete rules on how much testing must be performed. The decision is a trade-off between available resources to perform the testing and the amount of risk the program is willing to accept. These factors change with the situation, and when added to the biases of the individual decision maker, a quantitative answer is not likely. [25] Some even report that a "well-guided gut instinct works about as well as anything else". [26] The subjective nature of testing is a significant factor in every facet of a test program, as will be demonstrated throughout this thesis.

Due to the subjectivity, testing practices are not consistent across the aerospace industry or in comparison with the commercial aircraft industry. These inconsistencies can be observed in every phase of the test program. Most notably is in the inconsistent way in which terms are used to describe test activities. This lack of standardization in terms required the need to establish definitions for use in this thesis (Chapter 2). Another source of inconsistency is the wide variation in the way projects are organized, within and across programs and companies. A third inconsistency is in the process of defining, implementing and tracking test requirements. All programs treat testing differently due to unique requirements and conditions. Interestingly, most expert interviewees would like to see some level of standardization to minimize the subjectivity but they are very skeptical that this standardization could ever be successfully accomplished.

4.2.2 Paradoxical Nature of Testing

Another prevalent theme throughout the expert interview process was the paradoxical nature of testing. Intuitively, people know that more testing is better than less testing. With unlimited time and budget, testing would never be an issue because the program would simply test everything. In reality, cost and schedule constraints are always present. The program success is not only based on the successful operation of the system but also on whether it was completed on time and within budget. Testing is often considered an insurance policy against future failures. Program management must perform a risk-benefit analysis to determine the wisest use of program resources. [27]

Even on the technical side, what constitutes a successful test is itself a paradox. A hardware test that uncovers multiple problems can be considered just as successful as a test that verifies the flawless operation of a perfectly designed and built system. Conversely, for software it is easy to write test cases that will be successful, however, these cases will only be effective if they show how the system does not work. Both hardware and software testing serve the purpose of risk mitigation towards ensuring a correctly operating system. This paradox creates a managerial view of testing as an unrecoverable program cost rather than a value proposition of long-term program cost-avoidance. True costs for testing are not calculated in terms of cost prevention, only in terms of actual costs to perform the test. Hence, the cost savings of identifying an error on the ground and fixing it before launch is often not a consideration in determining a test program's success.

4.2.3 Vulnerability to Changes and Cutbacks

Adding to the subjectivity and paradoxical nature of testing is the fact that most testing occurs late in the development phase. The prevalence of late testing leads to the next common theme: testing is vulnerable to changes and cutbacks.

The first source of change comes from incomplete test requirements. Very often the actual design itself is in flux and the system requirements are continuously changing. This in turn results in a changing test program. The higher the fidelity of the early test requirements, the more stable the testing requirements and planning will be at the end of the development phase. [28] Also, as new knowledge is gained about system operation, new methods of testing may be developed and synergies between stages of testing can be incorporated into the test plan.

As budget and schedule pressures increase during the development phase, managers tend to accept more risks and are therefore willing to agree to more cutbacks in testing. Again, the more stringent the requirements, the less vulnerable the test program will be to these reductions. [29] Since the budget and schedule pressures are most keenly felt at the end of a development program, testing is extremely vulnerable because it is one of the only remaining opportunities left to make up schedule or cut costs. Some test programs are budgeted early in a program and the budget is placed in a reserve status. However, if proper care is not taken to preserve the funding, the test budget can be decimated by the time the testing is scheduled to begin. Even fiscal year considerations may change the timing of a test program. One interviewee reported that testing can be delayed until the start of the fiscal year when new funding became available. [30]

4.2.4 Inadequate Attention to Testing

The fact that testing, as an activity as well as an organization, does not receive the same attention as other elements of a program was another common theme of the expert interviews. In general, testing is recognized as an important part of a project. Yet as mentioned earlier, because testing typically occurs at the end of development, it does not receive as much emphasis during the initial planning phases as it should. The

inadequate attention can be in the form of time, budget, or forethought spent on the various activities. Understandably, design has to take precedence in defining the function and form of the system in the beginning of development. It is only after the program is well underway that testing begins to receive the consideration it deserves. If consideration to testing is not given at the beginning of a program, it may be too late to ensure the most efficient and thorough test planning and implementation. Many of the interviewees expressed frustration with this trend. It was common to hear how test requirements were not finalized soon enough. One expert summed it up as "not enough emphasis is placed early enough on high-quality test requirements, equipment, and procedures. We make it work but its not efficient". [31]

Another concern within the testing arena that was repeatedly mentioned in the expert interviews is a lack of training and mentoring. Formal training in testing is not typically provided. The skills needed to be a good test engineer are learned on-the-job but these skills are not proactively maintained or developed. Within NASA, the standard program processes and procedures do not address the detailed implementation strategies for testing. Much time and effort has been put into other areas of program and project planning, but testing has not received its fair share. According to some of the expert interviewees, the lack of a detailed documented testing strategy has meant that each program, or new program participant, has had to relearn important lessons from the past. The experts recommended that more should be done to capture these best practices in order to provide decision makers with some guidelines for test planning. [32] The subjective nature of testing, along with a concern about the ability to generalize the knowledge, may be possible reasons that these best practices have not been documented in the past.

Also emphasized during the expert interviews was the desire to perform test planning in parallel, but one step out of phase, with the development of the hardware and software. As one piece of the design process is completed, test planning should take place. The same is true for actual test implementation. Several of the expert interviewees suggested that the deficiencies in test planning might be an organizational issue. Not enough forethought tends to be given to how a program will be structured to allow for the most effective test planning and implementation. Many of the expert interviewees reported that test engineers do not hold the same status or standing that other engineers do within the program. These test engineers are often not as respected as other engineers and as such, do not receive the same opportunities or receive as much attention. Other divisions within the program and/or organization tend to receive more prominence than testing. This inequality leads to the next common theme found in the expert interviews: the effect of organizational structure on test program success.

4.2.5 Organizational Influences on the Testing Process

A number of the expert interviewees, especially the contractor representatives, described the benefits of having a separate testing organization that can concentrate on this one area of the development process. However, an independent testing organization does have limitations and poses new challenges that must be addressed.

This section will focus on these organizational issues, from the point of view of the expert interviewees.

One observation made from the interviews is that there was not a consistent interpretation of what was meant by "test organization". For some, the test organization includes the actual technicians that perform the operations on the flight and ground equipment. For others, the test organization is those individuals in a program office that are responsible for overseeing test activities. For the purpose of the interviews, and for this thesis, "test organization" will refer to those engineers accountable for implementing test requirements and the program management functions responsible for these activities.

A prevalent assertion encountered during the interview process was the need to develop test-engineering skills. Having a separate test organization was recommended as one method of developing skills, facilitating knowledge transfer and improving communication. This independent organization also allows for a consistent test philosophy to be established and implemented. It ensures that attention is focused on testing activities even while the program management may be concentrating on other issues. To receive the maximum benefit, the test organization should be involved in the very early stages of design and planning.

Another benefit of having a separate pool of test engineers is the ability to be flexible with the resources. Flexibility is particularly important in the component, qualification and integration testing stages of development. During component and qualification testing, the schedule is constantly changing and resource loading must always be adjusted. A pool of test engineers can be an efficient use of engineering resources. [33] For integration testing, a dedicated test team is important because the test engineers need to be are familiar with the overall system operation. Because these same individuals have been involved with the earlier testing, they will have developed an understanding of the operation of the system and any idiosyncrasies that may be present. [34] Having previous experience with the hardware and software can be very important in recognizing system trends and unexpected emergent behavior.

Some organizations provide a mixture of a dedicated test team, along with other independent reviews. As an example, the commercial aircraft industry uses a separate test group for highly critical systems. The Federal Aviation Administration (FAA) has mandated that independent testers and inspectors verify all critical requirements in the commercial airline industry. [35] For lower criticality systems, two developers verify each other's work.

While having a separate testing organization can be beneficial, it cannot operate in a vacuum. Good communication is essential, especially between test engineering, system engineering, software/hardware designers and operations personnel. Many expert interviewees emphasized the need to establish good relationships early and to maintain the interfaces throughout the project. Very often adversarial relationships are formed

between the different groups, severely hampering communication and negatively affecting the development process.

There are other limitations of having a separate test organization. Besides the increased need for good communication, comprehensive integration is also required. This need is magnified with geographical distance and contractual differences between designers, testers and system integration organizations. Having a separate test organization can actually detract from the effectiveness of the project and create inconsistencies rather than standardization when these geographic distances or contractual differences are present. Extra care is needed to provide strong system engineering and integration between the various groups when extreme organizational differences are present.

The need for a strong functional organization to maintain skills was also cited in the interviews. Having a strong functional organization is important for both a dedicated test teams, as well as for Integrated Product Teams (IPT). A number of contractor expert interviewees reported success using the IPT structure. Keys to its success included a strong functional organization to support the test representatives and strong integration between the individual IPTs. [36] Other interviewees described unsatisfactory experiences with the IPTs. A lack of a strong functional foundation and weak integration were possible explanations for the poor execution of the IPT structure. It should also be noted that even for those organizations that utilized IPTs, each was implemented in a different manner. These findings imply the need for better understanding of effective IPT structuring. A further description of an IPT structure is provided in the next chapter.

During the expert interviews, questions about the NASA software Independent Verification and Validation (IV&V) process were asked. The responses provided a unique, and useful, insight into NASA's test philosophies and practices. The following discussion was not utilized in the development of the overall findings because it is not technically considered a testing organization. However, parallels can be drawn between IV&V and testing. Because this information is deemed to be valuable, it is included in this chapter.

While the IV&V organization does not perform actual testing of software products, it does provide an independent review and monitoring of the processes used in test requirement development, test conduct and data analysis. NASA has criteria for determining when a particular project is required to employ IV&V. [37] The programs themselves are responsible for proactively performing a self-assessment and arranging for an IV&V review. The researchers found little evidence, however, that this process of self-assessment was well understood outside the IV&V community. In fact, the criteria used for performing the self-assessment was not clearly defined in program management processes, nor was it readily available for use by program managers.

The IV&V community does police programs to ensure proper use of the IV&V review. One factor hampering their ability to assess the need to perform IV&V on NASA projects is the difficulty in compiling a complete list of all projects currently underway.

A hit-or-miss record of involving IV&V in software development has resulted from the unavailability of a complete list of NASA projects. [38] Fortunately, IV&V has been involved in human-rated spacecraft development, most likely due to the limited number and high visibility of the programs.

Ironically, each individual project must finance the IV&V effort. There was evidence that some projects do not want to have an IV&V audit performed. This resistance is often a result of not understanding the value added in performing an independent review. Often projects see this review as a drain on resources, both in time and money. These resources are usually needed for other efforts that are perceived to be more valuable. From this standpoint, IV&V and testing suffer from the same barriers, biased perceptions and lack of appreciation.

Chapter 5: Findings Derived from the Expert Interviews

"Test Early and Often"

- Expert Interviewee [39]

5.1 Introduction

The expert interviews, along with the literature review, yielded some significant findings, which are described in this chapter. These findings are meant to shed light on some the challenges that the testing community currently faces as well as to provide the foundation for the recommendations described in Chapter 6. With over 500 individual responses collected and analyzed from the expert interviews, a great deal of information regarding human-rated spacecraft testing was captured. For this thesis, the most significant findings are discussed in detail. These findings include:

- For some projects, software is considered a stand-alone system
- Optimism varies with organizational position
- While testing practices vary, decision factors do not
- Upfront planning is a key to success but be prepared for change
- Current methods of tracking testing costs are not sufficient
- Testing is more of an art than a science
- Good sub-system engineering is not a substitute for proper system engineering
- Program/Agency culture strongly affects testing

5.2 Stand-alone Software System

Today's systems are becoming increasing more reliant on software to operate successfully. However, there is evidence that for some projects, software is not being treated as an integral part of the overall system. Instead, software is viewed as separate stand-alone system, posing a number of challenges and concerns, especially for the testing activities. Because the system hardware and software are usually highly interdependent, they must be considered as one integrated system. Very often both the hardware and software must be present for the system to operate. To truly verify the system requirements, the actual flight hardware must be integrated and tested with the flight software. Some of the evidence that points to the premise that software is being treated as a system of itself is provided by the organizational structure, the process by which the software is tested and delivered, and the limited insight that sub-system engineers have into the software development process.

A human-rated spacecraft project's organizational structure is one indicator of how software is handled in general. Some projects and programs use Integrated Product Teams (IPTs) with software being an integral part of each IPT. An IPT is an organizational structure that consists of cross-functional teams dedicated to an individual product, project

or system. Each team includes representatives from the various functional departments. The IPT structure is used to increase communication and coordination between the different disciplines involved in a common system. [40] A simplified example of an IPT is shown in Figure 4.

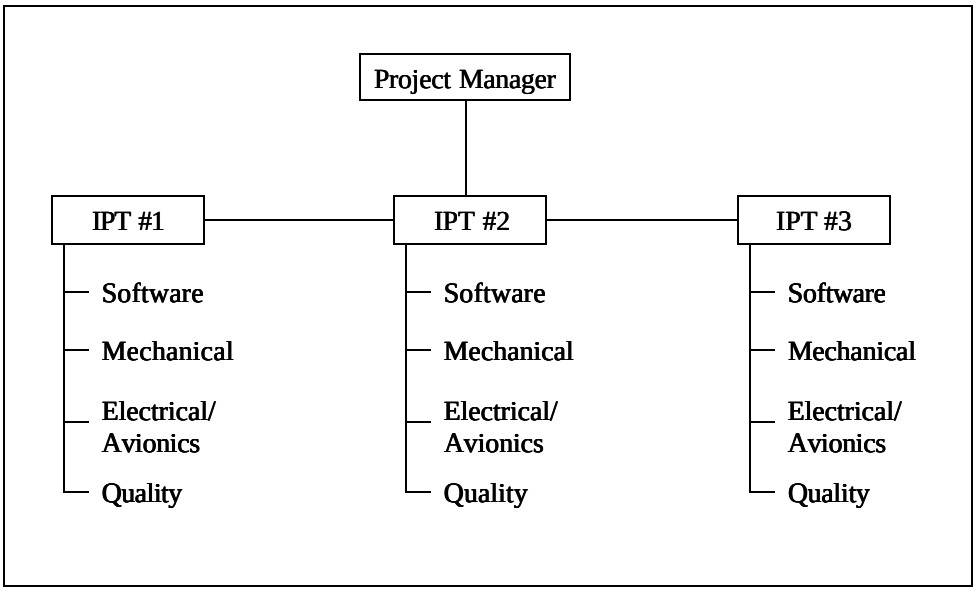

Figure 4 – Simplified Integrated Product Team Structure

On the other hand, some projects have segregated all software engineering into a separate organization. At the extreme, the separate software group is housed in an entirely different division from the systems engineers (as shown in Figure 5). This organizational distance also serves to disassociate software from the rest of the system and limits the sub-system engineer's visibility into the software development process. In one interview, a system engineering expert admitted that sub-system engineers do not typically see the software specifications or the test procedures. [41] This lack of visibility goes completely against the philosophy of true systems engineering.

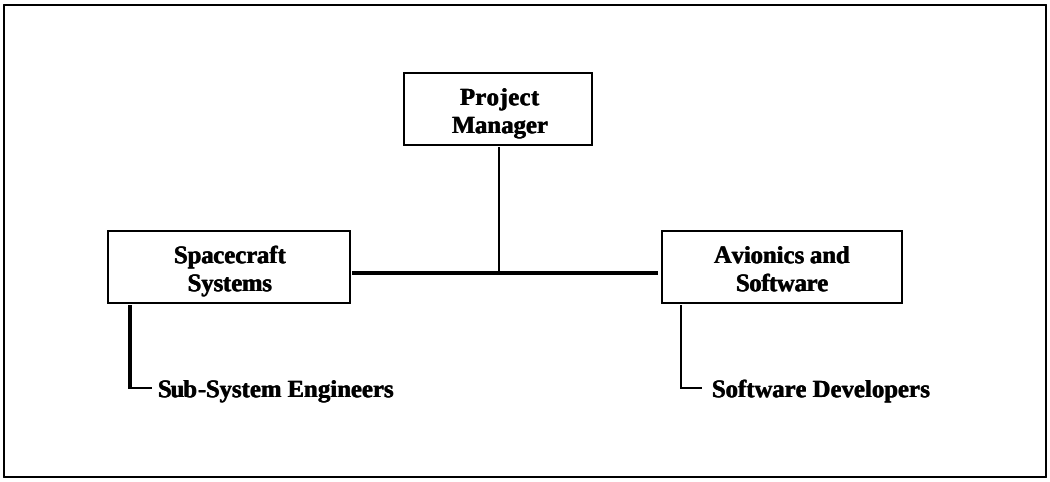

Figure 5 – Simplified Organizational Structure with Segregated Software Group

Another organizational concern with the software development process is the contractual structure for breaking down the software segments. Some program split the responsibility of software development between contractors. When the split also involves a substantial geographic distance (as is common), effective communication is hampered. In addition, the development of the hardware and software for the same system may also be parceled out to different companies. The software product then becomes a program deliverable in and of itself, providing a further indication that for some programs, software is handled as a distinct element. The inclination to treat software separately is not unique to NASA. US Department of Defense (DoD) standards have actually mandated this separation as recently as 1997 (Figure 6). [42]

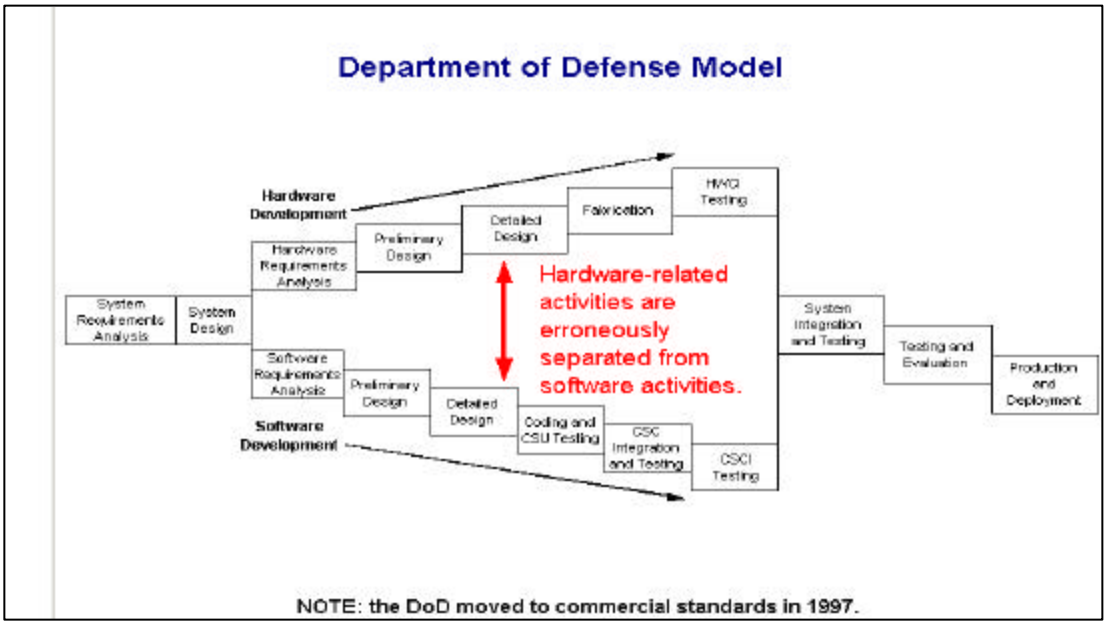

Figure 6 - DoD Model for System Development [43]

Once delivered, the process for testing the software differs from that of the hardware. Some differences should be anticipated because software is not a tangible element and cannot be tested to failure as is done with hardware. Because there is not a physical embodiment of the system, the tester must rely completely on the quality of the specification in order to verify the intent. In addition, the same building block approach for testing may not be applicable to software, as it is with hardware. While the software may be developed in discrete modules, it often cannot be tested as a standalone module.

There are lessons from hardware testing that should be incorporated into software testing. One example relates to retest. Typically when hardware is replaced after a test has been conducted, the retest is typically quite intensive. Not only are the interfaces retested but in many cases, the integration tests are also performed again. The tendency in software is to just retest the module in question and then to plug it back into the system without an integrated test. [44] This is one indication of the popular (but false) perception that software is easier, and safer, to change than hardware. In fact, this is not true. According to Leveson (Safeware, 1995):

> "Changes to software *are* easy to make. Unfortunately, making changes without introducing errors is extremely difficult. And, just as for hardware, the software must be completely reverified and recertified every time a change is made, at what may be an enormous cost. In addition, software quickly becomes more 'brittle' as changes are made – the difficulty of making a change without introducing errors may increase over the lifetime of the software" [45]

The system engineer must be an integral part of this process of evaluating the changes and subsequent implementation in order to maximize the chances of success.

One theory as to why software is treated separately from hardware may be the history of the program and the background of test managers themselves. In general, there is not a good understanding of software engineering by those not in this profession. Most managers currently have a hardware engineering background. They may not truly appreciate the unique aspects of software development and testing. Many of these managers try to apply to software the same hardware paradigm that worked so well for them in the past. Software, however, has many fundamental differences in the way it is developed and tested. As mentioned in the previous chapter, testing is inherently subjective, but the abstract nature of software makes the testing of software even more subjective. When software is treated as a separate system, compounded with this added subjectivity, stronger integration and greater attention to details becomes critical. Unfortunately, the temptation to delegate the decisions to the software community is also greatest under these circumstances.

5.3 Varying Attitudes Toward Test Effectiveness

An insight that resulted from the interview process was the observed differences in attitude toward the effectiveness of testing among those at various levels of the organization. It was

evident that the lower level system and test engineers were much more pessimistic and felt that too much risk was being taken by not testing enough. They did not have a clear understanding of the process that the decision makers went through, and they were the most likely to see the process as arbitrary. Our research showed that the optimism level clearly increased as one went up the chain-of-command toward the managers. Most managers were proud of what had been accomplished and were confident that their test programs were highly successful.

There are several possible reasons for why an individual's outlook on testing changes with their organizational position. The first is an individual's ability to take a holistic view of the entire project. Senior managers have a larger view than many engineers at the lower levels. The lower-level system and test engineers are primarily concerned with their own system and its successful operation. Managers on the other hand must balance the entire system, along with the external influences. They are the ones that understand the political factors involved with the project. In turn, the program manager's risk aversion level is different from that of the lower-level engineers. The program managers must make the trade-offs and, as such, are perceived to have a higher tolerance for risk. These project managers do understand, however, that they are responsible for accepting risk as part of the trade-off decisions. As one project manager stated "I was willing to take the risk of not testing what I flew. As the project manager for the …mission, I was the one who ultimately decided what risks to take…". [46]

Another possible reason for the differing level of optimism is that the understanding of the detailed technical information is most likely not consistent across the organization. It is reasonable to think that the lower-level engineers understand the operation of their system better than the managers. Information may not be conveyed to managers in a way that allows proper risk evaluation. When this less-than-perfect knowledge is combined with the other external pressures, managers may be accepting more risk than they think.

The established culture also plays a role in the differences in optimism. NASA has historically had a culture of risk taking but at times has also been accused of being too risk adverse. Depending on the culture of a particular project, acceptable risk level changes. In addition, while the current NASA culture is very accustomed to fixing technical problems as soon as they are discovered, the process for reversing bad decisions made early in a program has not yet been institutionalized. It should be noted that program and test managers do not make bad decisions on purpose or out of incompetence. Early in a program, all the information may not be available or may not be accurate. Managers make the best decisions they can at the time but these decisions are rarely re-evaluated. The lower-level engineers may recognize these as bad decisions, but the managers may not even be aware there is an issue. The tendency is to just work around the issue, rather than change the ground rules. This tendency is not unique to human-rated spacecraft or to NASA but can be observed in many industries.

5.4 Test Program Decision Factors

Because of the subjectivity and inconsistency of test programs, many interviewees thought that parallels could not be drawn across programs. In contrast, we found that there is consistency in the decision factors used by the various decision makers. What does changes are the decisions that are made based on these same factors. Each decision maker takes a slightly different approach to making test requirement decisions. Some rely on the inputs of the system engineers more than others. Some have a formalized thought process, while others rely more on instinct. When probed further, there is a common set of factors that are included in the decision process. These factors are not all equally weighted nor does the weighting remain constant throughout the process. Situational and other external factors do have a large influence. Perhaps the biggest variable is in the differences between the decision makers themselves.

5.4.1 Technical Factors

Three technical decision-making factors have been identified that deal directly with the spacecraft systems –safety, risk and confidence level. These technical factors are now described further.

5.4.1.1 Safety

It was clear through the expert interviews that safety is the first priority when making testing decisions. NASA's Program and Project Management Processes and Requirements document defines safety as:

> "Freedom from those conditions that can cause death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment" [47]

Any items that could pose a threat to the safety of the astronauts are weighted the highest. The next concern was the safety of the spacecraft and the people working on it.

Safety is never intentionally compromised in the decision-making process. This is not to say that safety in itself is never compromised. The failure of the Mars missions mentioned earlier in this thesis is evidence of this unintended consequence. Had more, or better, testing been performed, it is likely the missions would not have failed. A difficulty that program managers face is understanding what conditions constitute a threat to safety. While every program manager knows that safety is the first priority, they may not know when their decisions actually compromise safety.

Another component of safety that factors into decisions is that of ground operations. Some tests pose a serious safety concern when performed during integration testing, in particular, when they require the use of hazardous materials. In these cases, the project manager must weigh the benefits of performing the test in actual flight

conditions versus the possibility of the ground team being exposed to hazards during operations. Again, it appears that safety does weigh very heavy in this decision process. All other means are explored first and performing hazardous operations on the ground are only conducted when absolutely necessary.

5.4.1.2 Risk

Beyond safety, other aspects of risk were a major factor in the decision-making process. NASA's Program and Project Management Processes and Requirements document (NPG 7120) defines risk as:

> "The combination of (1) the probability (qualitative or quantitative) that a program or project will experience an undesired event such as cost overrun, schedule slippage, safety mishap, compromise of security, or failure to achieve a needed technological breakthrough; and (2) the consequences, impact, or severity of the undesired event were it to occur." [48]

The greater the risk, whether in the form of likelihood or consequence, the more testing will typically be performed. The risk assessment, and minimally acceptable risk, depends on the particular project and the circumstances. There are often pre-conceived biases of the likelihood and consequences are for particular projects. These pre-conceived opinions have led to some general misconceptions. For example, in one interview, the expert denoted that human-rated and expendable spacecraft must be treated differently. Human-tended spacecraft have the ability to make repairs on orbit while satellites do not have the same opportunity. [49] In reality, the same factors apply to both types of spacecraft but the only difference is in the respective consequences. In addition, it is often believed that human-tended spacecraft can accept more risk of finding non-critical errors on-orbit due to the ability of repairing. This does not address the fundamental issue of the likelihood that the error can be detected or corrected once discovered. The ability to detect and correct a failure must be included in determining the consequence level. Each project must assess the relative risk but the resulting actions for testing will differ based on the circumstances.

5.4.1.3 Building Confidence

Another prime factor that may not be as obvious to the decision maker is the amount of uncertainty present. Uncertainty defines how well the risk can be determined. It is based on the amount of system knowledge that is available at the time of the decision. For example, the amount of previous experience with a design or the amount of testing previously completed contributes to the overall system knowledge. As system knowledge increases, confidence in the system's operation also increases. There are a number of other factors that also contribute to the overall level of confidence in a system. The sources of uncertainty in the system's operation are outlined in Table 2. It is important to note that it is these factors that are often

overlooked or underestimated in the decision-making process. By not considering these factors, many program managers have encountered difficulties.

Sources of Uncertainty	
- New design	- New technology
- New interfaces	- New application for reuse
- Previous testing	- Reliability of information
- Hardware/software maturity (design and build)	- Team experience
- Complexity of system (including number of interfaces/interactions)	- Required fidelity of results
- Possible unexpected emergent behavior	

Table 2 - Sources of Uncertainty

Most managers and decision makers recognize that new designs, technology, and interfaces create additional uncertainty, while previous testing increases confidence for the project. However, one of the most under-appreciated sources of uncertainty is reusing existing components in new applications. Decision makers repeatedly assume that reuse of a component of hardware or software will reduce the uncertainty level of the system. Unfortunately this is often not the case. The new application and new interfaces may in fact increase the uncertainty, unless enough time and effort is invested in a thorough review and analysis. One example of the perils of software reuse was experienced on the maiden flight of the Ariane 5 launcher in June 1996. [50] Approximately 40 seconds into the flight, the launcher veered off course sharply, began to break up and then exploded. An investigation into the accident revealed the cause of the failure to lie in the Inertial Reference System. At approximately 37 seconds into flight, the backup inertial reference system became inoperative when an internal variable that refers to the horizontal velocity exceeded a specified limit within the software. Simultaneously, the active inertial reference system experienced the same error, because both were operating with the same software. At this point the internal reference system fed diagnostic information back to the main computer. The main computer interpreted the diagnostic data as flight data and adjusted accordingly, sending the launcher off course. The resulting forces created by the drastic correction caused the launcher to disintegrate and then self-destruct.

The software used in the Ariane 5 launcher was the same inertial reference system software that successfully operated in the Ariane 4. The team chose to reuse the software for the new launcher, even though all the features were not needed in the new application. The team failed to recognized ahead of time the differences between the Ariane 4 and Ariane 5 initial acceleration and trajectory. The Ariane 4 launcher would have never reached the internal horizontal velocity limit within the software. However, the Ariane 5 reached, and exceeded, the limit by design. The team did not appreciate the impact of reusing the software and their review process was not stringent enough to catch the problems that would be caused by the internal reference system sending bad data to the main computer. In addition, since the inertial reference system was used previously on the Ariane 4, it was not tested under the Ariane 5 flight conditions. This test could have caught the error prior to launch. [51]

NASA's Mars Climate Orbiter (MCO) also experienced a failure due to a reuse of software without a thorough understanding of the implications. The MCO reused software code originally developed for another spacecraft for the thruster trajectory equation. The conversion code was in British units but the specification called for metric. The code was obscure and the specification was not well understood or reviewed with the reuse application. The subsequent mismatch of units resulted in the loss of the MCO. [52]

These examples (and many other losses resulting from the reuse of software) demonstrate a possible downside of reuse; decision makers may be lulled into a false sense of security due to success in the past. There was a general impression among the expert interviewees that as they gained more experience with the systems, their confidence level increased. There is a potential that this confidence could translate into rationale to cut back on planned testing. Sometimes such reductions may be the appropriate action but a thorough review and analysis must take place first. Appropriate decision-making must take into account that similar components are not guaranteed to behave identically.

An additional source of uncertainty includes the overall experience level of the team. If the project team includes a large number of relatively inexperienced engineers, there is an increased chance that an error will be made in the development process. Project managers cannot necessarily rely on good engineering practices to ensure the proper design and manufacture of the spacecraft systems. Additional testing is warranted to mitigate the effects of an inexperienced team.

Most of the interviewees recognized unexpected emergent behavior as a source of uncertainty in the system's operation yet few could describe a standard process for driving out this behavior. Most system and test engineers suggested additional testing in off-nominal and maximum use conditions as a means of discovering unexpected behavior. A number of the managers, however, believed that their current test programs were sufficient to drive out this uncertainty. Consciously

addressing any potential unexpected emergent properties is important and the test program should be designed to uncover as much of this behavior as possible.

The remaining sources of uncertainty include: the maturity of the hardware and software, the complexity of the system, the reliability of the information available to the decision maker and the required fidelity of the results. These factors are recognized and understood by the decision makers but may not be a conscious consideration during the decision-making process. In order to achieve the best possible results, decision makers should think through each of the sources of uncertainty before making a final decision.

5.4.2 Non-Technical Factors

Just as strong of an influence can come from external or non-technical factors. Even though they occur outside of the system, non-technical factors can apply a great deal of pressure on the system or on the decision maker. These factors are included in Table 3 and are discussed further in the following sections.

Non-Technical Factors	
Available Resources	**Process**
- Budget	- Requirements
- Schedule	- Test strategy
- Test Equipment	- Hardware/software development
- Facility Readiness	- Contract type
- Ability to test	- Organizational structure
- Simulators/emulators	
- Procedures	
- Personnel resources	
- Limited life (lifecycle)	**Political/Cultural Environment**
	- Changing priorities
Individual Decision Making Behavior	- Nature of relationships
- Knowledge	- Optimism/pessimism
- Subjectivity	- International implications
- Biases	- Funding Source

Table 3 – Non-Technical Factors

5.4.2.1 Resource Availability

Testing decisions regarding resource availability involve cost, schedule, equipment, facility readiness and personnel. While resource issues are addressed during the planning process for testing, they often become subject to changing events that are

unrelated to the specific program under development. Therefore, in reaction to external events, adjustments to resources often appear as the best and only alternative. Consideration of resource alternatives in response to individual circumstances, while ignoring other related factors, can have a detrimental effect on a test program. For instance, delays in testing schedule are often made to accommodate the needs of other programs. An approach that accepts delays without concessions often results in increased costs to the program incurring the delay. Another example of the need to address resource availability in a holistic manner is illustrated by the various resource alternatives such as simulators, emulators and test equipment. By addressing the total resources available for accomplishing test activities, managers can often decide to substitute a planned test with an acceptable alternative using a different approach, thus preventing cost and schedule impacts.

5.4.2.2 Process

Processes have an important role in ensuring successful implementation of test programs. Decisions regarding the testing strategy, contract approach, process rigidity and organizational structure must be considered early in the program in order to ensure proper resource availability later in the program. Perhaps the most important of the process-related factors involves test requirement development and implementation. Ambiguity in requirements often has the down-stream effect of increased changes during test implementation phase, which often results in cost and schedule impacts. Contract strategy is another important process decision that should be given careful consideration early in a test program. While fixed-price contracts offer cost and schedule fidelity, they often do not adequately address changes nor are they flexible enough to add new work without significant cost impacts.

Test implementation involves a number of processes such as change management, configuration control and test procedure development and performance. The formality and stringency of these processes will affect the success of the overall test program. For example, decision-making processes for testing are established to deal with changes in testing requirements. A rigid process will increase the chances that testing decisions will be made in accordance with the established guidelines. Conversely, a process that is not firm will allow subjective considerations to heavily influence decisions. Because processes are used as a means to accomplish the tasks of a particular project, they should be tailored to the needs of the each project. In addition, changing processes once a test program has begun is extremely difficult without incurring heavy impacts on other aspects of the program, namely cost and schedule. Changes to the established processes should not be undertaken without serious consideration or unless absolutely necessary. It is much more efficient to ensure that the required processes are established during the planning phases of a project.

5.4.2.3 Individual Decision-Making Behavior

The ability of the decision maker to gather and interpret information regarding test decisions is rarely addressed while planning or evaluating test programs. Because it is accepted that decision makers achieve their positions through technical and managerial competence, there is usually no mechanism for reviewing the quality of previous decisions as part of the process for making current decisions. Factors such as the decision maker's technical and managerial ability, individual biases, and the subjectivity of the information all influence the quality of their testing decisions. Of these factors, decision biases are least likely to be controlled because sufficient oversight or independent review of the decisions is rarely provided. Therefore, any framework for decision-making should attempt to minimize biases of the decision maker. For instance, a manager with a background in hardware development might consider software testing decisions in the context of their previous hardware testing experience, while another manager with a project management background might emphasize cost and schedule considerations equally to those of technical requirements. Finally, it should be noted that decisions are only as good as the information that is available. Therefore, quantifiable data that can be compared in equivalent terms is essential for any testing decision to be effective.

5.4.2.4 Political/Cultural Environment

Political and cultural factors have the most under-appreciated effects on testing programs. While they often have significant influence on decisions later in the development process, they are rarely consciously considered in the early decision-making process. Both political and cultural factors are important, yet they often go unnoticed by all but those in the highest positions, such as program managers and other key decision makers. Therefore, the influence of political and cultural factors may be greater than the other decision factors on those decision makers that are keenly aware of the political and cultural climate. For NASA, political factors involve pressures from competing programs as well as from organizations outside of the agency, such as Congress. The increasing international involvement in NASA programs also provides a new dimension to the cultural differences and political pressures that project managers must consider.

It should be noted that decision makers might weigh these non-technical factors heavier than technical factors because they are often the pressures that are more readily felt at the time of the decision. Much of the risk component deals with "what-ifs" in the future but resource availability is a present concern for the decision makers. The near term nature of the non-technical factors, along with the vagueness of the long-term technical effects, creates an imbalance for the decision maker.

5.4.3 Retest Considerations

Another finding from the expert interviews was the general perception that retest decisions did not receive the same rigor as original test requirement decisions. When a

problem was uncovered during a test and subsequently repaired, the system often was not retested as thoroughly as the initial test. The decision to not retest is not necessarily improper, if a thorough review and evaluation is performed. Indications from the expert interviews point to the fact that this review does not always take place. The purpose of the retest is to verify the reconnected interfaces, to confirm the corrective action was sufficient to remove the error and to ensure that no other errors were introduced into the system during the replacement process. The same decision factors used in assessing the initial test requirements must be considered for retest.

5.5 The Role of Planning

Human-rated spacecraft have always been extremely complex systems. As more electronic components and software are introduced into the functional operations of the system, the complexity will continue to increase, requiring that more rather than less testing be performed on the spacecraft. To manage this complexity, early planning in both systems engineering and testing is necessary. Statements such as "test early and often" and "poor decisions made early will kill you later" [53] were common responses during the expert interviews.

The need for more planning of test activities is also recognized by organizations outside NASA, such as the commercial aircraft industry. For aircraft, the Federal Aviation Administration (FAA) establishes rigid requirements for testing that must be addressed early in the life-cycle of the program. These requirements are established by regulation and cannot be reduced. However, for unique spacecraft with no regulatory requirements to guide test development, it very difficult to establish a firm program for testing. Often the system requirements are so unique there is no previous test history to help determine the best approach, or in many cases, the time between programs hinders the knowledge transfer process. Furthermore, new designs are susceptible to changes, which translates into uncertainty because the engineer is unsure of the final system configuration. Ambiguity in system requirements can make the test requirements unstable and may result in more required testing.

The success of a testing program is only as good as the test requirements that are developed. However, early emphasis on high quality test requirements is often missing from program planning decisions. The expert interviews revealed that inadequate requirements could lead to poor test procedures and a lack of proper test equipment. [54] The experts also advised that getting an early start on philosophical and policy decisions is necessary to establish the right priorities regarding the level of testing and criticality level of the requirements. [55]

An illustration of the importance of test requirements can be drawn from the Huygens Probe that was launched with the Cassini Spacecraft in 1997. The probe experienced a communications anomaly during a routine test while en-route to its destination (the Saturn moon of Titan). The unexpected behavior occurred when a simulated Doppler shift was applied to a signal being sent to the Huygens Probe Support Avionic receiver. The Doppler shift caused the data signal to fall outside of the receiver's narrow-band bit-loop

detector. The enquiry board could find no evidence of requirements or specifications regarding Doppler Shift on subcarrier or data rate in the receiver radio frequency telemetry. Had this been a requirement and tested prior to launch, the problem could have been identified on the ground and corrected by a software change. The software was not changeable after launch. Although the Huygens Program made use of independent tests and verifications, the following conclusion was drawn: "The problem on Huygens has shown the value of an end-to-end test simulating the various mission conditions as close as possible during ground testing." [56]

In the case of Huygens, early involvement of test engineers may have been able to influence the decision not to perform the end-to-end testing. One role of the test engineer is to find system problems that the designers did not anticipate. The added evaluation by the test engineers and a subsequent end-to-end test would have helped to ensure the spacecraft's robustness. Finally, through early involvement, the test organization would be more familiar with decisions made by the designers and therefore would have the necessary understanding to develop stringent test requirements that would validate the actual mission conditions. These same findings can be applied to all programs.

The early involvement of the test organization is also essential in establishing a set of firm test requirements that will be less vulnerable to changes later in the program. By establishing critical test requirements that must be performed, program managers and test managers are in a better position to respond to program changes. Less critical requirements can also be established early in the program, but can be accomplished by other methods of verification, or not performed in favor of cost or schedule considerations. This is appropriate as long as the risks associated with not performing the test are acceptable.

Early decisions regarding the testing approach to human-rated spacecraft are made at the program management and system engineering levels. The program manager is responsible for overall resource allocation and cross-system integration decisions but must consider recommendations from the systems engineers. The systems engineers in turn base their recommendations on the inputs of the various hardware and software owners. For human-rated spacecraft, this simplified decision hierarchy involves many organizations and groups assigned to various aspects of the program. Decision makers at all levels must consider a wide range of variables when addressing testing programs. These include when to perform tests, what tests to perform, where to test, fidelity of tests, the amount of software IV&V to perform and how much retest should be allocated as contingency.

Upfront planning also allows for the opportunity to begin testing early. An advantage of early testing is to verify component behavior under system-level conditions. Gaining this knowledge of actual system operations and behaviors is useful in driving down the level of uncertainty in the system. This testing allows transients to be identified at an early stage. The experts regarded this early emphasis as necessary to begin the process of addressing the "little glitches", or unexpected emergent behavior, that must be understood as part of the overall system is developed. Ignoring these small details can lead to maintenance issues during operation, or even worse, on-orbit system failures.

The process of identifying unexpected emergent properties of human-rated spacecraft must begin at an early stage in the program to ensure that as many as possible are recognized and addressed. Analytical methods can be applied to early systems to find and correct possible interactions before the design is complete. Analysis alone will not identify all unexpected emergent behaviors and testing is necessary in order to ensure that those interactions that do present themselves will not pose an unacceptable situation for the spacecraft or crew.

An example of this kind of unexpected behavior occurred on the Wide Field Infrared Explorer (WIRE) experiment launch in March of 1999. After its launch, a system anomaly occurred in which the telescope aperture cover was opened prematurely, resulting in the release of the spacecraft's cryogenic hydrogen. The subsequent report on the incident traces the behavior to a field-programmable gate array (FPGA) that was used in a circuit for which it was not well suited. The mishap investigation determined that a transient signal occurred after start up of the FPGA. The WIRE report indicated that the testing method used for the spacecraft was performed at the hardware-box level only, a method that would not have identified the transient. The report also stressed the point that the spacecraft should be "tested as it is going to be used" in order to identify these types of behaviors. [57]

Allowing enough time in the testing schedule to address these unexpected emergent behaviors, and other contingencies, is also important. Because testing occurs at the end of the development phase, the remaining available schedule is tightly controlled. The objectives of later tests may become vulnerable if too much time is spent correcting errors uncovered by the early test procedures. While schedules are typically optimistic, testing will find errors and the planning should account for these contingencies.

Finally, early planning also allows program managers to develop enough flexibility in the test program to make adjustments during the later phases. Often, windows of opportunity develop from changes in other parts of the program (i.e. delays in launch schedule) that create time for more testing. If a prioritization of additional test activities is not readily available, these opportunities can be lost. The International Space Station (ISS) Program provides a prime example of taking advantage of a window of opportunity. The original ISS baseline contained almost no integrated testing of the major elements. When a schedule opportunity became available, ISS program management was able to use the time to perform a series of Multi-Element Integration Tests (MEIT). [58] These tests have proven to be successful in finding and correcting problems prior to launch.

5.6 Tracking Testing Costs

Actual cost figures for spacecraft testing programs are very difficult to determine because NASA does not typically track them as a discrete line item in spacecraft programs. Pure test costs are often obscured by engineering and operational funding, making them susceptible to misinterpretation. An effort was made to determine the availability of actual cost data for use in a case study for this thesis, but what we found was either cited as

proprietary or deemed to be in an unusable format for proper analysis. Therefore, it is recommended that actual financial data be obtained for use in evaluating testing cost performance and for use in program planning.

Test costs are not consistently defined from one program to another, making comparisons difficult between programs. Contractors, however, have attempted to establish heuristics for estimating testing costs for use in developing cost proposals. These estimating "rules of thumb" are subject to negotiation during the proposal development phase as well as during final contract negotiations with the customer. During the actual performance of testing activities, contractors also provide better cost tracking, but do not typically make the details of these costs available to others for proprietary reasons. Contractors are often required to submit detailed financial information to the government, but again this information is not easily broken down to actual test costs.

Better estimating and tracking of test activities is needed to establish a consistent estimating tool that can be used in early program planning. Accurate test cost data can also be used to establish the savings that testing brings to a program in the form of cost avoidance from errors that may have been overlooked had the testing been eliminated. Life-cycle cost consideration is another deficient area in making test decisions. For human-rated spacecraft it is often attractive to forego a test and accept the risk that an on-orbit repair will be necessary. However, when the cost of on-orbit repairs are considered in the decision, it often becomes apparent that performing a test is money well spent in the long run. The trade-off between performing testing and accepting the risk of an expensive on-orbit repair is often not assessed because of the urgent nature of the decision at hand and the lack of useful and accurate cost data. The manager will be tempted to "save" the cost of the test and apply it toward a possible repair effort. With improved cost data regarding testing and on-orbit activities, decisions to perform more testing may be easier to justify.

5.7 Testing Is An Art

The finding that testing is more of an art than a science further reinforces the theme of subjectivity in testing and also recognizes the innate difficulty of test engineering. Experience and mentoring are more important than formal training in developing test engineering expertise. One reason for the lower prominence of formal training may be the lack of testing emphasis in systems engineering literature, including government and corporate policies and procedures. Although NASA is attempting to address testing as part of its program and project management policies, our expert interviews revealed that documented guidelines may be one or two revisions away from giving testing adequate attention. [59]

The lack of testing literature is not limited to NASA. College curriculums and industry also give testing superficial emphasis. Testing is mentioned as a necessary component of system development but does not include a discussion of actual test implementation. Formal training is also not provided either in college or in most industries.

Because testing practices are not formally documented, the most valuable knowledge is retained within the individual experts as tacit knowledge. This finding is reflected by the expert interview responses that recognize the human relationship component that exists in the test engineering community. As organizations change from experienced-based knowledge to process-based knowledge, there is a danger of losing the testing expertise that resides within the organization. Once this knowledge is lost, the chances of repeating past mistakes increases.

In order to maintain the knowledge base and core capabilities, test engineering should be treated as a valid profession with the same prestige level as other engineering disciplines. Providing test engineers with a viable career path will prevent them from moving to other, more rewarding, assignments and will also serve to attract talented new engineers to the profession. Currently, test engineering is not given the recognition appropriate to the amount of creativity and difficulty involved as compared to design engineering. The expert interviews pointed out that test engineers need to know the system design and then must be more creative in trying to identify system faults. [60] Finally, any organizational structure should attempt to capture and retain the core knowledge of its testing group. Mentoring and On-the-Job-Training are two ways of retaining testing knowledge and passing it along to new programs.

Barriers to knowledge transfer exist in several forms. First, most knowledge management systems are poorly funded and fail to document root causes of problems in a form usable for application toward new problems. Second, a culture of addressing only those problems related to the current program exists, without regard to preventing problems in later programs. Contractual issues also pose a barrier to knowledge transfer in the form of test data created by contractors not being a required deliverable to the government. Finally, knowledge transfer is relationship dependent. Programs that have good relationships between organizations tend to transfer knowledge easier than those that have adversarial relationships or are geographically dispersed.

5.8 The Importance of System Engineering

Good sub-system engineering is not a substitute for proper overall system engineering. Designing and testing at the sub-system level is a necessary process that attempts to confine functionality within the sub-system while providing higher-level controls over the interfaces between the various sub-systems. In doing so, the complexity of the higher-level system can be managed through the lower-level sub-systems. Focusing on the sub-system level does not imply, however, that early attention should not be applied to the overall system level performance and the possible interactions between the sub-systems. Our expert interviews suggested that another role of testing is to capture those sub-system interactions without having to give them thoughtful attention during the early development stages. While testing does drive out errors, it is not intended to serve as a substitute for the design process (from a systems integration perspective). Nor should a conservative design process that attempts to provide system robustness and fault-tolerance serve as a substitute for proper testing. The bottom line is that system engineering must give a voice to both design and testing in the proper proportions.

The need for a strong system engineering perspective is becoming increasingly more important as systems become more complex and the projects more distributed. Many of NASA's human-rated spacecraft projects are broken down into individual work packages and dispersed between centers and contractors. In general, good system engineering is performed for each of these packages. When the project is broken up in this manner, the system engineering must be even stronger. More coordination and communication is required and only through the system engineering function can this be accomplished. Very often each of the work package owners follows their own similar, yet different, processes. The system engineers must ensure some level of consistency and that a minimum standard is met. Integration testing, by its very nature, crosses these organizational, contractual and geographic boundaries. Proper system engineering is essential to the success of a test program.

In very large programs, individual work package groups or organizations may be tempted to operate as autonomous entities. They can become focused on delivering their particular product and lose focus on the larger picture. It is the responsibility of system engineering to ensure that working relationships are established and a free flow of information is maintained. Several expert interviewees noted that these individual organizations sometimes exhibit "engineering arrogance". They do not recognize the need for system engineering and they do not think outside their particular area of expertise. As one interviewee described, these groups operate as if they were on separate islands. System engineering must build the bridges to connect each island. [61]

5.9 Program/Agency Culture

Program and agency culture strongly affects testing. A traditional engineering culture places emphasis on design because the design must come before testing. In addition, the role of the designer is to build a robust system while the role of the tester is to find errors and shortcomings with the design. Both groups must be disciplined enough to recognize the role of the other and avoid adversarial behavior. As mentioned earlier, test engineering has a stigma associated with it due to its lower priority within the organization. This inequality in organizational status exacerbates the adversarial relationship and can hamper communications between the design and test groups, resulting in severe negative effects on program success.

Another cultural aspect that affects testing programs is the mindset that each program stands on its own. Very little resources are spent on trying to make testing easier on the next program by understanding today's failures and sharing lessons learned. The repetitive nature of the tasks in the commercial aircraft industry makes the transfer of lessons learned easier. Even so, for most engineering cultures, the process of mentoring is much more effective in transferring knowledge than through documentation methods.

Chapter 6: Lessons for the Future

"Taking action in advance to prevent or deal with disasters is usually worthwhile, since the costs of doing so are inconsequential when measured against the losses that may ensue if no action is taken"

- Nancy Leveson (Safeware, 1995) [62]

From the themes and findings discussed previously in this thesis, recommendations for assessing future test requirement decisions are outlined in this chapter. First, the current risk management process is described, along with its applicability to test requirement decisions. Suggestions for improvement of the risk management process are also offered. Second, additional keys to a successful test program are discussed. Finally, this chapter explores the nuances and challenges faced in the individual decision-making process.

6.1 Risk Management Process

6.1.1 Current Risk Management Techniques

As discussed in the preceding chapters, testing serves the purpose of risk mitigation by discovering errors and building confidence in the overall system operation. All NASA programs, along with other industries, currently use risk management programs that have been tailored to their own individual needs, constraints and philosophies. Each program utilizes some form of likelihood and consequence as a means to assess risk. NASA defines likelihood as: "the probability that an identified risk event will occur." [63] Consequence is defined as "an assessment of the worst credible potential result(s) of a risk". [64] In order to assess the overall risk, likelihood and consequence must be evaluated together. One tool used to aid decision makers in this evaluation is a risk matrix. The generic risk matrix that NASA has adopted is shown in Figure 7.

	LIKELIHOOD ESTIMATE				
CONSEQUENCE CLASS	**Likely to Occur**	**Probably will Occur**	**May Occur**	**Unlikely to Occur**	**Improbable**
Catastrophic					
Critical					
Moderate					
Negligible					

High Risk
Medium Risk
Low Risk

Figure 7 - NASA Generic Risk Matrix [65]

The matrix in Figure 7 illustrates only one option for comparing risk. While this matrix has four levels of consequence and 5 levels of likelihood, each program and industry has tailored the matrix to suit their own needs. Typically the scaling varies between 3 and 6 levels for each component of risk. Likelihood is typically measured on a scale ranging from Likely-to-Occur to Improbable or Impossible. Consequence usually ranges from Catastrophic to Negligible. No matter which scaling method is used, the highest risks are those with the highest likelihood and the worst consequences.

Risks are not limited to technical concerns but can include risks to schedule and budget as well. Regardless of the source, once a risk is identified, likelihood and consequence must be determined so the risk can be mapped on the risk matrix. This evaluation can be either quantitative or qualitative. Some of the methods used to analyze risk include: Probabilistic Risk Assessment (PRA), statistical analysis of historical data, Fault Tree Analysis, and Failure Mode and Effects Analysis. [66]

Traditionally, hardware reliability also is used to determine the likelihood of failure. However, using hardware reliability as the only means of assessing likelihood represents two erroneous assumptions. The first incorrect assumption is that hardware failures are the only causes of accidents. This is not true and in fact is often not the case in many large-scale accidents. Accidents and unintended events can result from human error, bad requirements, processes not being followed or design deficiency. Managers and decision makers must consider all of the possible causes of accidents or overall system failure when assessing the total risk for a project.

The second incorrect assumption is that reliability can be equated to successful operation. This assumption is particularly not true for software. Each component or module may operate as intended and hence be very reliable. A combination of components may, however, produce unintended behavior that results in a system failure. Unclear requirements or design flaws may also yield highly reliable components that do not meet the intent of the system's operation. Program managers must be conscious of these misconceptions when evaluating the overall program risk.

Once risks are identified and plotted on the risk matrix, there must be a way to compare and prioritize all of the risks. Again, each program and industry has chosen different methods of accomplishing this risk ranking process. NASA's Risk Management Procedures and Guidelines (NPG8000.4) suggests using a Risk Assessment Code (RAC). The RAC are numerical values assigned to each field in the risk matrix, as illustrated in Figure 8. [67] The RAC values appear to be assigned in an evenly distributed, impartial manner. Risks with a RAC of 1 are the highest and should receive top priority in terms of resources and attention.

	LIKELIHOOD ESTIMATE				
CONSEQUENCE CLASS	**Likely to Occur**	**Probably will Occur**	**May Occur**	**Unlikely to Occur**	**Improbable**
Catastrophic	**1**	**1**	**2**	**3**	**4**
Critical	**1**	**2**	**3**	**4**	**5**
Moderate	**2**	**3**	**4**	**5**	**6**
Negligible	**3**	**4**	**5**	**6**	**7**

Figure 8 – Risk Matrix Showing Risk Assessment Code (RAC) (Adapted from [68])

Individual NASA programs use the risk matrix as a management tool to manage and track risk mitigation actions. Each risk is assigned a managing organization that has the responsibility of establishing a plan to reduce the likelihood of the risk occurring or of lessening the severity of the consequence should the event actually take place. This plan is entered into a database and the action tracked until the risk is reduced to an acceptable level. [69] The program manager is typically responsible for determining the acceptable risk level. One downside of this management tool is that risks are evaluated individually. Any unexpected emergent behavior that may result from a combination of risks will not be discovered or considered through the use of the risk matrix.

The NASA Risk Management Procedures and Guidelines (NPG 8000.4) provides the basic principles of a risk management plan. Each program manager is then required to develop a detailed risk management plan specific to their individual program/project, based on the general guidelines. Each program manager is responsible for determining how their risks should be ranked and prioritized and they must also determine what level constitutes acceptable risk. While the ability to tailor the risk management process does allow managers to better meet the needs of their project, it also adds to the inconsistencies of test approaches across the agency and industry.

One example of how a large-scale NASA program has adapted the NASA risk management guidelines is depicted in Figure 9. In this application, the risk matrix is comprised of 5 levels of likelihood and 5 levels of consequence. These levels are labeled numerically 1-5 with 5 being the highest or most severe.

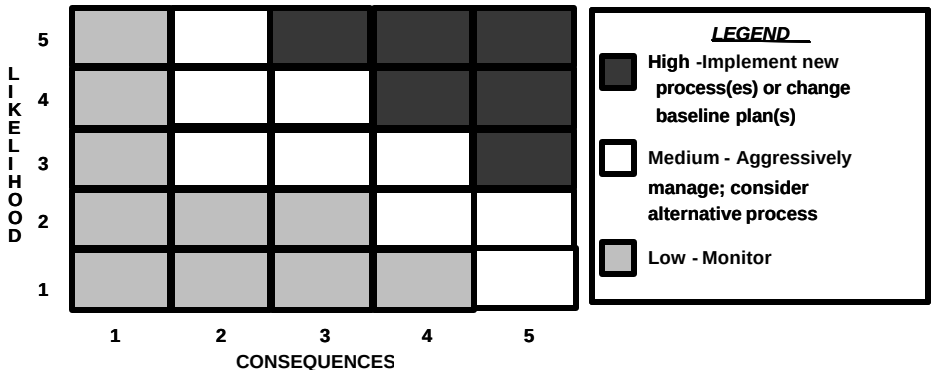

Figure 9 – NASA Program Risk Matrix [70]

The risk management plan for this program suggests that for each risk, the likelihood and consequence should be multiplied together to determine a risk score. [71] These scores are then used to rank and prioritize all of the risks. For example, a risk with a likelihood of 4 and a consequence of 5 would receive a risk score of 20. This score is not an absolute number but rather a relative ranking used to prioritize all risks. The methodology of multiplying likelihood times consequence, however, does little more than assign a value to each field in the matrix, similar to the concept of the RAC used in NASA's generic risk matrix. The assignment of risk scores to matrix fields is illustrated in Figure 10.

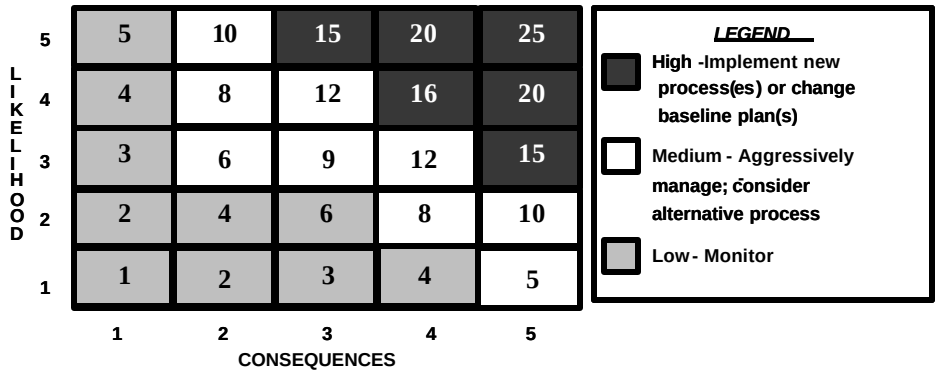

Figure 10 – Risk Matrix with Scoring [72]

The approach of multiplying likelihood and consequence can be misleading. First, likelihood and consequence are two different entities and are not presented in the same terms. Likelihood traditionally is in the form of a probability, while consequence is not –yielding a product with meaningless dimensions. Second, it may misrepresent the relative distance between two risks. For example, a risk with a consequence of 5 and a likelihood of 4 has a score of 20, while a risk with the same consequence of 5 but a

likelihood of 2 has a score of 10. It is unreasonable to assume that the risk with a score of 10 is actually twice as good as the risk with a score of 20.

The Department of Defense (DoD) uses a hazard criticality index matrix that may serve as a better approach to prioritizing risks. A simplified example of this matrix is shown in Figure 11.

	Frequent	**Probable**	**Occasional**	**Remote**	**Improbable**	**Impossible**
Catastrophic	**1**	**2**	**3**	**4**	9	12
Critical	**3**	**4**	**6**	**7**	12	12
Marginal	**5**	**6**	**8**	10	12	12
Negligible	10	11	12	12	12	12

Figure 11 – Adapted DoD Hazard Criticality Index Matrix [73]

For this application, the number in each field represents the criticality of various identified hazards as part of the DoD's safety program. The number values are not distributed evenly, showing that some thought and methodology went into assigning ratings to each field. Any hazard below the dark line is considered acceptable. The goal of the mitigation activity is to drive down the criticality to a rating of 9 or higher (below the line). [74] Reducing criticality can be accomplished through design changes, additional redundancy, safety features or operational workarounds. This same methodology can be used for risk management. In addition, if the matrix values can be assigned at the agency level, then consistency in determining acceptable risk can be achieved throughout the program. The subjectivity of assigning likelihood and consequence values would still remain, however.

The preceding discussion on risk management techniques is only a cursory review. Further analysis of the various risk management processes and the relative benefits and drawbacks is work that should be covered by a subsequent thesis. A general understanding of risk management is needed for the purposes of this thesis. The application of risk management to testing is discussed further in the following section.

6.1.2 Application to Testing

The intention of risk management activities is to identify and mitigate risks in every stage of a program. Poorly defined risks or those risks that will not be realized until far out into the future are usually not plotted on a risk matrix due to the lack of solid information on which to base likelihood and consequence magnitude decisions. In the early stages of programs, risk assessment focuses on the design activities. At this stage, test requirements are typically unclear and will not be implemented until the end of the program. As a result, risk assessments do not deal with the early test requirement

decisions. It is only when the test activities are underway that the risk matrix is typically applied to testing. The risk matrix approach can, and should, be applied to early test requirement decisions.

Because testing builds confidence in a system's operation and serves to mitigate risk, the same risk matrix principles described above apply equally to testing as to the other parts of a project, such as design and safety. There is a weakness, however, with the current implementation of the risk matrix: likelihood is no longer an easily measured quantity (such as hardware reliability) in complex, software-intensive systems. Because today's systems are becoming increasingly more complex assessing the likelihood of an event occurring is also becoming more difficult.

While consequence evaluations are still relatively straightforward, most assessments of likelihood currently leave out too many factors and only consider traditional hardware reliability data. Testing serves to build confidence in the system engineer's assessment of the likelihood that an event actually taking place. As described in Chapter 5, and shown again in Table 4, these sources of uncertainty factors also need to be a central component in any likelihood evaluation.

Sources of Uncertainty	
- New design	**- New technology**
- New interfaces	**- New application for reuse**
- Previous testing	**- Reliability of information**
- Hardware/software maturity (design and build)	**- Team experience**
- Complexity of system (including number of interfaces/interactions)	**- Required fidelity of results**
- Possible unexpected emergent behavior	

Table 4 – Sources of Uncertainty

While there does need to be a better method of estimating likelihood, it should be noted that testing alone does not change the likelihood of an event occurring. Testing does, however, increase the confidence in the system engineer's assessment of likelihood. Testing also allows for problems to be discovered on the ground so that corrective action can be taken. Testing also does not change the consequence, unless it provides the system engineers with new knowledge about the system's operation.

Another concern in the traditional use of the risk matrix is that risks are evaluated on an individual basis. By addressing each risk separately, any unexpected emergent behavior that may result from a combination of risks will not be captured by the matrix approach. Because another goal of testing is to drive out these unexpected emergent behaviors, the risk matrix approach should be expanded to evaluate a combination of risks. Combining risks, however, may be misleading because the number of components involved in each test will vary from requirement to requirement. Testing will often address multiple risks at once, making side-by-side comparisons more complicated than comparing individual risks. For example, a component test requirement typically involves only one or two components while an integrated test requirement can include many different components. The differences in testing configuration and the weighting of the factors must be assessed in conjunction with any other rating that is derived from the analysis of the combined risks.

Perhaps the greatest strength of the risk matrix, as it is applied to testing, is in the methodical thought process that the decision maker must go through in order to use the tool. The first benefit is in reviewing and evaluating all the factors that are included in likelihood and consequence, especially the confidence factors. Most system engineers and decision makers are experienced in plotting likelihood and consequence on a risk matrix. Traditionally, each of the confidence factors has not been assessed in an independent or formal manner. This is unfortunate because the main goal of testing is to drive down uncertainty and build up confidence in the system. As discussed previously, the confidence factors include many aspects beyond what is traditionally considered as a source of uncertainty. Typically a new design is identified as a source of uncertainty but too often project managers choose to reuse components to reduce uncertainty. As described earlier, reuse can actually be a large source of uncertainty because managers are too over-confident with the reuse. Thinking through the total level of uncertainty will aid decision makers in identifying weaknesses in time to test and correct before on-orbit operations. The true pay-off comes in consciously thinking through each decision factor individually and in context with the other factors.

Finally, one additional concern with the current risk management practices, as applied to testing, should be addressed. This concern deals with the competition between technical and cost risks. NASA, like other industries, typically tracks technical, cost and schedule risks. Because testing takes time and costs money, decision makers are forced to trade-off the technical risk of not performing a test with the cost and schedule risks of adding additional testing. When considering the cost risk, some programs only address the cost of abating the risk and not the cost of the risk if it would occur. [75] This is counter to the purpose of testing. As described previously in this thesis, testing is both vulnerable to cutbacks and testing costs are not tracked in a manner that allows true life-cycle comparisons to be made. These conditions only serve to give more weight to the cost and schedule risks over technical risks. Decision makers must be cognizant of these influences, especially when using the risk matrix to assess test requirements.

6.2 Additional Keys to Success

A key to any successful test program is a holistic view of both the technical and non-technical factors involved in all projects. One set of factors should not be weighted more than the other; each plays an important role in the overall process. In addition to the risk management and decision-making process factors described earlier, there are other important aspects that can increase the effectiveness of a test program and in turn increase the chances for the project's success. These recommendations for ensuring success are discussed in this section.

6.2.1 Status of Testing

As discussed in the previous chapters, test engineers, and testing in general do not receive the attention or status that they deserve. It is important for program managers to give testing as much consideration as the other aspects of a project. Testing also needs to receive attention from the very beginning of the development process. While testing may not be the top priority in the early stages, it cannot be ignored either. Including test engineering representation in the design definition phase is one way that a program manager can provide this early involvement.

Testing is a vital component in all projects and good test engineers are valuable assets not only to the project itself but also to NASA as a whole. NASA, and all contractors involved in human-rated spacecraft, should work to raise the prestige of testing within the agency. Test engineering needs to be considered a valid profession and it should have its own viable career path. A test organization should be made up of experienced system engineers, and it should not be used as a first assignment for new engineers. Because testing is more of an art than a science, management should proactively seek out those engineers that demonstrate the ability to be good artisans in the testing arena. Testing skills should be viewed as desirable and rewarded as such. When a project is successful, it is typically the design engineers who receive all of the recognition and rewards. Too often, test engineers do not receive the credit they deserve for their contribution to the success of a project. A good balance between all disciplines is needed to ensure the proper balance of the project itself.

6.2.2 Organizational Considerations

The organizational structure that a project manager selects has a significant impact on the overall success of the project. Consideration must be given at the earliest stages to the appropriate structure that best serves every phase of the project. It is easy for managers to focus solely on the tasks at hand when first organizing the project, believing that the organization can change along with phase of the project. This belief is valid but only to a limited extent. The organizational structure can be adjusted slightly as time progresses, but it is important that the organization allows for the proper flow of communication and facilitates the involvement of all the essential players. Both system engineers and test engineers should be included as essential participants from the earliest stages.

There are a number of advantages in establishing a separate test organization, as was discussed in previous chapters. The primary benefits are in the ability to maintain a focus on the testing activities, to ensure a consistent test philosophy, and to develop test engineering skills. While having a separate organization is beneficial, it also requires increased coordination. Strong system engineering is essential, especially if the team is widely dispersed. The coordination mechanisms must be addressed at the time the project is structured and put into place from the beginning.

Another important organizational consideration is in the division of a system into sub-systems. Project managers have become adept at partitioning systems in such a way to limit the number of interfaces between sub-systems. [76] Project managers must also consider what each sub-system should include, such as software. Software should not be treated as a standalone sub-system. To ensure that software is integrated into each sub-system, the organizational structure should also include software representation in each sub-system group. *Accountability, along with responsibility, for software must be given to the system engineers*. Software is an integral part of the system and should be treated equally with the other system components.

6.2.3 Knowledge Transfer

As discussed previously, test engineering skills are not formally taught and must be learned on the job. To ensure successful test programs, both project managers and functional supervisors should establish mentoring programs for test engineers. Rather than the typical ad hoc nature of testing training that is evident today, managers must be proactive in developing the testing talent that will be required in the future. Capturing and retaining testing skills is especially important after the completion of large projects. A great deal of knowledge is gained through the course of completing large-scale or long-duration projects. This knowledge and expertise is often lost after the team is disbanded, and the next project must start from scratch in developing the needed testing skills.

A more strategic approach should be taken in capturing testing knowledge and in maintaining skills. One solution would be to create a small testing corps that remains intact after one project completes and then moves on to another project. While the systems would vary from one project to another, the same testing skills would apply. In addition, this testing corps would also be mentoring other engineers on the project, proliferating the needed expertise throughout the agency.

Additionally, NASA should also consider the need for agency-wide knowledge-sharing on testing. Currently the program and project guidelines (NPG 7120.5) do not address actual test implementation. An agency-wide team should be formed to seek out best practices in the testing arena and update NPG 7120.5 to include more guidelines. Formal training programs in test engineering could also be investigated and established. This training would not replace the need for mentoring.

6.2.4 Alignment of Goals

For any project to be successful, all participants must be aligned with the overall goals of the project. The primarily tool that program managers use to determine the organization's ability to meet its goals is performance metrics. Each organization is typically only measured against the group's individual output. For example, the design organization is evaluated on their ability to deliver a good design while the testing organization is measured against the ability to meet testing milestones.

Because testing occurs at the end of the project, many of the initial groups involved with the design and manufacturing are disbanded. Problems found during testing that need to be corrected often result in a delay to the schedule. Because the test organization is evaluated on the ability to meet milestones, these delays can reflect negatively on their performance. If the problem is a result of a design deficiency, it should be attributed to the design organization. Too often the test organization is often left to assume the full responsibility of the system's operation. The design organization should share the responsibility for the systems final performance and should be rated on the testing activities as well. In fact, the test organization should be rewarded, rather than sanctioned, for finding the flaw prior to on-orbit operations, rather than sanctioned for the delay in schedule.

A better approach to performance measurement would be for each organization to be measured against the overall success of the project. Project managers need to establish a method of delaying the organization's final rating until the completion of the project, so that all groups can be held accountable for the systems ability to meet the project's ultimate goals.

6.3 Testing Decision-Making Considerations

As discussed in the previous chapter, decision makers are faced with both technical and non-technical influences when making decisions. This section will describe the differing effects these influences can have on individuals with various decision-making preferences and styles.

Every person has their own decision-making style based upon personal preferences, experiences and expertise toward viewing decision alternatives. In order to ensure that testing decisions are made in a consistent manner, these decision-making influences must be understood by the decision maker. Three broad perspectives have been identified that affect individual decision makers: risk-tolerance, long-term view and systems approach. By understanding the influences on the decision-making process, managers will be better equipped to face the challenges of a development program.

6.3.1 Risk-Tolerance

In an ideal world, managers compare decision alternatives from a neutral perspective and make choices based on the expected value of the outcomes under consideration.

When two or more alternatives have the same expected value, a manager will consistently make the same choice based on established selection criteria. However, in the real world, decisions are not always framed in terms of expected value and the selection process is not always consistent between individuals. One way to describe differences between decision makers is by the amount of risk they are willing to accept, referred to as their risk tolerance level. A manager with a high risk-tolerance level will be willing to accept more risk than a manager with a low risk-tolerance level (risk averse).

It is important for managers to understand their personal risk-tolerance levels when faced with difficult decisions. In the testing environment, a technical manager who is risk-averse will want to perform more tests in order to decrease the chance of a system failure. By making decisions that reduce risk, the test manager will contribute to improved safety of the system. However, if a risk-averse manager is considering a test decision from a cost perspective only, a decision to delete a test may be chosen in order to reduce the risk of a cost overrun. In the case of the cost-related decision, the manager is reducing business risk at the expense of technical risk. Therefore, to avoid the paradox described above, the technical and non-technical decision factors must be considered as a whole.

One way to benefit from the differences of individual decision-making preferences is to match decision-making styles to specific types of management positions. Managers responsible for safety-related issues should have a risk-averse personality, while a manager who has a high tolerance for risk may be more qualified making resource and political decisions. The best approach is to create a management team the blends risk averse personalities with those that can accept higher levels of risk. A balanced team will provide a collaborative environment that will place proper emphasis on all of the decision factors being considered.

6.3.2 Long-Term View

The timeframe for realizing the effects of a decision is another way that managers differ in their decision-making preferences. In general, managers will give the immediate impacts of a decision more weight than the long-term effects. One reason that short-term impacts have more influence is that there is less time to mitigate short-term effects and therefore they take on a sense of urgency. Another reason is the false perception that there may by more options available to address long-term concerns, allowing test managers time to avoid them in the future. By downplaying the long-term effects of decisions, managers are vulnerable to a "wishful thinking" mentality that may create more, and bigger, problems in the future.

Testing decisions are also viewed differently depending on the phase of the program in which the decision is made. Early in a development program, decisions can be made in an aggregate fashion in which trade-offs can be properly analyzed. As time progresses, however, decisions must be evaluated on an individual basis with fewer alternatives to

consider. The result is that decisions to add testing may be harder to implement in the later phases of a development program.

6.3.3 Systems Perspective

Decision makers at all levels must consider their decision alternatives from a systems perspective that considers the higher-level objectives of the organization, rather than those of their immediate domain. Often, managers from different parts of an organization are faced with situations in which they must compete for available resources. Without a systems approach, one manager's gain will become another's loss. Viewing decision alternatives in a holistic manner will often provide a synergistic alternative in which all factors can be successfully addressed. However, to be successful, managers must be willing to accept a less than optimum alternative for their particular part of the overall program. In an era of decreasing resources, this approach is vital to ensuring successful spacecraft test programs.

In addition to considering decisions alternatives in terms of higher organization objectives, the scope of the decision must also be viewed from a systems perspective. Individual decisions are often perceived to have a larger impact to the decision maker than those that are considered in the context of an overall program change. For instance, if a spacecraft test program had an estimated cost of $30M. Considered as an individual event, there would be concern that the cost was too high. However, if the cost of the testing were compared to the total cost of the entire program, say $12 billion, it becomes clear that the testing cost is more acceptable. In this case, the cost of the test program accounts for only small percentage of the total program cost and can provide benefits that far exceed the $30M cost of testing. These benefits can come in the form of increased confidence in the performance of the spacecraft.

Finally, taking a systems approach to decision making allows managers make the necessary trade-offs between the technical and non-technical factors. Risk mitigation, cost and schedule must be considered when making testing decisions, but the technical objectives of the program must be maintained to ensure overall success.

In summary, the three decision-making considerations discussed above are interrelated and can be summarized as follows. The systems approach helps decision-makers understand the importance of the long-term view of decisions. Systems' thinking also allows decision-makers to consider technical and non-technical factors relating to testing. The timing of decisions is also important in maintaining a long- term view. Decisions made early in a test program allow the decision maker to take a long-term view, but as time progresses the number of available alternatives may diminish. Finally, decision-makers must be aware of their individual decision-making preferences regarding risk tolerance. A balanced team that consists of risk averse and risk tolerant managers will provide the best combination for consistent decision-making.

Chapter 7: Summary

The overall objectives of this thesis were to document the test approaches and philosophies used within NASA and other organizations, to determine if the critical decision factors can be applied to all human-rated spacecraft programs, to provide lessons learned for future projects and to understand how various programs address unexpected emergent behavior. This final chapter will review the conclusions and recommendations that resulted from the research conducted towards meeting these objectives. Finally, suggestions for future research in this area are outlined.

7.1 Conclusions

The primary research for this thesis included interviews with experts in the system and test engineering fields. From these expert interviews, major themes were observed and findings established. This section will briefly summarize the major themes and findings.

7.1.1 Major Themes from Expert Interviews

Subjectivity of test requirements development: One purpose of testing is to reduce the risk that a system will fail to meet its intended performance goals. Because determining risk levels is highly subjective, especially for new designs and architectures, testing in turn becomes highly subjective. Typically, the actual test requirement decision process is not documented as a formal process but relies heavily on the judgment of the decision makers, adding to the overall subjectivity. Due to this subjectivity, testing practices are not consistent across the aerospace industry or in comparison with the commercial aircraft industry.

Paradoxical nature of testing: Intuitively, people know that more testing is better than less testing but all projects are under tight cost and schedule constraints. Managers must constantly balance the trade-off between too much and too little testing. Testing alone does not make a project successful, but it does raise confidence in the likelihood of success. Testing is often regarded as a drain on project resources, rather than a valuable undertaking.

Vulnerability to changes and cutbacks: Most testing occurs late in the development phase. Since budget and schedule pressures are most keenly felt at the end of a development program, testing is extremely vulnerable because it is one of the only remaining opportunities left to make up schedule or cut costs.

Inadequate attention to testing: Testing in general does not receive as much attention as it should in order to ensure a project's success. First, testing typically does not receive enough emphasis during the initial planning phases. Second, testing is not given enough attention in literature, training programs or academia. Third, test organizations do not hold the same status or standing as do other groups within a program due partly to the focus on design activities.

Organizational influence on the testing process: The structure of an organization does play a role in the overall success a program. Having a separate test organization was recommended as one method of developing skills, facilitating knowledge transfer, improving communication and establishing a consistent test philosophy. It ensures that attention is focused on testing activities even while the program management may be concentrating on other issues. In addition, having previous experience with hardware and software can be very important in recognizing system trends and unexpected emergent behavior. However, an independent testing organization does have limitations and requires increased integration and communication.

7.1.2 Major Findings of Thesis

For some projects, software is considered a stand-alone system: Today's systems are becoming increasingly more reliant on software to operate successfully. Yet for many projects software is not being treated as an integral part of the overall system. Instead, software is viewed as a separate stand-alone system, as evidenced by the project's organizational structure, the process by which the software is tested and delivered, and the limited insight that some sub-system engineers have into the software development process. In addition, many of today's managers have a background in hardware development and try to apply the same principles to software. Software, however, has many fundamental differences in the way it is developed and tested.

Optimism varies with organizational position: The interview process revealed differences in attitude toward the effectiveness of testing among those at various levels of the organization. The optimism level in the adequacy of the test programs increased as one went up the chain-of-command toward the managers. One possible explanation for this trend includes the senior managers' ability to take a larger view of the overall project. These managers are also required to balance limited resources, political factors and a larger constituency than many engineers at the lower levels. In addition, the information that managers receive may be less-than-perfect and they may be accepting more risk than they realize.

While testing practices vary, decision factors do not: Contrary to common belief, there is consistency in the decision factors used by the various decision makers. These common decision factors can be divided into technical (safety, risk and confidence building) and non-technical (resource availability, process, individual decision-making behavior and political/cultural influences). Due to their salient nature, non-technical factors often carry as much, or more, weight to the decision maker as do technical factors. Perhaps the most under-appreciated factors are those that build confidence by addressing the various sources of uncertainty. Decision makers should thoroughly consider each of the sources of uncertainty, especially the effects of reuse, an inexperienced team and potential unexpected emergent behavior.

Upfront planning is a key to success, but be prepared for change: Early planning in both systems engineering and testing is necessary to manage the complexity of

human-rated spacecraft programs. A key to the success of a testing program is in developing high-quality test requirements. The early involvement of the test organization is also essential in establishing a set of firm test requirements that will be less vulnerable to changes later in the program. In addition, upfront planning also allows for the opportunity to begin testing early and allows program managers to develop enough flexibility in the test program to make adjustments later in the project.

Current methods of tracking testing costs are not sufficient: Actual cost figures for spacecraft testing programs are very difficult to determine because they are not typically tracked as a discrete line item in spacecraft programs. Other engineering and operational funding often obscures pure test costs, which is unfortunate because accurate test cost data could be used to establish the savings that testing brings to a program in the form of cost avoidance. Life-cycle cost estimating is another area that should be enhanced and used in making test decisions. When the cost of on-orbit repairs are considered in the decision, it often becomes apparent that performing a test is money well spent in the long run.

Testing is more of an art than a science: Experience and mentoring are more important than formal training in developing test engineering expertise. Because testing practices are not formally documented, the most valuable knowledge is retained within the individual experts as tacit knowledge. In order to maintain the knowledge base and core capabilities, test engineering should be treated as a valid profession with the same prestige level as other engineering disciplines.

Good sub-system engineering is not a substitute for proper system engineering: While designing and testing at the sub-system level is necessary to reduce complexity, early attention should be applied to the overall system level performance and the possible interactions between the sub-systems. Because of the increased complexity of today's systems and distributed nature of projects, strong system engineering perspective is becoming increasingly more important. In addition, integration testing, by its very nature, crosses organizational, contractual and geographic boundaries—making proper system engineering essential to the success of a test program.

Program/Agency culture strongly affects testing: A traditional engineering culture places emphasis on design because the design must come before testing. In addition, design organizations tend to have more status within an organization than the testing team. This inequality can lead to an adversarial relationship and can hamper communications between the design and test groups, resulting in severe negative effects on program success. Another cultural affect observed within testing programs is the mindset that each program stands on its own. Few resources are spent on trying to improve testing practices for the next program by understanding the current failures and sharing lessons learned.

7.1.3 Review of Original Hypotheses

Through the interview process with experts in the testing field and subsequent analysis, the hypotheses stated at the beginning of the thesis were confirmed. The hypotheses were:

Hypothesis 1: Many decisions makers do not utilize a holistic approach in addressing testing requirements.

Most decision makers have good intentions and are making the best possible decisions based of the information they have available to them. In order to improve their decisions, all decision factors need to be considered, especially confidence factors and decision-making biases.

Hypothesis 2: Life-cycle costs are not adequately addressed in the decision-making process.

Again, decision makers must use the information that is available. Unfortunately, testing and on-orbit repair cost are not divided and tracked in a manner to yield high-fidelity life-cycle cost estimates.

Hypothesis 3: Decision makers are not fully aware of the influences inherent in the decision-making process.

There are numerous decision-making factors and influences that decision makers must evaluate and balance. Many decision makers may not fully appreciate the significance of each decision factor or the long-term effects of their decisions. In addition, proactive efforts are typically not taken to minimize unintended decision-making behaviors or to capitalize on the desired biases.

Hypothesis 4: The influence of organizational factors is not fully appreciated.

Most managers do understand that the organizational structure does influence the overall efficiency of the project. What is not fully appreciated is the significant impact that organizational considerations have on testing and integration efforts later in the program. Most important, the ability to integrate software into each sub-system is strongly affected by organizational considerations.

7.2 Recommendations

Numerous recommendations for improving future NASA human-rated spacecraft test programs have been discussed throughout this thesis. The most significant recommendations are summarized in this section. Even though these proposals specifically

address NASA programs, each can be applied, in principle, to other programs and industries.

- Form an agency-wide team to seek out best practices in the test engineering field. NPG 7120.5 should then be updated to give managers detailed guidelines for test planning and implementation.

- Enhance the current risk management practices to include an assessment of all decisions making factors that contribute to the overall level of likelihood. In addition, the current rationale used to rank competing risks should re-evaluated.

- Establish improved training and mentoring programs for test engineers. The agency should also evaluate the feasibility of establishing a small testing corps that would remain intact and move from one project to another, augmenting the resident testing teams of each project. This testing corps would mentor the other engineers on the project, while learning new skills and best practices as well. This approach would proliferate the needed test expertise throughout the agency, which is important to maintain because contractor organizations are not consistent between programs and projects.

- Assign both responsibility and accountability for software to system engineering. Software is an integral part of the system and should be treated equally with the other system components.

- Consider test engineering a valid profession with a viable career path for test engineers. This would help to improve the current level of prestige that testing receives within the agency and also help to retain the critical test engineering skills.

- Track actual testing costs in a manner that will allow for better estimates of future test requirement costs. In addition, a better method for tracking and estimating life-cycle costs should be established.

- Include testing in the earliest stages of the spacecraft's development process. Proper system and test engineering representation in the design definition phase will increase the fidelity of the test requirements and effectiveness of test programs. In addition, having previous experience with the hardware and software can be very important in recognizing system trends and unexpected emergent behavior in the later phases of the program.

- Recognize and understand personal risk-tolerance levels and how individual decision-making styles affect decisions. Furthermore, decision-making styles could be matched to specific types of management positions. A balanced team of risk-adverse and risk-tolerant personalities will provide a collaborative environment that will place proper emphasis on all of the decision factors being considered.

- Establish separate test organizations within the projects in order to develop skills, facilitate knowledge transfer, improve communication, and establish consistent test philosophies. If a separate test organization is established, however, increased integration and communication will be required.

- Align all project participants with the overall goals of the project and hold them accountable for the systems ultimate success or failure.

7.3 Future Work

Based on the findings and recommendations presented in this thesis, three suggestions for future research are provided. These suggestions are offered as next steps for additional understanding of the nature of testing within spacecraft development programs.

- Interviews from outside the human-rated spacecraft arena should be conducted to capture the expert testing knowledge that exists across NASA and outside of the agency. A comparison of the common themes and findings could be made to those identified in this thesis in order to build on the existing foundation of knowledge.

- Investigation into the current methods for estimating and tracking spacecraft testing life-cycle costs should be conducted and a reliable cost estimating tool developed. Such a tool would be useful for future spacecraft development program planning and budgeting.

- Further research into the individual decision-making process, as it applies to testing, is also suggested. A deeper understanding of the testing decision-making process will serve as a starting point for the development of a rigid decision making framework.

Notes

[1] Young, T. "**Mars Program Independent Assessment Team Report**". NASA, March 14, 2000.

[2] Link, D.C.R (Chairman). "**Report of the Huygens Communications System Inquiry Board**". NASA, December 2000.

[3] Zarakhovich, Y. "**Why No One Escaped From The Kursk**". Time Magazine. June 3, 2002.

[4] Zarakhovich, p. 1.

[5] NASA. **NASA Systems Engineering Handbook, SP-6105**. NASA, June, 1995.

[6] NASA. **Program and Project Management Policy Guide, NPG 7120.5B**. NASA. 2000.

[7] Bonabeau, Eric. "**Predicting The Unpredictable**". Harvard Business Review. March, 2002.

[8] Blanchard, B.S. and Fabrycky, W.J. **Systems Engineering and Analysis**. Prentice Hall. January, 1998, p.121.

[9] NASA (SP-6105), p. 3.

[10] NASA (SP-6105), p. 3.

[11] NASA (SP-6105), p. 21.

[12] NASA (SP-6105), p. 21.

[13] NASA (SP-6105), p. 106.

[14] Wilkins, D. J. **"The Bathtub Curve, Infant Mortality and Burn-in"**. Reliability Hot Wire, http://weibull.com/hotwire/issues21/hottopics21.htm.

[15] NASA (SP-6105), p. 106.

[16] NASA (SP-6105), p. 106.

[17] Blanchard, p. 126.

[18] NASA (SP-6105), p. 106.

[19] Blanchard, p. 121.

[20] Herbert, F. http://www.quotationspage.com.

[21] Expert Interview #8, October 1, 2002.

[22] NASA (SP-6105), p. 38.

[23] Expert Interview #9, October 2, 2002.

[24] Expert Interview #11, October 8, 2002.

[25] Expert Interview #1, August 13, 2002.

[26] Expert Interview #6, August 29, 2002.

[27] Expert Interview #6, August 29, 2002.

[28] Expert Interview #7, September 30, 2002.

[29] Expert Interview #13, October 29, 2002.

[30] Expert Interview #5, August 20, 2002.

[31] Expert Interview #7, September 30, 2002.

[32] Expert Interview #11, October 8, 2002.

[33] Expert Interview #6, August 29, 2002.

[34] Expert Interview #10, October 4, 2002.

[35] Expert Interview #13, October 29, 2002.

[36] Expert Interview #8, October 1, 2002.

[37] NASA (NPG 7120.5B), p. 64.

[38] Expert Interview 12, October 28, 2002.

[39] Expert Interview #13, October 29, 2002.

[40] Robbins, S.P. **Organizational Behavior, 9th Edition**. Prentice Hall. 2001, p. 424.

[41] Expert Interview #5, August 20, 2002.

[42] Forsberg, K and Mooz, H. **Visualizing Project Management**. John Wiley and Sons. May, 2000.

[43] Forsberg, p. 21.

[44] Expert Interview #6, August 29, 2002.

[45] Leveson, N.G. **Safeware: System Safety and Computers**. Addison Wesley, 1995.

[46] Margolies, D. "**Test What You Fly?**". NASA, ASK Magazine, Volume 9, p. 7 October 2002.

[47] NASA (NPG 7120.5B), p. 91.

[48] NASA (NPG 7120.5B), p. 91.

[49] Expert Interview #9, October 2, 2002.

[50] Lions, J.L. (Chairman). "**Ariane 501 Failure: Report by the Inquiry Board**". European Space Agency, 19 July, 1996.

[51] Lions, p. 16.

[52] Leveson, N.G. "**The Role of Software in Spacecraft Accidents**". Massachusetts Institute of Technology. 2002.

[53] Expert Interview #4, August 20, 2002.

[54] Expert Interview #7, September 30, 2002.

[55] Expert Interview #6, August 29, 2002.

[56] Link, p. 7.

[57] Branscome, D.R. (Chairman). "**WIRE Mishap Investigation Board Report**". NASA, June 8, 1999.

[58] Arceneaux, W.H. "**International Space Station (ISS) Integration and Verification Methods, Current Status, and Future Plans**". AIAA. October, 2002.

[59] Expert Interview #11, October 8, 2002.

[60] Expert Interview #9, October 2, 2002.

[61] Expert Interview #9, October 2, 2002.

[62] Leveson (Safeware), p. 75.

[63] NASA. "**Risk Management Procedures and Guidelines", NPG8000.4**. NASA. April, 2002.

[64] NASA (NPG 8000.4), p. 11.

[65] NASA (NPG 8000.4), p. 12.

[66] NASA (NPG 8000.4), p. 10.

[67] NASA (NPG 8000.4), p. 11.

[68] NASA (NPG 8000.4), p. 12.

[69] NASA. **International Space Station Program, Risk Management Plan, SSP 50175A**. NASA. April, 2002.

[70] NASA (SSP-50175A), p. 18.

[71] NASA (SSP-50175A), p. 14.

[72] NASA (SSP-50175A), p. 18.

[73] Leveson (Safeware), p. 273.

[74] Leveson (Safeware), pp. 272-273.

[75] NASA (SSP-50175A), p. 18.

[76] Allen, T.J. "**Organizing for More Effective Product Development** (Working Paper). January, 2002.

References

Allen, T.J. "**Organizing for More Effective Product Development** (Working Paper). January, 2002.

Arceneaux, W.H. "**International Space Station (ISS) Integration and Verification Methods, Current Status, and Future Plans**". AIAA. October, 2002.

Blanchard, B.S. and Fabrycky, W.J. **Systems Engineering and Analysis**. Prentice Hall. January, 1998.

Bonabeau, Eric. "**Predicting The Unpredictable**". Harvard Business Review. March, 2002.

Branscome, D.R. (Chairman). "**WIRE Mishap Investigation Board Report**". NASA, June 8, 1999.

Chandler, S. **MEIT Overview**. NASA, November, 2001.

Chiles, J. **Inviting Disaster**. Harper Collins. 2001.

Forsberg, K and Mooz, H. **Visualizing Project Management**. John Wiley and Sons. May, 2000.

Gallina, T. **6A MEIT I**. NASA. August, 2002.

Hasting, D. **Space Systems Architecture and Design**. Space Systems, Policy Architecture Research Consortium. March, 2002.

Jordan, J., Michel, F. **The LEAN Company, Making The Right Choices**. Society of Manufacturing Engineers. 2001.

Kahneman, D. **Choices, Values and Frames**. Cambridge University Press. 2002.

Kohler, H. **Statistics For Business and Economics, 2nd Edition**. Scott, Forsman and Company. 1988.

Krieck, J. "**Lessons on Program Management, How To Accumulate Scar Tissue**". MIT, Systems Design and Management. July, 2002.

Leveson, N.G. "**The Role of Software in Spacecraft Accidents**". Massachusetts Institute of Technology. 2002.

Leveson, N.G. **Safeware: System Safety and Computers**. Addison Wesley, 1995.

Link, D.C.R (Chairman). "**Report of the Huygens Communications System Inquiry Board**". NASA, December 2000.

Lions, J.L. (Chairman) "**Ariane 501 Failure: Report by the Inquiry Board**". European Space Agency, 19 July, 1996.

Margolies, D. "**Test What You Fly?**". NASA, ASK Magazine, Volume 9. October 2002.

McManus, H, Warmkessel, J. **New Methods for Architecture Selection and Conceptual Design, SPARC Program Overview**. Space Systems, Policy Architecture Research Consortium. March, 2002.

NASA. **Independent Verification and Validation (IV&V) Criteria, NPG 2820**. NASA. Proposed Appendix.

NASA. **International Space Station Program, Risk Management Plan, SSP 50175A**. NASA. April, 2002.

NASA. **NASA Systems Engineering Handbook, SP-6105**. NASA, June, 1995.

NASA. **Program and Project Management Policy Guide, NPG 7120.5A**. NASA. 2000.

NASA. **Risk Management Procedures and Guidelines, NPG 8000.4**. NASA. April, 2002.

NASA. **Software Independent Verification and Validation (IV&V) Policy, NPD 8730.4**. NASA, August 2001.

Robbins, S.P. **Organizational Behavior, 9th Edition**. Prentice Hall. 2001.

Schultz, D., Bachman, J. **A Matrix Approach to Software Process Definition**. November, 2000.

Sterman, J. **Business Dynamics, Systems Thinking and Modeling For A Complex World**. McGraw Hill. 2000.

Urlich, K, Eppinger, S. **Product Design and Development**. McGraw Hill Company. 2000.

Young, T. "**Mars Program Independent Assessment Team Report**". NASA, March 14, 2000.

Zarakhovich, Y. "**Why No One Escaped From The Kursk**". Time Magazine. June 3, 2002.

Appendix A – Expert Interview Questionnaire

Section I: Basic Information

1. Full Name (optional): _____ 2. Date: ___/___/___

3. Organization: _____ 3A. Location (City/State): _____

4. Job Title: _____

5. Number of years working at your job: ___ 6. Area of Expertise: _____

7. Which of these best describes your Job Title:
(a) Systems Engineer (b) Project Manager (c) Other (please specify): _____

8. Email Address: _____

9. Telephone Number: _____

Section II: The Role of Testing and Requirement Definition:

10. What type of testing is performed by your organization? (qualification, acceptance, integration, etc.).

11. What purpose does testing serve in your program, how important is it? (In other words, is it a double-check or is it the only way items are verified)?

12. Would you consider one type of testing more important than the others?

13. What would you consider to be the optimal phase in a new program to address test requirements for spacecraft programs and why?
    a. In your experience, when does it actually take place?
    b. What are the advantages and disadvantages?
    c. What are the barriers to the optimal timing?

14. How is criticality determined and functionality prioritized? What systematic way do you use to establish this priority and when in the process is it accomplished?

15. What criteria are used to establish the robustness and accuracy of emulators and simulators. How does this decision affect other testing decisions?

16. What process do you use to define/evaluate/approve test requirements initially?

17. What is the process for determining the level of testing to be performed and the time/phase when it is performed?

18. How and when are decisions made regarding the use of prototyping, proto-flights, or structural test articles?

19. Are test requirement reviews held in parallel with other system level reviews? Why or why not?

20. How are cost factors used in deciding when, and to what extent, testing is performed?

Section III: Decision Making Factors

21. What factors do you consider when deciding what tests to perform?
    a. Are the factors always the same or do they change based on the situation?
    b. Where does the data come from that decisions are based on?
    c. What do you use (or have you seen used) as the basis for decisions (for example: past experience, organization/program policy, defined framework, gut feeling, political influence?

22. What other external factors influence the decisions?

23. What cost factors influence test requirement development?

24. Relatively, how are these factors weighted and does this weighting change depending on the situation?

25. Under what circumstances does planned testing change?

26. Sometimes during testing the system will act in an unexpected manner, allowing you to learn more about the design and/or unintended behavior. Do you anticipate that this will happen and if so, how do you handle this in developing requirements? How do you handle the risk of unknown system behavior?

27. Is hardware and software system testing treated differently? In what way?

28. What is the best time to merge hardware and software testing? Who makes that decision?
    a. What are the trade-offs between the different options?
    b. How are these trade-offs handled…what determines the priority?

29. What are the similarities between the hardware and software testing environments and what are the differences?

30. For software testing, how do you determine what tests are repeated (after a new s/w release or change) vs. those that can be simulated?

Section IV: Testing Budgets and Cost Considerations

31. Is there a separately identifiable budget line item for testing in your program? If, so what amount is it and what percentage of the total program cost does it represent?

32. How does this compare to other programs?

33. How much of the program budget should be spent on testing?

34. Are life cycle costs considered in testing decisions and if so how?

Section V: Organizational Factors and Knowledge Transfer

35. What would you consider to be the optimal way to organize to support testing (should there be a separate testing group)?
    a. In your experience, how are the programs organized?
    b. What are the advantages and disadvantages?
    c. What are the barriers to the optimal organization?

36. Who "owns" (or should own) the requirements?

37. Who makes the recommendations and who is the final decision maker when it comes to testing?

38. How does System Engineering affect the different stages of testing? How does reality compare to the ideal situation?

39. What best practices can be obtained from earlier programs and how can they be incorporated into current testing programs? How well are these lessons transferred to other programs?

40. What would you change about your testing program if you could?

Appendix B - Glossary

Analysis:	The use of specific techniques, such as statistics, modeling, or qualitative methods to verify compliance to specifications/requirements.
Consequence:	An assessment of the worst credible potential result(s) of a risk.
Inspection:	Techniques used to physically verify design features.
Likelihood:	The probability that an identified risk event will occur.
Risk:	The combination of (1) the probability (qualitative or quantitative) that a program or project will experience an undesired event such as cost overrun, schedule slippage, safety mishap, compromise of security, or failure to achieve a needed technological breakthrough; and (2) the consequences, impact, or severity of the undesired event were it to occur.
Safety:	Freedom from those conditions that can cause death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment.
Similarity:	A verification technique that uses comparison of similar hardware components.
Simulation:	A technique for verification that uses hardware or software other than flight items, but built to behave in an identical manner as flight systems.
Test Organization:	Those engineers accountable for implementing test requirements and the program management functions for these activities.
Testing:	A method of verification in which the actual equipment is allowed to operate under conditions that demonstrate its ability to perform as specified by the design requirements.
Uncertainty:	The lack of knowledge about an outcome or event that hampers a manager's ability to make decisions and draw conclusions.
Validation:	Determining that a system performs as intended by the customer.
Verification:	Determining that a system performs according to the design requirements.