The Thesis committee for Michael Dax Garner
Certifies that this is the approved version of the following thesis:

Systems Engineering Processes for a Student-based Design Laboratory

APPROVED BY

SUPERVISING COMMITTEE:

Robert H. Bishop, Supervisor

Lisa Guerra

Systems Engineering Processes for a Student-based Design Laboratory

by

Michael Dax Garner, B.S.

THESIS

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2009

To my wife, Ashlea Dawn, for all her love and support. Without her motivation none of this would have come to fruition. Also, to my son, Azden Zayde, whose birth on October 24, 2009 made writing a thesis and graduation all the more eventful. I could not have done it without either of them.

Acknowledgments

I owe a great deal of gratitude to the following people that have supported and inspired me throughout my higher education culminating into this thesis.

Henri C. Kjellberg[⋆], colleague, best man, best friend. I am forever indebted to Henri for long hours of discussions producing innovative ideas in more areas than just aerospace engineering. His endless loyalty and support helped make this thesis possible and my college career truly enjoyable.

Dr. E. Glenn Lightsey, for encouraging an active student satellite program. This application-oriented program complements the course work, provides students unparalleled hands-on experience, fosters a strong sense of teamwork, and has been the cornerstone of my education. Thank you Dr. Lightsey for simply providing one of the greatest opportunities in my life.

Lisa A. Guerra for opening my eyes to the big picture. As my mentor in systems engineering, Lisa provided encouragement, sound advice, great teaching, good company and lots of wonderful ideas. I would have been lost without her.

Finally, I would like to thank Dr. Robert H. Bishop for the opportunity to freely pursue my research interests. As well as, fellow graduate students: John Christian, Serena Zhang, Cinnamon Wright and Sebastian Munoz; and my student colleagues who provided dedication, camaraderie and entertain-

ment in the UT Satellite Design Lab: Bryan Anderson, Chris Young, Karl McDonald, Lindsey Roberts, Carlos Rocha, Jahshan Bhatti, Travis Imken and Ron Maglothin. It is a pleasure to thank those who made this thesis possible.

Systems Engineering Processes for a Student-based Design Laboratory

Michael Dax Garner, M.S.E.
The University of Texas at Austin, 2009

Supervisor: Robert H. Bishop

A student-based university environment for engineering design and development is much different from a product development environment within the aerospace industry. Therefore, a different approach to systems engineering should be considered. By its very nature, a university product development laboratory thrives on creativity and rejects bureaucracy. Experience shows that continuity and discipline within a project is crucial for success. The practice of systems engineering enables technical project discipline. Systems engineering is the art and science of developing an operable system that meets requirements within imposed constraints. The purpose of this thesis is to describe the systems engineering processes and techniques necessary for a student-based project, and explicitly show how to implement these processes. Although attempts have been made to utilize a few systems engineering techniques in past projects, many students did not properly and consistently apply those techniques to the technical design work. The goal of the thesis is to tailor the NASA systems engineering processes to a student-based design laboratory

environment and to apply the methodologies to the mission design of Paradox. The Picosatellite for Autonomous Rendezvous and Docking on-Orbit eXperiment, or Paradox, is the second of four missions to demonstrate autonomous rendezvous and docking with a picosatellite-class satellite.

A strong technical contribution highlighted within the thesis involves developing an open architecture rendezvous targeting algorithm for the Paradox mission in the face of large mission architecture uncertainties. The robust targeting algorithm builds from previous work utilizing an optimizer based on the Clohessey-Wiltshire equations and an iterative Lambert targeter. The contribution extends the rendezvous transfer times by including a multi-revolution Lambert targeter. The rendezvous algorithm will perform successfully given any launch vehicle and target spacecraft vehicle supporting the notion of an open architecture to satisfy the mission. The development of the algorithm is embedded within the context of the systems engineering processes to clearly showcase the intimate connection between systems engineering processes and the technical engineering design of a mission.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The goal of the thesis is to present systems engineering processes for a student-based design laboratory and to apply these necessary processes to an ongoing spacecraft mission design. However, there are multiple styles of systems engineering across multiple disciplines including ones from the US Department of Defense and the International Council on Systems Engineering (INCOSE). The NASA standard was selected as the baseline and was tailored to the needs of a student-based design laboratory.

As background, an undergraduate course entitled Space Systems Engineering is offered in Aerospace Engineering at the University of Texas at Austin (UT Austin). The pilot was sponsored by NASA's Exploration Systems Mission Directorate in the spring semester of 2008, and now the course is required at UT Austin and offered at other universities. The systems engineering practices taught in the class encompass the full spectrum of the NASA systems engineering procedures supported with examples from NASA missions. The goal of the undergraduate systems engineering class is not to transform the aerospace engineers into systems engineers, but to provide an awareness and appreciation of systems thinking, tools, and processes encountered as professionals. The challenge in gaining a deep understanding of systems engineering is that actual project experience is a key component and not generally available

in a standard lecture course.[15]

Fortunately, students at UT Austin are gaining project experience in student-based design laboratories. Specifically, projects in the UT Satellite Design Lab (SDL) introduce students to many of the facets of systems engineering.[15] Since a majority of students involved with the SDL have taken, or will eventually take, the Space Systems Engineering course, a feedback process is developing within the lab. Students gain initial project experience within the SDL, then take the Space Systems Engineering course and begin to realize the context of all of their previous work. In the end, students refocus their engineering design efforts in the SDL with respect to the "big picture" provided by the systems engineering course. In a sense, the feedback loop is developing a systems engineering culture within the SDL.

1.1 Paradox Mission

The Picosatellite for Autonomous Rendezvous and Docking on-Orbit eXperiment, or Paradox mission, is one of two missions currently being supported by the SDL. This mission will be used in this thesis to demonstrate the principles of systems engineering. Paradox is the second in the series of four collaborative picosatellite missions with Texas A&M University sponsored by NASA Johnson Space Center's (JSC) Lonestar Program. Specifically, Paradox is to demonstrate and evaluate enabling technologies for an autonomous rendezvous and docking mission for picosatellite-class spacecraft. The technologies include satellite-to-satellite crosslink communications, high-bandwidth ground communications, attitude determination and control, rela-

tive navigation, propulsion, and an imaging capability.

Significant mission design experience was obtained on a previous project in the SDL. The project known as Texas 2-STEP was a nanosatellite designed to perform autonomous rendezvous. The development of an autonomous rendezvous algorithm on Texas 2-STEP aided the Paradox mission.[3] In light of the purpose of the thesis - to connect the systems engineering practices with the design - the rendezvous research on Texas 2-STEP is revisited in depth with respect to the Paradox mission. Additionally, due to the mission focus on Guidance, Navigation and Control (GNC), the focus here will be on the GNC aspects of the Paradox mission and the GNC subsystem itself.

As a project within the SDL, Paradox is unique because of the recent inclusion of two small companies in the design process. Due to the increased technical complexity of the Paradox mission, NASA recognized early on the need to provide more support for the Lonestar Program; and thus, utilized the NASA Small Business Technology Transfer (STTR) program to infuse more technical expertise into the process. The STTR Program is a three-phase approach to develop technology in response to a specific set of NASA mission driven needs. The STTR program involves a research institution partnering with small businesses to develop new technology. The mission design team expanded to include professors, students and practicing engineers working in Phase I in partnership.

The SDL finds itself once again at the front-end of a project and challenged to "get it right". Systems engineering is the key factor. The Paradox project will be the test bed for the systems engineering processes described

for use in student-based design laboratories. The systems procedures will be demonstrated at multiple levels within the student project. Particularly, the GNC subsystem or GNC-related aspects of the mission will be used throughout the thesis as an example of applying the systems practices at the subsystem and component level.

1.2 Project Life Cycle

The Paradox mission development is underway. At the writing of this thesis, the design maturation is only at a conceptual level with a focus on technology development to enable the mission. A fundamental aspect of systems engineering is understanding the project life cycle. The project life cycle is the sequential categorization of everything that should be done to accomplish a project in phases separated by decision points or control gates. Although not covered in detail in this thesis, the project life cycle plays a key role in overall mission success. Briefly, the NASA Systems Engineering project life cycle is separated into seven different phases outlining the classic waterfall approach[1],

Pre-Phase A Conceptual Studies - Produces a broad spectrum of ideas and alternatives for missions from which new projects can be selected.

Phase A Conceptual Design and Technology Development - Determines the feasibility and desirability of a suggested new system.

Phase B Preliminary Design and Technology Completion - Defines the project in enough detail to establish an initial baseline capable of meeting mis-

sion needs.

Phase C Final Design and Fabrication - To complete the detailed design at every level of the system and fabricate or procure the hardware.

Phase D Integration and System Verification - To assemble and integrate the individual elements to create the larger system. To perform end-to-end testing and verification to ensure system will meet mission requirements. The phase is concluded upon launch.

Phase E Operations - The mission is conducted and the needs, goals and objectives are met.

Phase F Close-out - The mission is completed, and the system is properly disposed.

The systems processes must be actively engaged throughout the project life cycle and at all the various levels of the system development. The intent of Phase I in the STTR program is to develop the concept of a new system and perform technology development in accordance with work expected in Phase A of a project life cycle. As Paradox is the project example implementing the systems processes described in this thesis, the level of design maturation should be considered Phase A with respect to the project life cycle.

Phase A has a focus based on mission concept studies and technology development. Therefore, the following technical products for hardware and software system elements are expected by the end of the phase[18]:

1. System architecture.*
2. System requirements document.*
3. Updated concept of operations.*
4. Updated mission requirements, if applicable.
5. Preliminary system requirements allocation to the next lower level system.*
6. Technology development assessment.
7. Preferred system solution definition including major trades and options.*
8. Updated risk assessment and mitigations.*
9. Configuration management plan.
10. Verification approach.*

*The examples herein showcase these aspects of the Paradox mission in accordance with what is expected by the end of Phase A.

1.3 Thesis Organization

The thesis has a total of fourteen chapters. Chapters 2 and 3 further motivate the need for systems engineering practices in a student-based design laboratory and provide a basic definition of a student-based design laboratory. Chapter 4 describes systems engineering, technical engineering design and the fundamental connection between the two disciplines. The connection is primarily described through a "road-map" that outlines the input-output relationship between the systems engineering practices for student-based design laboratories and the technical engineering design.

Chapters 5 through 12 are the eight systems engineering practices for

a student-based design laboratory. Each chapter describes the purpose and extent of using the process within a student-based design laboratory as well as example(s) to characterize the process' use. Last, Chapter 13 highlights the importance of iterating the systems engineering processes within a project and Chapter 14 presents conclusions and summarizes the systems engineering processes for a student-based design laboratory.

Chapter 2

Motivation

The need for usable systems engineering practices within a student-based design laboratory derives primarily from the experiences of students as they moved through the entire system development life cycle of three spacecraft systems: FASTRAC, Texas 2-STEP and PARADIGM. For the past ten years, students have strived to design, fabricate and integrate small satellites in the Satellite Design Lab (SDL). The PARADIGM spacecraft, a picosatellite reached orbit in July 2009. The FASTRAC spacecraft, a nanosatellite, is manifested for launch in the summer of 2010. Texas 2-STEP was planned as an integrated spacecraft mission with a nanosatellite chaser vehicle and a picosatellite target vehicle. This was the first attempt to design a complex autonomous rendezvous mission in the SDL. Through each of these major projects, students gained valuable engineering skills in the design, fabrication and operation of actual flight satellites, but students also realized the need for the application of systems engineering principles and have tried to implement aspects of this discipline. The following is an overview of how each major satellite project in the SDL implemented systems engineering processes and subsequently the lessons learned at the end of each project.

2.1 FASTRAC

FASTRAC (Formation Autonomy Spacecraft with Thrust, Relnav, Attitude and Crosslink) was the winner of the University Nanosat-3 (UNP-3) competition held in January 2005 hosted by the Air Force Research Laboratory (AFRL). The purpose of FASTRAC is to investigate and demonstrate enabling technologies for satellite formations. FASTRAC has been manifested for launch by the US Air Force on STP-26 in the summer of 2010.

The FASTRAC team charted new territory for designing and fabricating a flight worthy satellite in a student-based design laboratory environment. The project encountered many of the classic problems experienced when systems engineering principles are neglected. For example, prior to the UNP-3 Flight Competition Review (FCR), the FASTRAC student team struggled with balancing time spent developing the satellites and time dedicated to writing the necessary documentation required by the UNP-3 competition. After winning FCR, AFRL required the team to implement configuration management for all documentation and to implement quality control measures for fabrication of flight hardware.[7]

In addition, the project underwent significant redesign in order to move from the engineering design unit to flight hardware. Almost every element on the satellite (electrical and mechanical) had to be modified to satisfy the AFRL standards. The redesign can be attributed to a misunderstanding of UNP-3 flight-build requirements, the onset of requirements creep and new requirements imposed by AFRL after the UNP-3 flight selection.[7]

As the first UT student-built satellite, the FASTRAC project success-

fully showed student-based design laboratories are capable of delivering flight hardware. However, the experience demonstrated to students the essential need for systems practices early on in the project life cycle to alleviate problems during flight hardware fabrication.

2.2 Texas 2-STEP

In support of its mission to perform autonomous, on-orbit, proximity operations with a rapidly producible nanosatellite, Texas 2-STEP developed technologies in small satellite sensor fusion, attitude determination, navigation filtering and control logic. Texas 2-STEP began in January 2005 as ARTEMIS immediately after FASTRAC was announced the winner of the University Nanosat-3 (UNP-3) competition. ARTEMIS (Autonomous Rendezvous and rapid Turnaround Experiment Maneuverable Inspection Satellite) was the UT entry in the University Nanosat-4 (UNP-4) competition from 2005 to 2007. The project was then reentered into the next competition cycle UNP-5 with a new name, Texas 2-STEP, and the same mission goals and objectives.

The ARTEMIS team implemented some systems engineering practices early in the life cycle. The project developed a high-level functional concept of operations, derived requirements to the subsystem level and created mass and volume budgets. Then as a separate task, the team began designing the satellite. The systems engineering practices implemented in the first two years were seemingly abandoned in the design and build phase. Requirements were not updated and became invisible to the subsystem engineers. Mass and power margins were not managed, and interfaces between subsystems were

never defined.

A primary reason that the early systems engineering was lost, was that the students who initiated and developed the mission were not the same as the students who performed the detailed design and fabrication of the engineering design unit. A major student turn-over after ARTEMIS and at the initiation of Texas 2-STEP left much of the initial design misunderstood and consequently reworked. The incoming student design team was focused on delivering subsystem hardware and insufficient time was spent on ensuring the design was integratable. The students recognized the need but lacked the basic understanding of systems engineering practices and failed to carry the practices throughout the project life cycle. This proceeded to be a major hindrance to success.

After the UNP-5 Flight Competition Review in January 2009, Texas 2-STEP was abandoned after not being selected. The primary reason for not continuing the project is due to the philosophy that student projects should not span more than four to five years. Therefore, students have a chance to participate in a student project from inception to flight before they graduate.

A "lessons learned" review was held immediately after the competition review. Systems engineering related problems were explored in the lessons learned exercise. These included scope, interfaces and resource management. The scope for Texas 2-STEP was too large for the SDL at the time and possibly too large for a student-based design laboratory. As opposed to the mission architecture devised for the NASA Lonestar Program, which is much more digestible for a student-based design laboratory for a similar broad mission goal,

Texas 2-STEP essentially attempted to do a fully autonomous rendezvous between two spacecraft from scratch. Subsystem interfaces were a key problem, which innately permeated the subsystem design causing significant system design deficiencies. Most students simply did not realize whether they or someone else should have incorporated the interface within the design. Heritage systems from FASTRAC were recognized in the design, but the necessary modifications were never carried through. Last, resource management, particularly the power budget, was not actively updated to the extent necessary to even prove whether the current design was feasible.

A positive outcome of Texas 2-STEP is that many of the technologies, lessons learned and personnel are now supporting the new projects in the SDL, specifically Paradox. The experience gained from Texas 2-STEP is invaluable to the incoming student teams.

2.3 PARADIGM

PARADIGM (Platform for Autonomous Rendezvous and Docking with Innovative GN&C Methods) was the first of the four missions outlined by the NASA JSC Lonestar Program. The goal is to perform an autonomous rendezvous and docking by the fourth and final mission. The series of four missions are to be incremental steps to reach the goal. For the first mission, each university contributed a picosatellite that was launched from the Space Shuttle STS-127 in July 2009. The mission was to collect GPS measurements from the NASA DRAGON GPS receiver as the first step in picosatellite navigation.

PARADIGM, similarly with Texas 2-STEP, focused early on systems

engineering. While the first year of effort was spent on properly scoping the mission and developing requirements, PARADIGM as-built lacked working requirements and mass or power resource budgets. The concept of operations was developed in parallel with the flight software and interfaces were only successfully handled because each student intimately knew the entire design.

PARADIGM had also no significant management structure aside from having a student program manager. Instead, the students collectively attacked design or fabrication problems as they developed. In one sense, the students acted as subsystem design, fabrication engineers and as systems integrators all at the same time. The project culture on PARADIGM is highly recommended because it forced each student to understand the complexities of every subsystem with respect to the entire system. However, a major problem was the lack of design verification and full system testing. The team performed extensive subsystem testing, but little in the way of full system testing, especially environmental testing.

As UT's first satellite in space, the mission was far from successful. PARADIGM did demonstrate the ability of a student-based design laboratory to deliver a flight quality satellite, but was unable to perform the mission because the UT and A&M satellites did not separate from each other. A detailed failure analysis pointed specifically to an overly sophisticated separation design and the lack of system environmental testing. In the end, time spent performing full system verification with the Texas A&M satellite may have uncovered the underlying design problems.

As each of these projects has demonstrated, there needs to be a focused

effort in implementing systems engineering practices throughout the project life cycle. A systems engineering culture needs to be developed within the student-based design laboratory, and the processes must continually be implemented.

Chapter 3

Student-based Design Laboratory Definition

A student-based design laboratory is composed of undergraduate and graduate student engineers with one or more student leaders. The purpose of a student-based design laboratory is for the students to design, build, test and deliver a final product to a customer. A student-based design laboratory is overseen by a faculty advisor whom is the Principle Investigator (PI) for projects within the laboratory and acts as a base for expertise and direction. However, design decisions are ultimately made by the students. Specifically at UT Austin, the faculty are exceptionally "hands-off" leaving all engineering and even purchasing decisions to the student leadership. Funding is primarily provided for hardware by faculty or external organizations. Some of the funding may support a few select students to provide a level of continuity and quality.

There are several characteristics that define a student-based design laboratory that make it distinctly different from a professional organization within the aerospace industry. The student-based design laboratory is characterized by a volunteer workforce, unusual time constraints, large student turnover rates, lack of experiences, innovation and a general dislike for bureaucracy. These six characteristics are defined as follows:

1. Volunteer Workforce A defining characteristic of a student-based design

laboratory is the (mostly) volunteer workforce. With the exception of technical elective course credit and possibly a handful of funded student positions, most students volunteer their time to a student-based design laboratory to supplement their academic education. The appeal of a student-based design laboratory is working on a project that moves from concept to operations. Unfortunately, a volunteer workforce is difficult to maintain and the 80-20 rule applies to a student-based design laboratory. That is, 20% of the team does 80% of the work. The rule often creates a "hero-environments"[7] where dedicated students perform most of the work.

2. **Unusual Time Constraints** As students at a university, volunteers are already committed to a full engineering curriculum and possibly other university affiliated extracurricular activities. Progress is often made in short spurts on the weekends, in between weekly homework assignments and mid-term examinations. Weekly productivity is not constant and, unfortunately, by the end of a semester, students are busy with finals and final projects. In addition, some students are eager and able to work during the summer, but most go home, participate in internships or other similar activities. The problem is aggravated since commercial contractors, vendors and government organizations supporting the project often do not consider the academic calendar when scheduling external reviews, site visits, product shipments, project funding or expert teleconferences.

3. **Large Turnover Rate** As expected, graduation is the primary reason for quality students to leave a project. Students graduate every semester

taking with them the knowledge and experience provided through the student-based design laboratory environment. Unfortunately, this occurs throughout the project life cycle and assumptions, design rationale, system testing results and more is often lost, leaving new students to perform a significant amount of rework. In addition, the cyclical turnover rate adds another task for incumbent students to recruit and replenish ranks at the start of each new academic semester.

4. **Lack of Experience** An aerospace project is a multi-disciplinary endeavor and students are often in the middle of their degrees as they work in student-based design laboratories. Without previous experience working on an actual project, many students have a steep learning curve. Different majors have different strengths and recruitment efforts purposely span multiple engineering and science disciplines to acquire a variety of skills necessary for the project.

5. **Innovation** Student-based design laboratories are often unencumbered with the past; thus, allowing creative concepts to emerge. External organizations often invest in student-based design laboratories to solve unique problems due to the innovative culture existing within student-based design laboratories.

6. **Rejects Bureaucracy** Students are very motivated to design, build and test their ideas, but are very disinclined towards documentation, meetings and other forms of bureaucracy that provide a level of quality assurance and continuity within the project. Consistent meetings are rarely

successful due to students' chaotic schedules. The tendency not to document work causes problems within a project because of the high turnover rate. Nevertheless, a student-based design laboratory does not require the level of overhead larger organizations need. Within a student-based design laboratory, the students act in multiple roles (such as designer and technician), which enables a student-based design laboratory to function without the high level of bureaucracy seen in industry.

Due to these characteristics that uniquely define a student-based design laboratory, the NASA defined systems engineering processes must be modified to better serve the students and their projects within this unique environment.

Chapter 4

The Systems Engineering and Technical Design Link

4.1 Systems Engineering

Systems engineering is a difficult discipline to define in a single sentence. The NASA Systems Engineering Handbook has many definitions including, "Systems engineering is a methodical, disciplined approach for the design, realization, technical management, operations and retirement of a system." and "Systems engineering is a holistic, integrative discipline, wherein the contributions of structural engineers, electrical engineers, mechanism designers, power engineers, human factors engineers and many more disciplines are evaluated and balanced, one against another, to produce a coherent whole that is not dominated by the perspective of a single discipline."[1]

Each definition takes a different viewpoint of systems engineering in the attempt to define systems engineering. Key aspects are often omitted in any definition. Basic topics encompassed in systems engineering are discussed here for student-based design laboratories. These include scope, systems hierarchy, interface management, requirements, trade studies, resource management, risk management, system verification and technical reviews. Other topics within the systems engineering umbrella include technical planning, functional analysis, technical decision analysis, cost analysis, reliability, product integration,

etc. In addition, the breath and depth of each of these topics spans multiple handbooks and technical journals, such as the Journal of Systems Engineering produced by INCOSE, addressing a variety of industries, not just aerospace. The sheer volume is a key reason to tailor these processes for the scope of a small student-based design laboratory given the unique environment described in Chapter 3.

In fact, the NASA Systems Engineering Handbook briefly addresses the topic of modifying systems engineering based on the size or scale of a project,

> The exact role and responsibility of the systems engineer may change from project to project depending on the size and complexity... For large projects, there may be one or more systems engineers. For small projects, sometimes the project manager may perform these practices. But, whoever assumes those responsibilities, the systems engineering functions must be performed.[1]

Each of the general systems engineering topics discussed in the thesis are described below,

Scope is the broadest definition of the program or project. Starting with the stakeholder's expectations, the project scope outlines the needs, goals and objectives for a project; establishing the basic architecture and developing the concept of operations for the resulting system.

System Hierarchy is the framework defining the functional aspects for each element within a system.

Interfaces defines the interrelationships between elements within a system.

Requirements is the process of translating the scope into a definition of the specific problem such that the end-product satisfies functions to a level of performance to achieve the mission.

Trade Studies are the high level technical analyses that support design decisions.

Resource Management maintains the resources shared between elements of a system such as mass, volume and power.

Risk Management identifies and mitigates potential problems of the design in order to meet the mission need.

System Verification acknowledges that the system designed, built and flown successfully satisfies all mission objectives and requirements.

Technical Reviews are the key decision points identified in the project life cycle to allow independent review of progress.

4.2 Technical Engineering Design

Technical engineering design is a broad topic spanning multiple disciplines that is not easily characterized. Design establishes and defines solutions to and pertinent structures for problems not solved before, or new solutions to problems which have previously been solved in a different way.[19] Every engineering discipline has its own analysis tools and design techniques to develop solutions to problems common within that discipline.

The ability to design is both a science and an art.[19] Systems engineering is also often expressed as both a science and an art.[1] The science aspect

of both engineering design and systems engineering can be learned through techniques and processes, like the systems processes outlined herein. However, the art can only be learned through performing systems engineering or design in the context of a realistic project.

Systems engineering is often considered the glue by providing discipline and continuity to the design effort. By essentially wrapping around the design and development of a project, systems engineering gives the design effort purpose and context within the overall needs for the end-product. While the thesis is not looking specifically at design principles or processes for student-based design laboratories, systems engineering is not independent from the design and development effort; and thus, cannot be treated independently. That is to say the systems engineering and engineering design are intimately connected.

4.3 The Ubiquitous Connection

In any basic analysis or design effort, "What is good enough?" is a question often asked. A common reply is, "Keep it simple; better is the enemy of good enough." Thus is the approach for systems engineering for student-based design laboratories. With limited personnel and expertise, students can often become inundated with the array of systems engineering tools one could implement in a project. Additionally, there is a natural tendency for students to focus on technical engineering design and analysis. In light of the basic definition, a student-based design laboratory is unable to support a large amount of systems engineering that is independent of the design, fabrication and testing of a product. Therefore, the primary aspect of the thesis is to merge

the systems engineering with the technical engineering design and analysis to simplify the overall process.

Fig. 4.1 depicts the road-map for the thesis, describing the general connections of each system engineering topic with respect to the technical design and effectively relating the basic systems processes directly to the design process. The road-map depicts the inherent iterative nature of systems engineering. Each numbered relationship is described in more detail as both an input and an output with respect to each systems engineering process. Based on the need of the reader, the following detail descriptions of the inputs and outputs will guide the reader from one systems process to another.

4.3.1 Scope

Inputs

1. Mission need and initial customer expectations provide initial purpose for the project.

21. As the system is partitioned, the architecture and concept of operations for each element is recursively evaluated to ensure each element satisfies the larger mission needs, goals and objective.

31. The requirements are validated against customer expectations to ensure the correct system is built.

41. Trade studies provide analysis for architecture decisions and concept of operations planning.

71. System architectures and concept of operations are evaluated for risks and mitigation strategies are implemented.

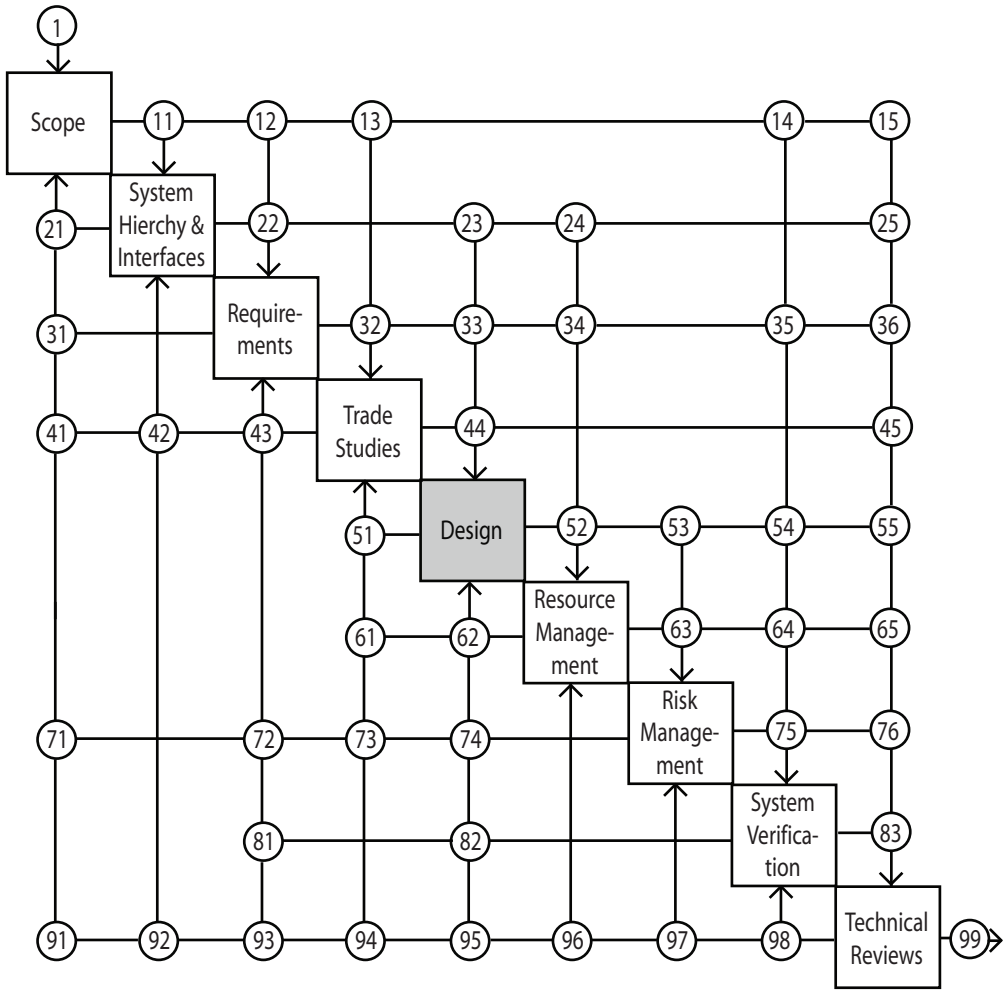

Figure 4.1: Showcases the inherent connections between systems engineering topics and the fundamental core of design for a student-based design laboratory.

91. Technical reviews, especially the Mission Concept Review, provide critical feedback into the feasibility and the overall mission plan at achieving stakeholder expectations.

Outputs

11. Architecture and concept of operations directly support the system hierarchy definition and interfaces.
12. The needs, goals and objectives provide the root for all requirements development. The concept of operations and architecture act as a source of requirements.
13. Various architectures are analyzed in more detail to determine the most viable option as well as identifying de-scope options as the fidelity of the design increases and grows past constraints.
14. Stakeholder expectations are a metric from which the design is validated against.
15. Mission Concept Review focuses on the scope of the mission, albeit all reviews focus on the refinement and fulfillment of the scope as the project steps through the life cycle.

4.3.2 Systems Hierarchy and Interfaces

Inputs

11. Scope identifies the overarching system to meet objectives and subsequently the system is decomposed into components based on the architecture and concept of operations.

42. Trade studies often provide technical rationale for partitioning the system into specific elements.
92. Technical reviews, particularly Mission Concept Review and Preliminary Design Review, provide feedback on the allocation of the system.

Outputs

21. The systems hierarchy identifies elements to a larger system, which feeds back into the Scope for architecture and operations concept development.
22. The systems hierarchy provides a clear structure for requirements flow-down. Interfaces are established and interface requirements are developed.
23. The hierarchy functionally divides the design into aspects providing a pseudo-work breakdown structure for a student-based design laboratory. Interface management defines where the interfaces are and who is responsible for them within the design and development.
24. The hierarchy allocates functionality and thus resources as the system is decomposed into pieces.
25. Mission Concept Review and Preliminary Design Review focus on the systems hierarchy and interfaces, although all technical reviews expect refinement.

4.3.3 Requirements

Inputs

12. Mission scope provides the mission needs goals and objectives from which all requirements are derived.
22. The systems hierarchy allocates functions to elements for which requirements are written. In addition, the hierarchy provides a clear structure for the organization of requirements.
43. Trade studies directly provide performance requirements, but also provide assumptions, constraints and rationales to support requirements.
72. Risk analysis verifies the design will meet the requirements and risk mitigation strategies often get translated into requirements to ensure mission success.
81. Failure to verify requirements will lead to requirement modification or waivers.
93. Requirements must be reviewed prior to Preliminary Design Review to ensure complete and internally consistent requirements for the system to be verified against.

Outputs

31. Requirements are validated against stakeholder expectations.
32. Performance requirements must be quantified by analysis to bound the mission design space.
33. The requirements directly provide the design space for a product to be realized.
34. Requirements bound the technical resources in the form of constraints.

35. The requirement verification matrix provides a method to verify the design is correct.
36. Requirements are a focus prior to Preliminary Design Review and requirement verification is a focus in any Test Readiness Review and the Flight Readiness Review.

4.3.4 Trade Studies

Inputs

13. Various architecture trades and studies on the feasibility of the concept of operations directly fall from the scoping exercise for a mission to provide analysis and thus rationale to refine the mission scope.
32. Performance requirements must be quantified by trade studies.
51. Design solutions feed back into trade studies to properly compare design options.
61. Resource margins for the designs accompany design options to compare.
73. Additionally, risk analysis enters into the trade studies to compare and select the most appropriate solution for the mission with the intent of minimizing risk.
94. Mission Concept, Preliminary and Critical Design Reviews provide assessments on active trade studies and evaluation of finalized trade study results.

Outputs

41. Architectural trade studies influence the scope and concept of operations.
42. Trade studies help determine the best engineering solution when partitioning the system and defining interfaces to represent it within the systems hierarchy.
43. Trade studies define performance requirements, justify assumptions and outline constraints.
44. Trade studies directly impact the design and development, inform component-level hardware selection and provide assumptions and constraints to the design.
45. Architectural trade study results are a priority in Mission Concept Review, while design related trade studies are the focus in Preliminary Design Review and Critical Design Review.

4.3.5 Design and Development

Inputs

23. The system hierarchy defines the full system down to the component level and interfaces for the design.
33. Requirements provide the definition of the design space. That is the design and development must satisfy the requirements.
44. Trade studies provide the analysis and rationale for design solutions.
62. Resource management characterizes and maintains the design and development to satisfy constraints.
74. Risk mitigation strategies directly translate into design modification to reduce mission risk.

82. Failure in the design of not meeting verification requirements involves redesign or requiring a waiver for the requirement that is not satisfied.
95. All technical reviews inherently evaluate the design and development of the project to ensure progress.

Outputs

51. The design and development effort feeds directly back into the trade study process as more information is provided to make a decision.
52. Design provides actual data for resource management. Resource management ensures proper resource allocation to meet constraints within the design and development.
53. Risk analysis identifies risks specific to a design.
54. The design and development must be properly documented and verified against the requirements to ensure mission success.
55. The design and development are the focus of every technical review.

4.3.6 Resource Management

Inputs

24. The systems hierarchy provides the structure for general allocation of resources between the elements of a larger system.
34. Requirements bound the technical resources in the form of constraints.
52. Resource management is directly performed on the design and development to ensure constraints are being met.

96. Preliminary Design Review, Critical Design Review and Flight Readiness Review provide essential feedback with respect to the resource margins.

Outputs

61. Resource budgets provide a platform to trade options and make the best decision with respect to the entire system.
62. Constraints in resources directly affect the design and development in a system by restricting the design space.
63. There is an inherent risk for not managing resources properly, thus analysis on resource margins inform the risk to the mission.
64. Resource budgets must be properly documented and are used to verify system requirements.
65. Resource management is important in Preliminary Design Review, Critical Design Review and ultimately the Flight Readiness Review as verification the system meets constraints.

4.3.7 Risk Management

Inputs

53. The design is a direct input for risk analysis.
63. Minimal margins from resource management is often a direct indicator of design risks, and risk analysis must ensue to mitigate the risk of the system not meeting constraints.
97. Technical reviews throughout the life cycle provide feedback on risk analysis and mitigation strategies.

Outputs

71. Risk assessment of the design within the context of the architecture and concept of operations will feedback into the scope of the mission with the intent to re-scope the mission to reduce risk.
72. The result from risk analysis often requires the mitigation of the risk within the design and thus, the mitigation becomes embedded in the requirements.
73. Risk analysis results often feed into trade studies to make an informed decision.
74. Mitigation strategies directly feed into the design and development effort.
75. Major mission risks should be documented and mitigation strategies should be tracked. In addition, risk analysis drives the level of testing required to verify mission success.
76. Mission risks are initially reviewed in the Mission Concept Review. Risks are then consistently reviewed at Preliminary Design Review, Critical Design Review and then Flight Readiness Review to ensure successful mitigation throughout development.

4.3.8 System Verification

Inputs

14. Scope provides stakeholder expectations for the design to be validated against.
35. Verification is performed with respect to the requirements, thus the

requirements verification matrix is actively used during the verification process.

54. The design and development is verified against the requirements through analysis, demonstration, inspection and/or testing.
64. Resource management provides quantitative analysis that the design is verified with respect to technical resource constraints.
75. The risk analysis identifies verification methods as mitigation strategies for alleviating mission risk, such as a test to demonstrate antenna deployment in a relevant environment.
98. Technical reviews particularly the Test and Flight Readiness Reviews act as an independent verification that the system has been properly verified against the requirements.

Outputs

81. Failure to properly verify the system against the requirements leads to modification or waivers of the requirements.
82. Failure to properly verify the requirements leads to redesign.
83. System verification is a focus in any Test Readiness Review needed and the Flight Readiness Review to ensure mission success.

4.3.9 Technical Reviews

Inputs

15. Scope is primarily evaluated in the Mission Concept Review.
25. The system hierarchy and interfaces are reviewed primarily at the Mission Concept Review, Preliminary and Critical Design Reviews.

36. Requirements should be constantly under review between Mission Concept Review to the Critical Design Review.
45. Trade studies are reviewed at the Mission Concept Review, Preliminary and Critical Design Reviews.
55. The design is evaluated at every technical review.
65. Resource analysis is primarily focused upon during Preliminary and Critical Design Reviews to ensure healthy margins exist against imposed constraints.
76. Mission risks are evaluated and watched from the Mission Concept Review to the Flight Readiness Review to ensure successful mitigation.
83. System verification is important in the Flight Readiness Reviews.

Outputs

91. Mission Concept Review and subsequent reviews provide feedback to the architecture and concept of operations of the mission.
92. Mission Concept Review and Preliminary Design Review evaluate the hierarchy and interface definition.
93. Requirements and requirements verification must be constantly reviewed, focused on prior to Preliminary Design Review.
94. Architecture trade studies are evaluated at the Mission Concept Review and detailed trade results are reviewed in Preliminary and Critical Design Reviews.
95. The maturity of the design and progress in development is inherently assessed in every technical review.

96. The design in the context of meeting resource constraints is a focus in Preliminary and Critical Design Reviews to ensure mission success.
97. Risk analysis and mitigation strategies are reviewed at Mission Concept, Preliminary and Critical Design Reviews to ensure proper mitigation by Flight Readiness Review.
98. Flight Readiness Reviews confirm that the product is successfully verified.
99. At the end of the Flight Readiness Review the final product is confirmed to satisfy all expectations, objectives and is properly verified for mission success.

Chapter 5

Scope

Scope is the broadest definition of the program or project: identifying stakeholders; providing the need, goals and objectives for a project; establishing the basic architecture and developing the concept of operations for the resulting system. Properly defining the scope of a project or program, establishes a firm foundation upon which a design solution is fabricated, integrated and operated to ultimately satisfy stakeholder expectations.

5.1 Stakeholders

A stakeholder is a group or individual who is affected by or is in some way accountable for the outcome of an undertaking. A customer is a direct beneficiary of the work, while other interested parties affect the project by providing broad constraints within which the customers' needs must be achieved.[1] The identification of stakeholders and their expectations is the first step in the systems engineering process. The primary purpose of identifying stakeholders and their expectations is to ensure a mutually exclusive set of goals and objectives for which a project team is focused on achieving with the intent of satisfying the customer.

For student-based design laboratories, a basic set of stakeholders on any project includes external organizations (both governmental agencies and/or

private companies), faculty advisors and, of course, the students. The interesting distinction in a student-based design laboratory is which stakeholder is considered the customer. From the students' point of view, the external organizations are the customers for which a student-based design laboratory is to deliver a product to satisfy the technical or scientific expectations outlined by the external organizations. In this case, the students are stakeholders interested in the technical challenges or content within a project.

However, from the external organizations' point of view, a student-based design laboratory is not necessarily contracted to deliver a product. The primary expectation is educational. The development of an actual system that satisfies a specified technical goal is often considered a bonus, not necessarily an expectation. In this case, the students are actually the customers benefiting from the educational experience with the external organizations only providing a basic goal for the students to work towards and some level of mentoring. Thus, the external organization is simply an interested party. In either case, the faculty advisor is also an interested party with a purpose to provide expert guidance directly to the students and act as an interface for the external organizations.

In light of the educational standpoint, the stakeholder-student relationship is a primary difference between student-based design laboratories and industry because a student-based design laboratory is not contracted to deliver an end-product. In addition, the lack of technical expectations on behalf of the external organizations leads to ambiguous mission goals and objectives. Instead, an external organization often provides a vague need or goal, and al-

lows the student-based design laboratory to satisfy the need in whatever shape or form it desires. Understanding this duality in the stakeholder-student relationship is crucial to how a student-based design laboratory must derive the mission goals and objectives.

5.2 Needs, Goals and Objectives

The stakeholders' expectations specify what is desired as an end state or as an item to be produced and thus directly identify the need for a project and derive the goals and objectives for a project.[1] A need explains the rationale for the system. A goal is the fundamental aim of the project. The objectives are measurable outcomes necessary to satisfy the mission need.

A student-based design laboratory actually plays a unique role in identifying a need and defining the mission goals and objectives. Currently in the aerospace industry, the team that identifies a need and subsequently outlines goals and objectives is often not the same as the team whom satisfies the need. For example, the US government and NASA recently identified the need to develop a new crewed exploration vehicle and defines the goals and objectives for the vehicle, but most of the design and fabrication work is contracted to private companies with NASA performing the systems engineering and acting as a very specific customer with insight and oversight of the project.

Due to the educational aspect, student-based design laboratories perform both roles since they are often only provided with a general need from an external organization, but are left to outline specific goals and objectives necessary to actually design and build a system. The process of converging

on a set of objectives for a mission is by no means a trivial task and must be understood by students to be the most important part of designing the system. An agreed upon set of objectives to satisfy a need is a powerful force providing a clear direction for a team of students.

Since student-based design laboratories have considerable influence over the scope of a project (given that it satisfies the external organizations' general need) it is important for students to define a mission that is both interesting and challenging, but is within the laboratory's ability considering the constraints of a student-based design laboratory, particularly limited personnel, budget and schedule.

5.3 Example: Paradox Need, Goals and Objectives

The need for the Paradox mission is derived from the NASA JSC Lonestar Program. NASA outlined a series of four individual missions to make significant steps towards an automated rendezvous and docking of two satellites, but said little about the necessary step for each mission in order to achieve the broad goal by the fourth mission. Often for student-based design laboratories, high level functional requirements or even objectives are not provided from the customer and must be developed by the students. Thus, the specific need for the second mission is allocated from the broad Lonestar Program need to perform an autonomous rendezvous and docking.

> *To demonstrate and evaluate enabling technologies for an autonomous rendezvous and docking mission.*

From the need, students derived general mission goals that focused on the rendezvous aspect of the mission, leaving close proximity and docking technologies for the third mission. The rationale was that a rendezvous must occur successfully first. Thus, the three mission goals for the second mission are,

1. ***Develop and test on-orbit sensor and actuator technologies.***
2. ***Develop and test target sensing technologies.***
3. ***Perform a rendezvous.***

To achieve the outlined mission goals the specific objectives are outline below. The mission shall,

1. ***Evaluate sensor suite performance.***
2. ***Evaluate actuator suite performance.***
3. ***Evaluate guidance, navigation and control capability in performing a rendezvous.***
4. ***Establish a communication crosslink between two satellites.***
5. ***Evaluate a high band-width communications groundlink.***
6. ***Evaluate imaging capability.***

5.4 Concept of Operations

The concept of operations describes how the system will be operated during the life cycle phases to meet stakeholder expectations, mission goals and objectives. It describes the system characteristics from an operational

perspective to facilitate an understanding of system goals and to develop requirements.[1] The operations concept should have a broad appeal because the audience will have a wide range of technical and managerial backgrounds. The operations concept should tell a story or a series of stories. Graphics, functional flow diagrams and timelines should be extensively used to describe the concept of operations from multiple points of view. The concept of operations must encapsulate both the users of a system, such as ground operators, and the end-users of the mission results. A single graphic will never have the capability to capture the full operations concept of the system nor will it properly reflect all aspects for the diverse audience.[12]

A critical component to the scope of a project, and perhaps where most of the time on project scope is spent, is defining a concept of operations where the end-product is realized and satisfies the mission needs, goals and objectives. The student-based design laboratory must work to produce a high fidelity operations concept that is understood by every team member and involves the customer as much as possible. The operations concept must be viewed by the students as an evolving document subject to change as more information is gained throughout the life cycle. More information consistently updates the operations concept by further defining execution sequences along a timeline and often directly translating into software architecture. The operations concept will be referenced through the design period and validated during the subsystem and system testing alike.

5.5 Example: Paradox Concept of Operations

The Paradox concept of operations is represented in many different forms, each conveying similar information in different ways for different audiences. A general or high level operations concept is necessary to express the need and objectives of a mission. At the same time, a very thorough explanation of the operations concept is necessary for design efforts. The following example shows both the high-level Paradox concept of operations and a relevant excerpt of the very detailed Paradox mission plan.

5.5.1 General Operations Concept for Paradox

Through an initial scoping exercise to define the concept of operations that satisfies both the objectives identified and the expectations from NASA, the nominal operations scenario is described in Fig 5.1. Graphics are exceptional communications tools to ensure students are working under the same concept, especially in the early design phases. A basic graphic begins the process of further definition of an operations timeline and functional analysis of the system itself.

As a brief explanation, the UT Austin and Texas A&M satellites are expected to launch from an expendable launch vehicle (currently unspecified), separate and initialize according to the launch vehicle requirements. The satellites will first attempt to establish crosslink communications. Unexpected separation dynamics between the two satellites and unexpected range capabilities for the crosslink technology present great challenges. After demonstrating the crosslink, the UT satellite will activate and evaluate each subsystem and

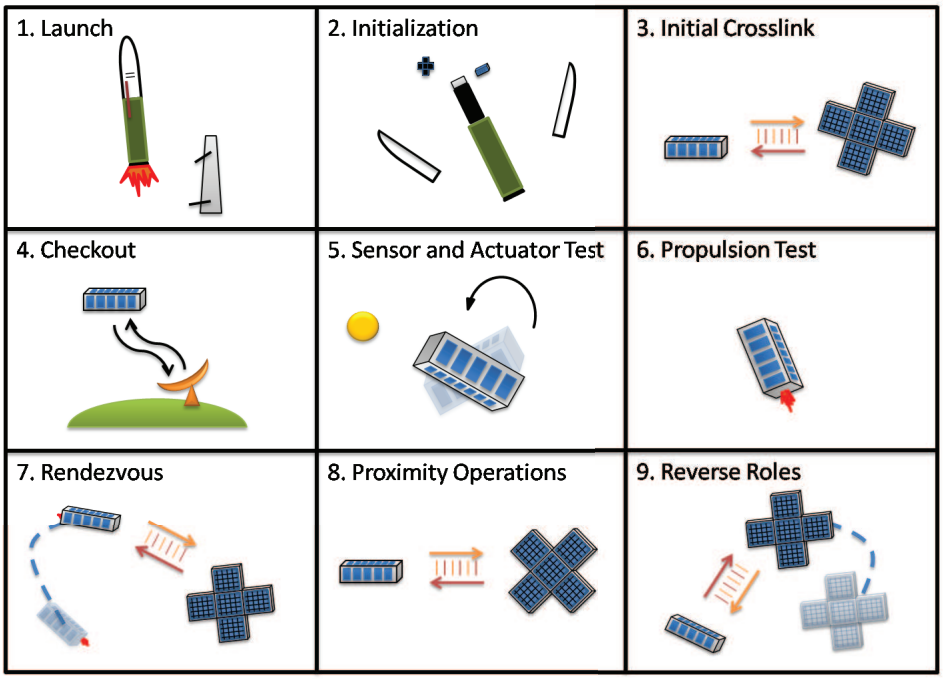

Figure 5.1: Pictorial Concept of Nominal Operations for the Paradox Mission.

establish a high bandwidth ground communications to relay the health data.

The ground will confirm all-subsystems-go and command the UT satellite to perform a series of increasingly difficult actions to evaluate the sensors, actuators, propulsion subsystem and guidance, navigation and control (GNC) algorithms. After each action, the satellite will downlink pertinent data for the ground to analyze and command the next action. Once the satellite's basic capabilities have been proven, the ground will command the satellite to initiate the rendezvous sequence with the Texas A&M satellite. If crosslink communications were discontinued, the two satellite will have to reestablish crosslink communications before initializing the rendezvous sequence. The UT

satellite is called the "chaser" and the Texas A&M satellite in this case is the "target". Then, the roles will be switched and Texas A&M will perform a rendezvous sequence as the chaser with respect to the UT satellite.

However, upon further evaluation of stakeholder expectations and due to the results from the PARADIGM mission, there were increasing concerns about the level of interfacing required to perform the nominal operations scenario described above. Recall, it was the separation interface between the UT Austin and Texas A&M satellite that caused mission failure. Therefore, two further operations scenarios were developed that still satisfied the mission objectives, but substantially alleviated the need for an interface with Texas A&M. Often called "open architecture", Paradox will be required to meet any of the following off-nominal target operations scenarios.

The first off-nominal scenario is a UT-based architecture whereby the Texas A&M satellite is replaced by a small passive target satellite also developed by the UT student-based design laboratory. Such a design choice would allow for complete control and compatibility of the target satellite. Thus, the crosslink, rendezvous and proximity operations will be carried out with respect to the UT target satellite. The second off-nominal scenario should be regarded more as a contingency scenario to satisfy the mission objectives if the capability to rendezvous with a physical satellite is not possible. The scenario is due to analyses that suggest both a Texas A&M or UT target satellite may drift too far before a rendezvous sequence can be initiated. In such an event, the UT satellite will perform what will be termed a "state" rendezvous. A state rendezvous is defined as a rendezvous with a phantom position and velocity in

space, which is similar to how a rendezvous would be carried out with respect to a physical target.

A much more general concept of operations was developed to showcase these three operations scenarios in the form of a functional flow diagram. The three different operational scenarios each had a common set of operations; thus, Fig. 5.2 showcases the general operational functions for each scenario on the left, and on the right the diagram distinguishes the difference between each scenario.

5.5.2 Detailed Operations Concept

While expressing the mission concept of operations pictorially or in a functional flow diagram form is important, there is still a strong need to elaborate on the details of each mission phase and mode, especially to indicate the functions of each subsystem during an operational mode.

Within each mission phase, there is a sequence of mission modes. However, the Paradox operations concept is designed where some of the mission modes appear within multiple phases, such as the mission mode: Ground Communications. After each mission phase the satellite will downlink pertinent data and await for the command to continue to the next phase or repeat the previous mission phase. One mission phase of the Paradox operations concept was chosen to show the level of detail achieved in the current iteration.

Figure 5.3 is an excerpt of the current Paradox operations document, specifically the Actuator Evaluation Mission Phase, which is one of many mission phases depicted in the nominal operational sequence in Fig. 5.2. Within

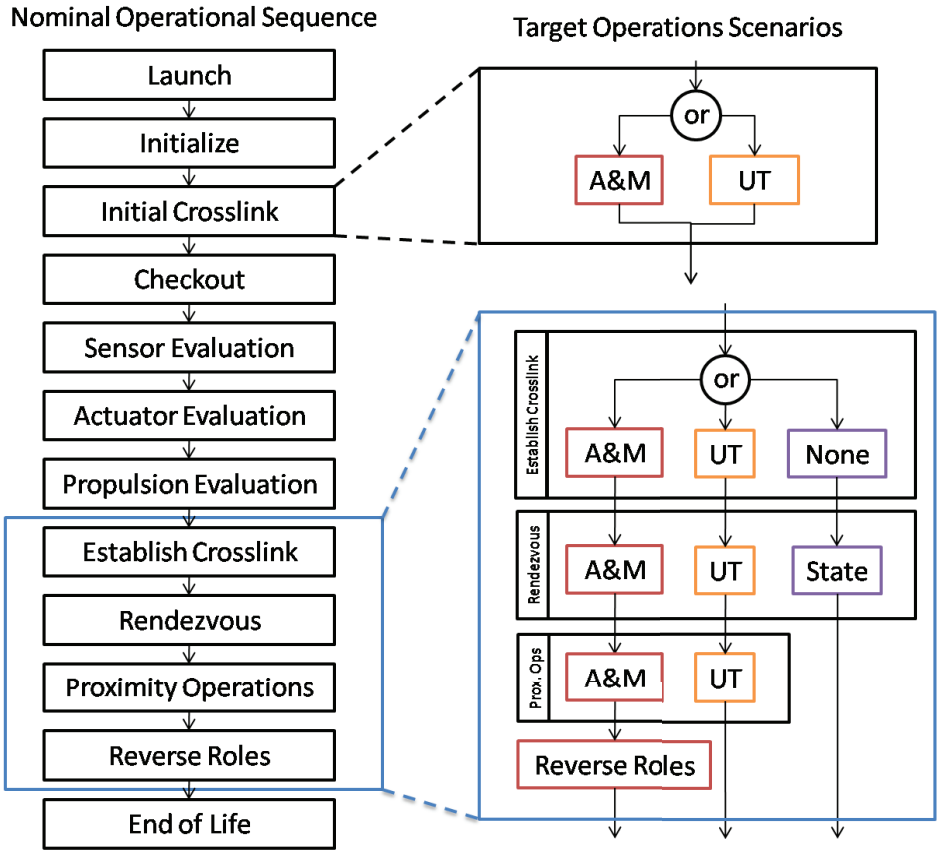

Figure 5.2: General concept of operations indicating the three different mission scenarios for the Paradox mission.

the mission phase, the excerpt shows three mission modes. For each mission mode, there is a set of five data fields to further define the concept of operations of the previous pictorial descriptions.

The key aspect to representing the concept of operations in this manner is to directly map the mission operations to the functional operations of the subsystems. As indicated by the "Subsystems" field, every powered subsystem is called out with its associated functional operations mode. These subsystem

functional modes are further described elsewhere in the Paradox concept of operations, but for example, "CDH-1" indicates specifically that the Command and Data Handling subsystem executes current mission mode and/or ground commands, schedules events, manages all mission data. Such fidelity in the concept of operations cannot be accomplished without first iterating through most of the systems engineering practices outlined within the thesis. Thus, Fig. 5.3 showcases a concept of operations after multiple iterations in the Paradox mission design.

3.4 Actuator Evaluation

3.4.1 Rotation Series

Rationale	Partial satisfaction of mission objective.
Trigger	Ground command.
Subsystems	CDH-1 EPS-1 GNC-2
Description	To initially evaluate the attitude actuators, the satellite will be commanded through a series of predetermined rotations.
References	None.

3.4.2 Sun Imaging

Rationale	Partial satisfaction of mission objective.
Trigger	End of previous mission mode.
Subsystems	CAM-2 CDH-1 EPS-1 GNC-2
Description	To further evaluate the attitude actuators, the satellite will be commanded to point the camera at the sun for imaging. The external reference will independently verify the sensor and actuator performance.
References	None.

3.4.3 Momentum Management

Rationale	Mission support.
Trigger	End of previous mission mode.
Subsystems	CDH-1 EPS-1 GNC-3
Description	To unload momentum stored in the reaction wheels and prevent saturation for mission operations. The mode will be initiated after end of a previous mission mode or unless otherwise directed by the ground or on-board momentum management system.
References	None.

Figure 5.3: Section of the Paradox Concept of Operations detailing three different mission modes within the Actuator Evaluation mission phase.

5.6 Architecture

The concept of operations gives the proper perspective for which the system-level functional requirements are realized. That is, from the mission objectives flow derived functions for the system to perform, and it is within the concept of operations where the system functions collectively act to satisfy the mission need. Without such a construct, a capability of any system is just a trivial artifact. Thus, the concept of operations gives rise to a system architecture for which the abstract notion of high-level functions translates into a conceptual hardware solution. A system architecture is the overall design or structure of a system, including the hardware and the software required. Defining the system architecture is the link between needs analysis, project scoping and functional analysis and the first descriptions of the system structure.

A primary aspect of a student-based design laboratory is its innovation. Students often do not know how something has been achieved before and therefore devise new approaches to a problem. Besides providing an educational experience, external organizations are often attracted to student-based design laboratory for their innovation. Specifically, the SDL as an engineering laboratory, focuses on technology demonstration missions as opposed to missions that support a scientific payload. That is, all of the missions to date have a need to demonstrate enabling technologies that will be used in future missions. In the context of the SDL projects, the mission architecture studies are primarily driven by what and how much enabling technology the laboratory is interested in supporting for a single mission. On the other hand, missions supporting a science payload are less inclined to demonstrate new technolo-

gies that may jeopardize the science. Unless the science payload requires new supporting technology the mission architecture is not focused on technology, but on the composition of existing technology to achieve the science goals.

For technology demonstration missions, a system architecture is perhaps the best tool for student-based design laboratories to evaluate the technological aspects of a project. For example, a concept of operations may require that the system provide a communications capability to a ground station. Would the student-based design laboratory like to purchase a commercial off-the-shelf (COTS) product to satisfy the high-level function or would the students like to develop a new technology, say a software-defined radio? Much of these decisions will be based on the expertise and interests of the student-based design laboratory, but review of the mission objectives and concept of operations might require such technology developments. It is then incumbent on the students to evaluate what technology developments are necessary given the mission and what can be achieved by integrating COTS components. Architecting a system involves making such decisions early in the life cycle.

A simple and valuable metric by which to judge technologies being implemented in a fledgling mission architecture is the Technology Readiness Level (TRL) Scale developed by John C. Mankins[13]. The scale measures the maturity of a specific technology. The range goes from a TRL 1 indicating that basic principle have been observed to a TRL 9, which means the technology has been demonstrated in an operational environment successfully. The full scale will not be explicitly stated here, but will be used. When constructing an architecture for a mission, the TRL scale is an extremely useful metric to

initially judge design solutions with the primary function of keeping the project scope in the realm of possibility. Often student-based design laboratories want to develop a whole range of new technologies, but it is important to decide on only a few technology developments per mission. Using the TRL scale when architecting a mission will provide a clear rationale for developing the certain technologies for the mission.

5.7 Example: Paradox GNC Architecture

5.7.1 Lonestar Program GNC Architecture

The Lonestar Program architecture is defined as the technology developments chosen over the remaining two to three missions to achieve the automated rendezvous and docking by the fourth individual mission. The individual mission architectures are defined as the technologies that will be implemented as a solution to the program architecture. Due to the nature of the NASA Lonestar Program, the Guidance, Navigation and Control (GNC) subsystem will drive mission success and thus will be the focus of this example. Keep in mind, it is likely many more technologies will need to be developed over the mission series and the architecture for each will need to balance the technology development between the GNC subsystem and other subsystems.

The Lonestar Program architecture, as defined by the SDL, is such that an automated rendezvous and docking could be successfully achieved by the third mission, but in the event the goals are not fully satisfied, a fourth mission can be used as a risk mitigation strategy to effectively achieve the customer's expectations. Then, the GNC architecture for Paradox must be

viewed in context with respect to the spread of the technology development over the series of missions. Fig. 5.4 depicts the GNC technology architecture for the Lonestar Program.

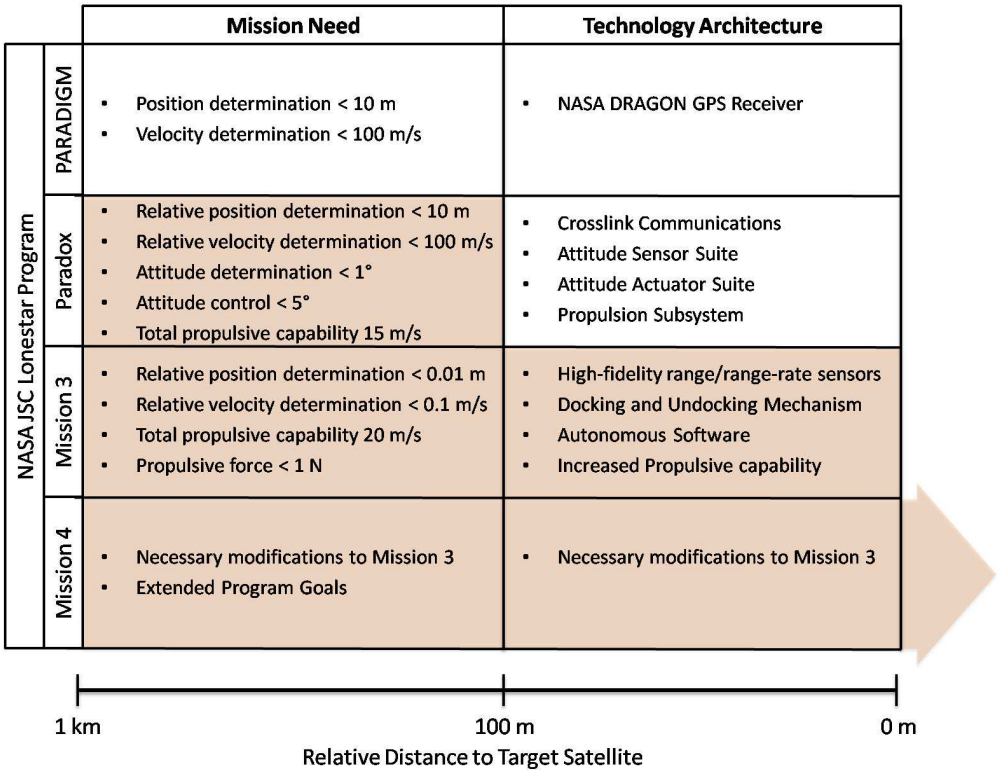

Figure 5.4: Lonestar Program GNC architecture to achieve rendezvous and docking.

Each mission takes steps to develop the necessary technology towards the automated rendezvous and docking. Expanding on PARADIGM's goal to test the NASA DRAGON GPS receiver for picosatellite navigation, Paradox will implement attitude determination and control, relative navigation, semi-autonomous flight software and propulsion capabilities determined as a necessary progression towards satisfying program goals. Thus, the Paradox

sensor suite will be composed of the DRAGON GPS receiver as well as other external and inertial sensors for attitude and relative state estimation. An actuator suite will handle attitude control with momentum exchange devices, and the propulsion subsystem will enable the satellite to change its orbit. In addition, on-board flight software will need to determine the position, velocity and attitude given the sensor measurements, determine what maneuvers to make and when, and then command the actuators to achieve the desired maneuvers.

Mission 3 implements the remaining technologies necessary for an autonomous rendezvous and docking, including the docking and undocking mechanism, high-fidelity relative navigation, full-autonomous flight software and precision propulsive capability.

Mission 4 exists to address extended mission goals from NASA, or to fulfill Lonestar Program mission goals not demonstrated in Mission 3. Thus, the fourth mission will either incorporate a different architecture with different technologies depending on a new set of mission goals, or will be composed of a similar GNC architecture as Mission 3. Nevertheless, the Mission 4 architecture will need to be properly modified depending on the results of Mission 3.

5.7.2 Mission 2: Paradox GNC Architecture

The Paradox team must decide how to architect the second mission to satisfy the overall Lonestar Program GNC architecture. One possible mission architecture is to develop all of these technologies from scratch. Sensors

and actuators would be designed and fabricated specifically for the mission along with the propulsion subsystem and the flight software. Another possible architecture is to purchase any or all of the technologies, given their availability. Classified as a "make-buy" decision, these are often a crucial aspect of architecting a system for a technology demonstration mission. A student-based design laboratory must make a "make-buy" decision with respect to the available laboratory resources, such as personnel, "in-house" expertise and schedule, to ensure the technology development is within the capability of the laboratory. These decisions are often made by the student leadership with extensive input from faculty.

The SDL as a student-based design laboratory is much more interested in designing and integrating a system to satisfy a mission than developing individual technologies, especially if there are commercially available technologies on the market. The current personnel level and schedule suggest that the SDL has the resources to perform only a few technology developments from scratch. In addition, the aerospace department at UT has an expertise in developing GNC algorithms for flight software, but does not have experience designing and fabricating flight quality sensors and actuators. Thus, Paradox is adopting the strategy to purchase as much commercial hardware as possible while designing the flight algorithms.

The primary reason for the decision is due to the technology readiness of the available hardware. Student-based design laboratories still must show technology development to a TRL of 6 (Demonstration of model or prototype of the technology in a relevant environment[1]) by Preliminary Design Review

(see Chapter 12) as suggested by NASA for all projects. Any technologies developed from scratch within the SDL would currently have a TRL between 2 to 4 (Breadboard validation of a technology in a laboratory environment[1]). Thus, more time and personnel would be needed to properly raise the TRL before flight. An online hardware search recognized a handful of individual sensors and actuators with flight heritage and a few sensor and actuator packages that fit the constraints. Thus, the sensor and actuators found commercially have a TRL from 5 to 7.

On the other hand, there are few commercially available propulsion subsystems for the desired application. At best there is an array of components from which a significant design effort would be required. However, a significant technology development from The Aerospace Corporation was found, but required a modification. In fact, the Aerospace Corporation's propulsion system was flight tested, but due to the necessary modifications the TRL would only be a 7 (Technology demonstration in an operational environment).[8]

The GNC architecture for Paradox resulted in performing significant technology developments in flight software development and the propulsion subsystem. The sensor suite and attitude actuation suite will be composed of commercial products, thus the work is reduced to just integration and testing.

Given the broad goal of the Lonestar Program, the SDL architected the technology developments necessary to achieve the NASA's expectations over the allocated number of missions. Then, given the program architecture allocated to Paradox, the specific GNC architecture was determined using the TRL scale as the primary metric. Using the TRL scale in this manner is

an effective way for a student-based design laboratory to make architecture choices with respect to available resources.

Chapter 6

System Hierarchy and Interfaces

6.1 System Hierarchy

NASA defines the system hierarchy or the product-breakdown structure (PBS) as the framework and interrelationships of elements in a system.[1] The concept of operations and the architecture development from the mission scope defines the overarching approach to satisfy the mission, including major system elements and technology. Thus, scope is a direct input to the systems hierarchy. The base of the hierarchy defines the final products, ie. hardware components or software deliverables, that are integrated to create the final system to meet mission objectives. Alongside the system hierarchy, NASA defines a work-breakdown structure (WBS) as the hierarchical breakdown of the work necessary to complete the project.[1] Naturally, the WBS contains the system hierarchy as the full product to be delivered, but also includes the work necessary to integrate and produce a working system, such as management, system engineering, testing, etc.

Student-based design laboratories do not have the personnel to dedicate solely to management. Identified managers, systems engineers, or other student leaders are often simultaneously the most proficient at design and technical analysis. Thus, the leadership will find themselves developing designs, as well as managing requirements, students and the budgets.[7] A well developed

system hierarchy will identify students with specific deliverables and simultaneously acts as a WBS. All the work must directly support the development of the system as all the tasks are connected to the deliverables defined in the hierarchy. Students are expected to perform design work, trade studies, create software, fabricate prototypes and perform testing in pursuit of the final deliverable identified in the system hierarchy.

Unlike large aerospace organizations, student-based design laboratories can not compartmentalize work functions, such as design, analysis, testing and fabrication. Instead, the same student that designs a component or subsystem must also perform the proper analysis and testing before ultimately fabricating the final design and participating in integration. In this way, a student-based design laboratory has a very flat personnel hierarchy and recognizes few student leaders. As a simple visual description of the entire system, the system hierarchy for a student-based design laboratory defines the deliverables to produce, the relationship between the deliverables, the work to be performed and the student(s) to perform the work. The identified student leaders are expected to work on the technical engineering design throughout the system hierarchy.

The systems hierarchy, when viewed as a depiction of the product system and the work for a student-based design laboratory, is constantly evolving. Herein lies an intimate connection between the technical design and systems engineering. Note, it is not desirable to fully define the system hierarchy initially because it will stifle the student creativity. Instead, it is preferred to allow the student engineers to define the component level and update the hi-

erarchy to reflect progress. Again, the system hierarchy is expected to evolve through the project life cycle.

6.2 Example: Paradox System Hierarchy

As a basis for both the design solution and a work-breakdown structure, the top-level system hierarchy for Paradox is depicted in Fig. 6.1,

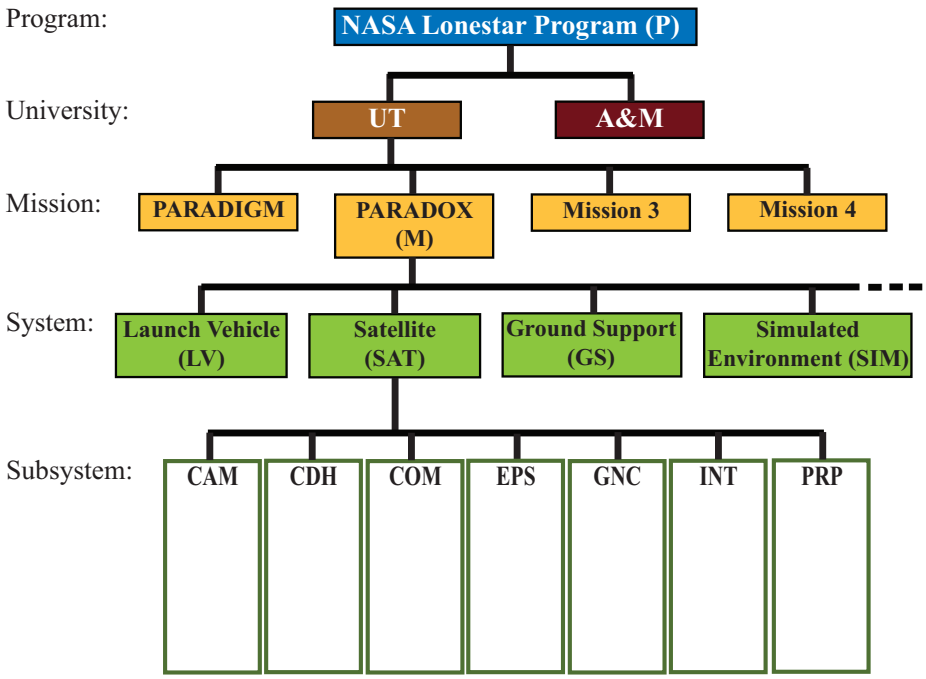

Figure 6.1: System Hierarchy for the Paradox Mission.

Beginning with the customers's definition of the mission, Fig 6.1 defines the hierarchy through the to satellite subsystems (acronyms are defined in Table 6.1). For Paradox, there is an additional hierarchical level entitled "University" due to the nature of the Lonestar Program. It is understood that Texas A&M will have a similar system hierarchy. In addition, for the

Paradox mission there are other identified systems besides the satellite. Not all of the systems supporting the Paradox mission could be easily represented in Fig. 6.1.

The launch vehicle includes the launch vehicle separation device, known as the Poly-Picosatellite Orbital Deployer (PPOD), which integrates the satellite with the launch vehicle and deploys the satellite after launch. The ground support system will be simultaneously designed with the satellite for both electrical and physical support, as well as communications support as a ground station once the satellite is on-orbit. The simulated environment is separately identified due to the nature of the Paradox mission. Heavy simulation testing will be required to verify the flight software and an accurate environmental model will be necessary to test expected conditions.

The satellite subsystems shown in Fig. 6.1 are defined in Table 6.1.

Table 6.1: Paradox satellite subsystem definition.

CAM	Camera
CDH	Command and Data Handling
COM	Communications
EPS	Electrical Power Subsystem
GNC	Guidance, Navigation and Control
INT	Integration (Structures)
PRP	Propulsion

The three letter acronyms are by convention, and are useful for configuration management and consistent documentation (see Chapter 11). Note, the Integration subsystem (INT) is classically the subsystem for structures and mechanisms found on most satellite hierarchies. The choice to rename the

subsystem Integration is due to a common misconception students have about the work related to the "structures" subsystem. Often students only focus on designing a structure that can sustain launch stresses and vibrations. In the end, students forget to mount hardware components and ultimately a re-design is necessary. As the Integrations subsystem, students understand that it is their first priority to design structures and mechanisms around the other subsystems' hardware components and then prove the structure will survive launch.

The satellite subsystems are separated in this manner because much of the design and fabrication can be performed independently of the other subsystems as long as the interfaces are managed. Each student is responsible for developing the satellite system to satisfy the mission, but individual students are responsible for the different subsystems as noted by the hierarchy. For student-based design laboratories, the system hierarchy identifies deliverables and the students responsible for specific deliverables, but it is implied that the students are also responsible for the technical analysis to support the design, prototyping, fabricating and testing necessary for the deliverable.

To illustrate the continued breakdown, Fig. 6.2 represents the GNC subsystem hierarchy and depicts the further delegation of tasks to the student team. Further iteration is needed to refine the system hierarchy, but at the base of the hierarchy will be all the hardware and software components.

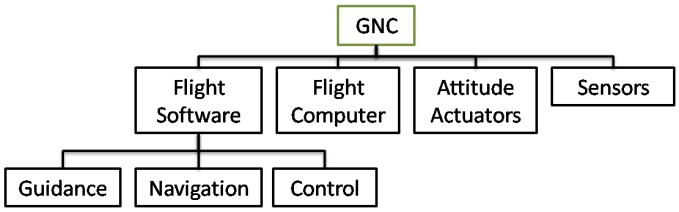

Figure 6.2: GNC subsystem hierarchy for the Paradox Mission.

To make the system hierarchy and WBS useful and visible to the students, Paradox has prominently displayed the hierarchy in the SDL. With a laminated finish and dry erase markers, the hierarchy is updated and student names are able to be added and removed as needed throughout the semester. Implementing the hierarchy in this way provides the project with four necessary attributes;

1. Provides students with a reference to the entire system.
2. Provides a sense of how an individual student and their work fits with respect to the entire system.
3. Provides a sense of responsibility between students for their deliverables.
4. Increases communication between students.

The hierarchy depicted in Fig. 6.1 is different from the photo in Fig. 6.3. The students actively modified and iterated on the hierarchy until it accurately reflected the project. Figure 6.3 showcases the active evolution of improving the system hierarchy over time.

Figure 6.3: Paradox implementation of the system hierarchy.

6.3 Interface Definition and Control

An interface is defined as a common boundary between two or more functions and applies at many different levels within a project.[1] Within a project, work is often partitioned because tasks can be done simultaneously or require a specific skill set. Interface definition outlines the way in which two functions interact and is often documented through an Interface Requirements Document (IRD). An IRD is a document that defines all physical, functional, and procedural interface requirements between two or more end items, elements or components of a system and ensures compatibility. As opposed to an Interface Design Document (IDD), which is a unilateral document controlled by the end-item provider, the IRD provides the details of the interface for a

design solution that is already established.[1]

The system hierarchy depicts the existence of an interface between subsystems and components as the connecting lines being integrated into a system one level above. Unfortunately, these lines are not explicit about subsystem or component interfaces. Interfaces are the glue for a system and interface definition facilitates the design process. For a student-based design laboratory, an IRD and IDD are not necessary, except with respect to external organizations. IRDs within a student project only add extra management work to ensure documents are consistently updated for basic interfaces between subsystems.

Instead, interfaces must be embedded within the design. Similar to an IRD, interfaces should be defined and managed by a single entity. Thus, the functions are consolidated within subsystems to minimize the number of interfaces and requirements. That is, grouping the interface responsibilities into a few subsystems is important in order to centralize the interface definition and design. For example, a structural subsystem will handle all physical and wire harness interfaces, while a command and data handling subsystem will define and design all data interfaces. Performing interface management in this way is possible due to the generally flat personnel hierarchy within a student-based design laboratory. Defining and managing interfaces within a student project in this manner will increase chances of success while preventing overhead documentation.

6.4 Example: GNC Interface Definition

The guidance, navigation and control (GNC) subsystem is a good example for interface definition and management as it interacts with most of the subsystems defined on Paradox. For the Paradox project only four of the six subsystems own interfaces: CDH, EPS, GNC and INT. The GNC subsystem is responsible for determining when, where and how to execute maneuvers for meeting mission objectives, hence the GNC subsystem is an excellent subsystem to manage interfaces and embed their design. Ownership indicates that the subsystem is responsible for understanding the interface between any two subsystems and appropriately developing requirements, designing and fabricating the interface as part of the overall subsystem design.

The GNC interfaces can be categorized into three different types: physical, electrical and data. Figure 6.4 depicts the interfaces to the GNC subsystem with respect to the rest of the satellite system. The interfaces are represented as an N^2 diagram. N^2 diagrams are often used to develop system interfaces; system components or functions are placed along the diagonal, outputs run along the horizontal axis and inputs to a subsystem are along the vertical axis. Figure 6.4 also depicts the interfaces between the hardware elements within the GNC subsystem. Note Figure 6.4 only depicts the interfaces with respect to the GNC subsystem for clarity. A full N^2 diagram of the satellite system would include other interfaces such as power being provided from EPS to COM. Table 6.2 then further describes each depicted interface in Fig. 6.4.

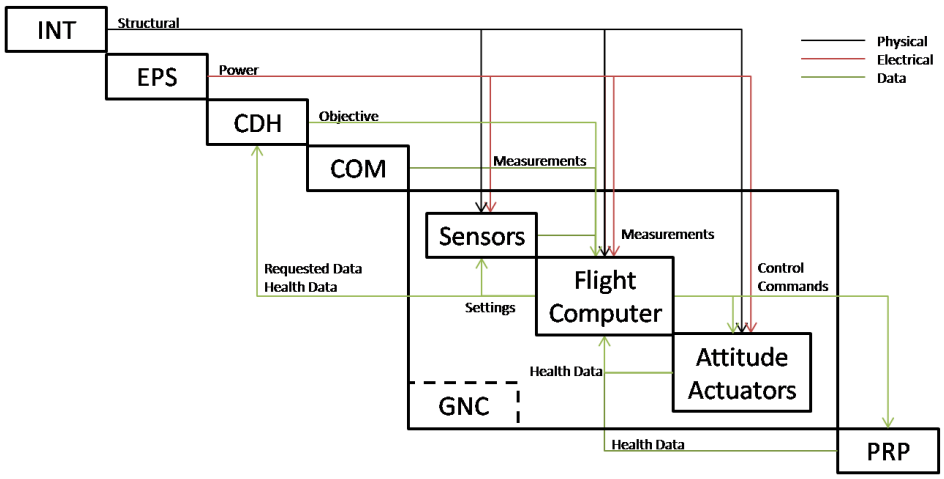

Figure 6.4: Basic interface definition for the GNC subsystem with respect to every other subsystem.

Table 6.2: Subsystem interface definition with respect to the GNC subsystem.

Subsystem	Ownership	Description
CAM	N/A	No direct interface.
CDH	CDH	A mission objective is provided for execution and pertinent mission data is returned.
COM	GNC	Receives relative navigation sensor data communicated from the target satellite.
EPS	EPS	Power is provided to the GNC hardware.
INT	INT	GNC hardware is mounted to the main satellite structure for support. Wire harness provides electrical connections.
PRP	GNC	Pressure and temperature data is received. Commands to open or close valves to perform orbit maneuvers.

The interface definition and management process is recursive. Within the GNC subsystem itself, interfaces are handled similar as at the subsystem level, where the flight computer owns the data interface between the sensors and attitude actuators. However, further technical design is needed to define detailed interface solutions. These initial interface definitions will become, for example, design drawings for mounting the flight computer circuit board to the main satellite structure and a messaging protocol for data transfer. Nevertheless, outlining the interfaces to this extent and communicating ownership to the student-based design laboratory will enable proper interface management and successful integration.

Chapter 7

Requirements

7.1 Functional and Performance Requirements

The requirements definition process transforms the stakeholder expectations into a definition of the problem and thus a design solution.[1] A major purpose for requirements is to clearly communicate mission expectations all the way down to manufacturing specific instructions such that the proper product is designed, fabricated and implemented. Developing requirements starts with defining functional requirements. "Functional requirements define what functions need to be done to accomplish the objectives."[1] Primarily derived from the mission needs, goals and objectives, functional requirements also come from the concept of operations (see Section 5.4). A complete set of functional requirements properly specify all intended uses or functions of the product over its lifetime.

Functional requirements describe "what", but performance requirements define "how well" the system must perform the functions. "Performance requirements define quantitatively how well the system needs to perform the functions."[1] Trade studies (discussed in Chapter 8) are the primary means to quantify performance requirements; however, they are not the only way. Both functional and performance requirements are written specifically to be independent of the design solution and only outline the engineering problem

that must be solved. The intent is to leave designers and engineers to develop an optimal solution based on technical data and analysis.

Distinctly different from requirements, constraints are imposed by an existing system, environment or stakeholder. Constraints often enter into the requirements development through the mission architecture description because of interfaces with an existing systems or the environment of operations. The system design is then required to satisfy the imposed constraints. The difference from requirements is that constraints are typically not able to be changed based on tradeoff analyses.[1] Thus, an engineer has no control over constraints except in how to design the system to meet them.

Initially, requirements are written to outline the problem and the general use of a product for designers to develop a design. Once a conceptual design is established, requirements are written to explicitly outline the engineering problem that shall satisfy the functional and performance requirements. Last, fabrication and integration requirements, commonly referred to as specifications, are written to detail how the design is to be manufactured and assembled.

The primary reason for each type of requirement in an aerospace organization is due to the exchange between different functions people perform. Requirements are crucial to maintain translation between these different groups. For student-based design laboratories, the consecutive levels of requirements are not necessary. A small student group is expected to perform at all levels of a project by working on design, fabrication, testing, integration and system testing. Ideally the student designing hardware is also fabricating and testing

the hardware. Thus, specification development after the design formulation is not necessary. Documentation of the actual design is more critical, due to the high turnover rate, than writing specifications of the design that meets the system functional and performance requirements.

Nevertheless, functional and performance requirements development is a key aspect to the systems engineering of a project within a student-based design laboratory because it is the direct link between customer expectations, technical analysis and the design solution. They essentially provide a communication platform for performing trade studies and functional analysis with the ultimate intent of producing the design solution.

7.2 Rationale

Often, a requirement may not appear to directly support the mission; thus, a rationale provides additional information to clarify the intent of a requirement.[1] Requirement rationales are used to document the purpose, assumptions, constraints and analysis built into the requirement as it is written. The purpose of a requirement can range from directly supporting a mission objective to defining the relationship between other systems and subsystems without implying a design solution. Naturally technology development, concept of operations and architecture, directly translate into requirements and must be stated within the rationale as the source for the requirement. In addition, requirements are derived from external constraints, such as from the operating environment or an existing system. These constraints must be stated in the rationale and referenced with supporting documentation or interface de-

sign documents. Last, analysis plays a large part in determining performance requirements. Often the rationale for the performance requirement will reference the analysis which supports the measure of performance cited in the requirement.

For a student-based design laboratory a requirement is worthless without a rationale. Student-based design laboratories have a high enough turnover rate for work to get lost as students move on and off a project from semester to semester. As new students join a project they bring with them different ideas and designs that meet the same mission need, such is the very nature of the highly innovative environment. The high turnover rate can cause a significant amount of rework because students have a tendency to personalize the design to what makes sense to them. If the previous work is not properly rationalized, then students will redo the design, analysis and test. Therefore, requirements must be supported with rationale because the originator of the assumptions, constraints and analysis that went into the requirement may have recently graduated.

7.3 Traceability

Traceability is the most difficult part of writing requirements, but it ensures that each requirement is necessary. To trace a requirement means to identify a set of requirements that map directly to one or more mission objectives. That way every derived requirement has either a direct or indirect purpose of supporting a mission objective. If project requirements are not properly traced, requirements can be orphaned without a clear reason to sup-

port the mission. Also, someone's favorite requirement, affectionately called "pet" requirements, can find themselves unnecessarily imposed on the design. Properly tracing requirements will prevent this requirements creep.

Requirements creep is the onset of additional requirements throughout the development life cycle. The design must constantly take into account added requirements regardless of the maturity. Requirements creep has been attributed directly to the extended development life cycle of FASTRAC.[7] To prevent requirements creep, a student-based design laboratory must actively engage all stakeholders to ensure expectations are being meet early in the life cycle. In addition, managing requirements and tracing the flow-down of requirements will prevent orphan requirements from creeping into a project.

Traceability of requirements can be even more difficult if professional software tools are not provided. A student-based design laboratory cannot afford a dedicated project requirements expert; all students must work on requirements. A proper numbering sequence, some level of formatting and direction provided by the student leadership should allow for the requirements to be easily traced from the mission need to the component level. The SDL uses Microsoft®Excel to capture all of the requirements information.

7.4 Example: Allocation of Functional Requirements from Mission Objectives

The mission need, goals and objectives for the Paradox mission are stated in Section 5.3. The objectives are restated as requirements in Table 7.1.

Table 7.1: Paradox Mission Objectives.

ID	**Mission Objective** "The mission shall..."
MO-1	Evaluate sensor suite performance.
MO-2	Evaluate actuator suite performance.
MO-3	Evaluate guidance, navigation and control capability in performing a rendezvous.
MO-4	Establish a communication crosslink between two satellites.
MO-5	Evaluate a high band-width communications groundlink.
MO-6	Evaluate imaging capability.

In addition to the six nominal mission objectives in Table 7.1 (per customer expectations), three extended mission objectives were determined based on the interests of the SDL and other stakeholders as described in Table 7.2.

Table 7.2: Paradox Extended Mission Objectives.

ID	**Extended Mission Objective** "The mission shall..."
ME-1	Adhere to the picosatellite form factor definition.
ME-2	Function autonomously or with limited input from ground controllers.
ME-3	Rendezvous with a separate physical satellite.

Table 7.3 presents the first twenty-one systems functional requirements for the Paradox satellite. A first and obvious functional requirement derived from all of the objectives is for the satellite to provide all electrical components with power. None of the mission objectives could be satisfied without providing power to the sensors, actuators or communications subsystems. Thus, the requirement is,

The Paradox satellite system shall provide power to electrical components.

Next, not only to satisfy a mission objective, but to also provide data for ground verification of the mission objectives, the satellite must provide ground communications. Thus the following functional requirement is derived from all six mission objectives,

The Paradox satellite system shall provide a communication capability to a ground station.

The process is similar for all of the functional requirements allocated for the satellite system. Following the structure of the Paradox system hierarchy in Section 6.2, Table 7.3 lists a subset of the entire set of functional requirements for the satellite system. Note that the requirements are directly traceable to either a mission objective (MO), an extended mission objective (ME) or to an operational requirement (OPS) derived from the development of the concept of operations as seen in Section 5.4.

Table 7.4 provides examples of rationales for five of the functional requirements in Table 7.3.

Table 7.3: Paradox Satellite (SAT) System Functional Requirements.

ID	**Requirement** "The Satellite system shall..."	**Parent Req.**
SAT-1	Provide power to electrical components.	MO-1,2,3,4,5,6
SAT-2	Provide communication capability to a ground station.	MO-1,2,3,4,5,6
SAT-3	Provide cross-link communications with a separate satellite.	MO-3,4
SAT-4	Perform a rendezvous.	MO-3
SAT-5	Determine its position.	MO-1,3
SAT-6	Determine its velocity.	MO-1,3
SAT-7	Determine its acceleration.	MO-1,3
SAT-8	Determine its attitude.	MO-1,3
SAT-9	Determine its angular velocity.	MO-1,3
SAT-10	Perform translational maneuvers.	MO-2,3
SAT-11	Perform rotational maneuvers.	MO-2,3
SAT-12	Process subsystem data.	MO-1,2,3,4,5,6
SAT-13	Execute commands.	MO-1,2,3,4,5,6
SAT-14	Provide video imaging capability.	MO-1,6
SAT-15	Provide a physical interface for subsystem hardware components.	MO-1,2,3,4,5,6
SAT-16	Provide structural support to subsystem hardware components.	MO-1,2,3,4,5,6
SAT-17	Adhere to all specified launch vehicle constraints.	MO-1,2,3,4,5,6
SAT-18	Adhere to CubeSat form factor.	ME-1
SAT-19	Perform mission objectives autonomously.	ME-2
SAT-20	Perform a rendezvous with a separate physical satellite.	ME-3
SAT-21	Provide health data as requested by ground mission control.	OPS-1

Table 7.4: Rationale for the Paradox Satellite System Functional Requirements.

ID	Rationale
SAT-1	Every subsystem utilizing electronics during the mission will need power.
SAT-2	To determine mission success, telemetry from the satellite must be communicated down to a ground station for post processing.
SAT-3	To satisfy the mission objective to evaluate communications with a separate satellite and to relay data for purposes of rendezvous.
SAT-4	To evaluate the vehicle's sensor, actuator and GNC capabilities, the vehicle will attempt a rendezvous and the system's performance will be evaluated.
SAT-5	To satisfy the mission objective to evaluate the sensor suite and GNC algorithms.

Functional requirements may also be allocated from mission objectives to other systems identified in the system hierarchy, such as ground support equipment. In addition, these satellite functional requirements are then allocated to the subsystem level. For example, Table 7.5 showcases the further derivation of functional requirements to the electrical power subsystem,

Table 7.5: Paradox Electrical Power Subsystem Functional Requirements.

ID	**Requirement** "The Electrical Power Subsystem (EPS) shall"	**Parent Req.**
EPS-1	Provide power to subsystems.	SAT-1
EPS-1.1	Provide power source.	EPS-1
EPS-1.2	Provide power storage.	EPS-1
EPS-1.3	Provide power to subsystems at specified voltages.	EPS-1
EPS-1.4	Provide power to subsystems at specified currents.	EPS-1
EPS-2	Provide electrical inhibits between the power source, power storage and electrical loads prior to separation from launch vehicle.	SAT-18
EPS-3	Provide health data as requested.	SAT-21

A similar process is carried out to derive functional requirements at any level within the system hierarchy. The next step is to quantify a system's ability to perform all of these allocated functions. For the example EPS subsystem above, the next questions to ask are: "How much and over what time is power required to be produced on-orbit?" or "How much power storage is needed to perform the full mission?".

7.5 Example: Derivation of Actuator Performance Requirements

Once functions are allocated to the various levels of the system hierarchy, performance requirements further define the system to be designed and built. Recall the following satellite system functional requirements from Table 7.3,

SAT-10 *Perform translational maneuvers.*

SAT-11 *Perform rotational maneuvers.*

The SAT-11 functional requirement is allocated to the guidance, navigation and control (GNC) subsystem, but there is no indication as to what kind of maneuvers and how fast the maneuvers must take place to satisfy mission objectives. Thus, performance requirements must be derived to specify quantitatively how well the rotational maneuvers shall take place. In the first writing of performance requirements, they are often augmented with the acronym To-Be-Resolved (TBR) primarily because the requirements have not been quantified at the time. That is, the technical analysis needs to be completed before populating the performance requirement with numbers. Table 7.6 and 7.7 showcases the first

Table 7.6: Paradox GNC Subsystem Requirements with TBRs.

GNC-1	Perform rotational maneuvers.	SAT-11
GNC-1.1	Perform rotational maneuvers at a minimum rate of TBR rad/s (TBR deg).	GNC-1
GNC-1.2	Reject environmental disturbance torques of at least TBR N-m.	GNC-1
GNC-1.3	Maintain an attitude with a maximum error of TBR rad (TBR deg).	GNC-1

The SAT-11 requirement is allocated to the Propulsion subsystem (PRP) and the performance requirements further define the satellite function.

Table 7.7: Paradox Propulsion Subsystem Requirements with TBR.

PRP-1	Perform translational maneuvers.	SAT-10
PRP-1.1	Have a total ΔV capacity of TBR m/s.	PRP-1
PRP-1.2	Maneuver with a minimum force of TBR N.	PRP-1

The derivation of performance requirements is one of the many intimate links between the systems engineering effort and the technical design effort. In order to quantify the performance requirements, trade studies and sensitivity analyses must be performed. The trade studies for these performance requirements are presented in the next chapter, Section 8.4. A level of iteration will be necessary to fully derive all requirements. Once the proper analyses have been performed, the requirements are populated with numbers. Therefore, the requirements from Table 7.6 and 7.7 are quantified as Table 7.8 and 7.9 show, respectively.

Table 7.8: Final Paradox GNC Subsystem Requirements.

GNC-1	Perform rotational maneuvers.	SAT-11
GNC-1.1	Perform rotational maneuvers at a minimum rate of 0.011 rad/s (0.6 deg/s).	GNC-1
GNC-1.2	Reject environmental disturbance torques of at least 2×10^{-5} N-m.	GNC-1
GNC-1.3	Maintain an attitude with a maximum error of 0.087 rad (5.0 deg).	GNC-1

Table 7.9: Final Paradox Propulsion Subsystem Requirements.

PRP-1	Perform translational maneuvers.	SAT-10
PRP-1.1	Have a total ΔV capacity of 15.0 m/s.	PRP-1
PRP-1.2	Maneuver with a minimum force of 1.0 N.	PRP-1

Each of the requirements in Tables 7.8 and 7.9 will include a rationale briefly explaining the analysis to support the performance specification as well as a reference to the documented analysis, if any. The SDL requires students not only to be responsible for specific subsystem design and fabrication, but to also be responsible for maintaining the subsystem requirements. The requirement maintenance includes performing technical analysis, writing rationales, ensuring traceability, and other subsequent requirement meta-data that is further covered in Chapter 11.

Chapter 8

Trade Studies

Trade studies are the technical analysis that support design decisions. Trades studies are the conduit from which design decisions are made and reflected in the systems engineering.[1] In fact, aside from designing, most of the effort for students will be performing trade studies to inform design solutions that meet mission requirements and constraints. Often trade studies and system design are concurrent activities. Both systems engineers and designers will perform trade studies. However, it is the job of the systems engineer or the student leadership to evaluate the impact to the overall system of a trade study result. The purpose of a trade study is to:

- Further bound the box in which to design. (Develop Requirements)
- Answer questions necessary for the design effort. (Explore Design Space)
- Evaluate design alternatives to decide the optimal for the system. (Compare Designs)

8.1 Trade Study Process

To perform a trade study there are eight basic steps[1]:

1. Identify the objective(s) of trade study result(s).
2. Review and state assumptions, constraints and requirements that impact the trade study.
3. Define evaluation criteria.
4. Identify design alternatives.
5. Evaluate design alternatives against established criteria.
6. Determine decision and record result(s).
7. Perform system sensitivity analyses.
8. Iterate.

To explain briefly each of these steps;

Objective(s). Each trade study requires time and a dedicated engineer; thus, the objective for the trade study must be clear and the expected result(s) must support the design effort. In addition, the student leadership must recognize the purpose and need for the trade study. Otherwise, the trade study is a waste of both personnel effort and time. The specific trade study objectives should be based on one of the three general purposes for a trade study, noted earlier.

Assumptions and Constraints. Rarely does a trade study result apply globally. For example, a choice to size a propellent tank at 0.1 m^2 is not fundamental to all aerospace vehicles, instead the trade study result makes sense only in the context of the assumptions and constraints of the specific mission. An assumption is made in the trade study to simplify the analysis, whereas a constraint is externally imposed on the analysis from either an aspect of the mission or another external system not being designed. Thus,

every trade study is accompanied with assumptions and constraints, and are often only reasonable when considering pertinent system requirements. It is very important to convey and document the assumptions, constraints and requirements.

Evaluation Criteria. Evaluation criteria is the key to correctly performing any trade study analysis. Criteria are the measures by which you evaluate the options. For the case when a trade study needs to answer a question or define a requirement, the criteria is rather the performance index to quantify. Criteria is often quantitative, but can also be qualitative. In addition, trade studies must take into account not just the engineering-based criteria, such as performance or mass, but also customer-based criteria, such as schedule or cost. Keep in mind not all criteria are created equal and ranking criteria will help in making a decision between two options. Last, an important point and easy mistake, when selecting the evaluation criteria choose criteria that does not favor one or a few of the options. For example, a trade study's objective is to select a power source for a crewed mission to Mars and one of the evaluation criteria is "radiation hazard". Given just three design options: nuclear, solar and batteries; it is clear that the criteria of "radiation hazard" will prejudice the trade study against a nuclear power source. Instead, the criteria should be "safety" where all safety concerns could be evaluated for all three options.

Design Alternatives. Identifying design alternatives is exactly where systems engineering and the technical design meet. Depending on the mission, finding design alternatives could mean performing internet searches for com-

mercial solutions, creating custom in-house designs or a combination. Regardless, it is important to recognize each design solution that meets the constraints and requirements for the mission before performing the trade study. On the other hand, it is equally important not to include alternatives that are out of the scope of the mission, do not satisfy the requirements or would not have worked.

Evaluate Alternatives. The technical evaluation is where most of the time is spent in a trade study. Often information is not available and tests and analysis must be performed, or direct contact with a manufacturer is the only way to glean the necessary information to make an objective comparison. Besides collecting data, the technical evaluation also includes an internal sensitivity analysis, which is different from the system sensitivity analysis. An internal sensitivity analysis provides an engineer with the relationships between the design alternatives and the evaluation criteria, thereby generalizing the trade study and simplifying future work if the study must be revisited.

Decision. Making an objective decision can often feel very subjective. Conventual wisdom and group-think can often drive trade study results even when the data suggests otherwise. Fortunately, there are decision making tools used by engineers to facilitate objective decision making such as Pro/Con lists, Analytical Hierarchy Process (AHP) and Pugh Matrices. Decision making can and should be done by student engineers on specific component level trade studies. Not every decision needs to make its way to a student manager or systems engineer. Nevertheless, the analysis should support the decision and the decision should be documented within a requirement or design, with the

analysis as support or rationale. Documenting the result(s) is crucial in a student-based design laboratory as the turnover rate is large and new students must understand past design decisions in order to maintain progress.

System Sensitivity Analysis. After one option has been selected during an iteration, the consequences of the choice must be evaluated against the entire system design. For student projects, the best sensitivity analysis to perform is to engage the team members with some form of the question, "How will this design decision, or trade study result affect your design?". In many cases there may be no effect, but assessing the implications on other aspects of the system design will provide new information to include in the next iteration of the trade study. Even if the new information is garnered through a student-to-student dialogue. More information is provide in Section 8.3.

Iterate. The last step, and most important, is to iterate on the trade study. Does the result make sense? How does the design decision affect the entire system design? Do you have new information that may affect the trade study result? The next iteration may not occur immediately. Often the trade study is revisited periodically in the life cycle. On the other hand, it is important not to slip into "analysis paralysis". Consider the project schedule before iterating on the trade study again.

8.2 Concurrent Design Effort

For a student-based design laboratory, trade studies are a unique systems engineering process that heavily involves the design development. For

a small organization with tight time constraints, design development and the evaluation of those alternative designs must become a single process for a small number of students. That is, students must be able to refine a couple of possible designs to a point at which the trade study can make the necessary comparison and decision.

8.2.1 Initial Downsize of Alternative Designs

Within a single trade study, an array of very different design solutions may be present, but for a proper evaluation each design must have a sufficient level of detail. Depending on the trade study, developing even more than three detailed designs at the component or subsystem level will drain resources; thus, downsizing from the large initial pool of concepts will enable a team to investigate a few designs, compare and ultimately select one.

To determine the subset of designs to refine, consider using qualitative comparison tools such as a Pro/Con list, or a basic quantitative tool such as AHP. While these are basic tools, they can be indispensable in an initial design down-select.

8.2.2 Parallel Design Development

Once an initial analysis indicates a few viable design options, the next step is to explore each alternative and quantify the performance towards meeting the mission objectives. The intent is to refine each design option enough to determine first, if it is still an option that meets mission requirements, and second, how well each alternative quantifiably meets mission requirements. Using the quantifiable data for the evaluation criteria, instead of the qualitative data

for the initial down-select, enables a decision to be made.

8.2.3 Decision Point

Designs could always be better and analysis could always be more accurate, but be aware of schedule constraints. It is very important to identify deadlines in which trade studies must have a decision, simply because the rest of the system design is depending on the result. Once the decision is made, more time can be spent on improving the chosen design alternative as part of the entire system. Returning to a trade study is always possible if the decision has unforeseen ramifications, but often other aspects of the design depend on the result. Solicit input from student leaders or faculty, but fear of making a decision should be avoided.

8.3 System Sensitivity Analysis

After all the work leading to a design choice, a sensitivity analysis could reverse the decision. Yet, sensitivity analyses are critical to ensure the overall design will function as a single system. A system sensitivity analysis should be performed at the subsystem level and focused on the implication of one subsystem design to the rest of the subsystems. The analysis should focus on interfaces and resources shared between the subsystems. Resource management is covered in more detail in Chapter 9. However, the time to perform a detailed sensitivity analysis in a student-based design environment is often unavailable. Student engineers working in subsystem teams often optimize a subsystem design without regard to other subsystems. Therefore, student-based design laboratories must perform a different, shortened sensitivity analysis to

prevent isolated designing.

Given trade study results or a chosen design, a basic sensitivity analysis against the other aspects of the system will determine how the decision affects the overall design. To perform the analysis, first qualitatively brief team members within each separate division of the project. Technical reviews, as discussed in Chapter 12, also provide a means to communicate trade study results. Depending on the trade study and the system being designed, the following questions should be asked;

1. Will the result(s)/design(s) change any system or subsystem requirements, fail to meet constraints or alter any assumptions previously made?
2. Will the result(s) define or change an interface between subsystems?
3. Will the result(s) affect the system resource margins (ie. mass, volume, power, etc.)?
4. How will the result(s) affect the subsystems?
5. Will the result(s) affect the cost of the subsystem?
6. Will the result(s) drive other subsystem designs?
7. Will the result(s) cause significant re-work in the subsystem design?

The initial qualitative analysis could be performed during a team meeting or one-on-one communications. This quick and simple communication will simultaneously inform the team of design decisions and contact those impacted by the decisions. Once impacted aspects of the system have been identified, a quantitative analysis can be performed, if needed. Then, once data has been collected on the impact, the result or design decision needs to be reevaluated taking into account the new information.

8.4 Example: Analysis of GNC Actuator Performance

Based on the Paradox satellite system functional requirements derived from the mission objectives, there is a need for both translational and rotational actuation. Actuator performance is a function of both the environment and the guidance software determining when and how to move the vehicle.

The example trade study will go through a series of technical analyses showing the derivation of the performance requirements imposed on the design of actuators and the design of the rendezvous guidance algorithm.

8.4.1 Translational Actuation

To perform the mission outlined by mission goals and objectives, regardless of the choice of the actuator, an analysis must be performed to understand how well the vehicle must make a series of translational maneuvers. The question to be answered is, "What is the maximum translational capability (ΔV) the system shall employ to successfully satisfy the mission?" Answering the question will directly derive a performance requirement that the design of the propulsion subsystem must satisfy to provide the necessary translational actuation.

8.4.1.1 Problem Definition

Objective. Given the mission objective of rendezvous, a system functional requirement dictates, " SAT-10: The Paradox satellite system shall perform translational maneuvers." The next step is to determine qualitatively to what extent the vehicle will perform translational maneuvers

to accomplish a rendezvous.

Assumptions and Constraints.

1. The three-unit CubeSat standard.[10]
2. Earth orbit altitudes [300 - 800 km].
3. Total $\Delta V < 20$ m/s.[8]
4. Relevant total rendezvous time $0.5 < t_{rendez} < 24$ hours.
5. Unknown launch vehicle.
6. Unknown launch vehicle separation conditions.

Evaluation Criteria. The desire is to minimize the total ΔV required to rendezvous given the assumptions and constraints.

Alternative Designs. For this trade study there are less obvious alternative designs, which is common for performance requirement definition. Instead, the design of the guidance algorithm is synonymous with specifying the performance requirement, and thus, is to be developed through the trade. The guidance algorithm will incorporate elements of analytical optimization and an iterative Lambert targeter to minimize the total ΔV required to rendezvous.

Analysis. Instead of comparing designs, the algorithm is based on design heritage from the past project, Texas 2-STEP. However, initial testing indicated that further design work was needed. Thus, the analysis consisted of developing a robust algorithm that could handle the unknowns posed in the assumptions and constraints for Paradox. The full analysis is detailed in the following sections.

8.4.1.2 Guidance for Small Satellite Rendezvous Applications

Silva[3] developed an autonomous guidance algorithm for nanosatellite rendezvous applications between two satellites. The algorithm incorporates a quick and robust analytical Clohessy-Wilshire (CW) optimizer to determine the drift and transfer times to optimize the two impulse rendezvous. Fig. 8.1 details the relative spacecraft dynamics of the rendezvous problem and the algorithm is described below.

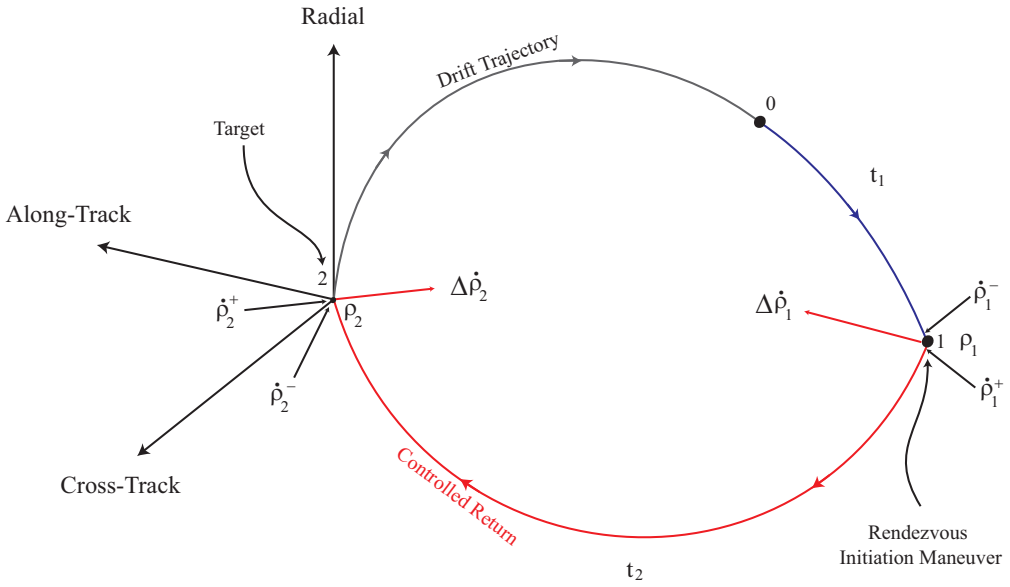

Figure 8.1: Description of drift and return trajectories in the relative reference frame.[3]

The CW optimizer computes the first impulse, $\Delta\dot{\boldsymbol{\rho}}_1$ and second impulse, $\Delta\dot{\boldsymbol{\rho}}_2$ based on a given drift time, t_1 and transfer time, t_2, using

$$\begin{bmatrix}\Delta\dot{\boldsymbol{\rho}}_1\\ \Delta\dot{\boldsymbol{\rho}}_2\end{bmatrix} = \begin{bmatrix}\mathbf{A}(t_2)\boldsymbol{\Phi}_{11}(t_1) - \boldsymbol{\Phi}_{21}(t_1) & \mathbf{A}(t_2)\boldsymbol{\Phi}_{12}(t_1) - \boldsymbol{\Phi}_{22}(t_1)\\ -\mathbf{B}(t_2)\boldsymbol{\Phi}_{11}(t_1) & -\mathbf{B}(t_2)\boldsymbol{\Phi}_{12}(t_1)\end{bmatrix}\begin{bmatrix}\boldsymbol{\rho}_0\\ \dot{\boldsymbol{\rho}}_0\end{bmatrix} \tag{8.1}$$

where

$$\kappa = 3nt\sin(nt) + 8\cos(nt) - 8,$$

$\boldsymbol{\rho}_0$ and $\dot{\boldsymbol{\rho}}_0$ are the initial relative position and velocity of the chaser vehicle, and

$$\begin{aligned}
\mathbf{A}(t) &= \frac{1}{\kappa}\begin{bmatrix} n\sin(nt) & 0 & -2n(7\cos(nt) + 3nt\sin(nt) - 7) \\ 0 & -\frac{n\kappa}{\tan(nt)} & 0 \\ 2n(\cos(nt) - 1) & 0 & n(4\sin(nt) - 3nt\cos(nt)) \end{bmatrix} \\
\mathbf{B}(t) &= \frac{1}{\kappa}\begin{bmatrix} n\sin(nt) & 0 & -2n(\cos(nt) - 1) \\ 0 & -\frac{n\kappa}{\sin(nt)} & 0 \\ -2n(\cos(nt) - 1) & 0 & n(4\sin(nt) - 3nt) \end{bmatrix}.
\end{aligned}$$

See Appendix A for the definition of the relative reference frame. Then, through a grid search, the algorithm selects the drift, t_1, and transfer times, t_2, that minimize the $\Delta V_{CW} = |\Delta\dot{\boldsymbol{\rho}}_1| + |\Delta\dot{\boldsymbol{\rho}}_2|$.

Given an optimal drift and transfer time, an iterative Lambert targeter more accurately determines the direction and magnitude of the $\Delta\mathbf{V}_L$ in the presence of environmental perturbations. Fig. 8.2 describes the orbital geometry.

A basis for on-orbit guidance, given an initial position and a desired final position within a specified time period, is posed as Lambert's Problem. Carl Friedrich Gauss first solved Lambert's problem and since the solution has seen many improvements. Richard Battin more recently presents a technique which will be the basis for the guidance law. For a full derivation and further instruction see Battin [2, 3] Gauss' solution begins with Kepler's basic time-transfer equation,

$$\frac{1}{2}\sqrt{\frac{\mu}{a^3}}(t_f - t_0) = E - e\sin E \tag{8.2}$$

Gauss' solution manipulated Kepler's time-transfer equation into a cubic equation in terms of the variable y and Battin's[2] solution augmented Gauss' form of the cubic equation to remove the singularity at half orbit periods for the transfer orbit.

$$y^3 - y^2 - h_1 y^2 - h_2 = 0, \tag{8.3}$$

The algorithm is described by beginning with an initial guess for x, as

$$x_0 = \ell.$$

The free parameter used to flatten Gauss' cubic form of Kepler's transfer time equation is defined as,

$$h_1 = \frac{(\ell + x)^2(1 + 3x + \xi)}{(1 + 2x + \ell)[4x + \xi(3 + x)]}\ , \tag{8.4}$$

$$h_2 = \frac{m(x - \ell + \xi)}{(1 + 2x + \ell)[4x + \xi(3 + x)]}\ . \tag{8.5}$$

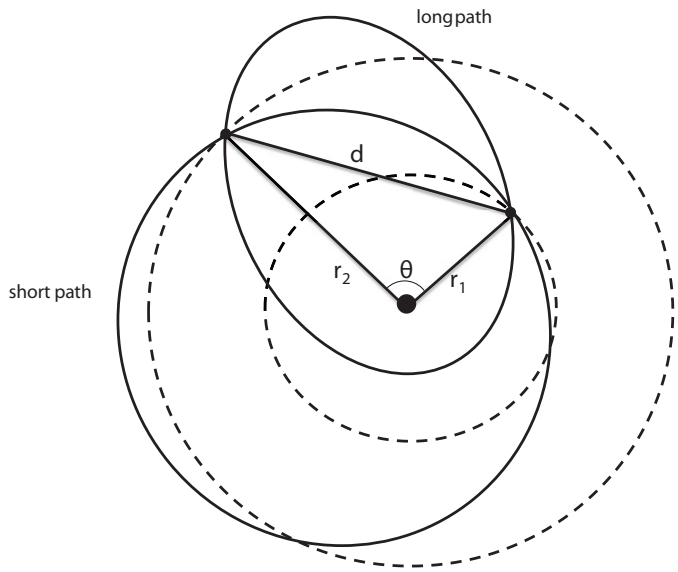

Figure 8.2: Lambert orbital geometry.

To simplify future equations, define the following variables,

$$B = \frac{27h_2}{4(1+h_1)^3}, \tag{8.6}$$

$$u = -\frac{B}{2(\sqrt{B+1}+1)}. \tag{8.7}$$

Solve Eqn.8.3 for y, yields

$$y = \left(\frac{1+h_1}{3}\right)\left(2+\frac{\sqrt{B+1}}{1-2uK^2(u)}\right), \tag{8.8}$$

and using the definition of y^2 to solve for x,

$$x = \sqrt{\left(\frac{1-\ell}{2}\right)^2+\frac{m}{y^2}}-\frac{1+\ell}{2}. \tag{8.9}$$

The algorithm is iterated until the change in x is below a desired tolerance. The desire velocity at the initial and final positions are calculated, via

$$\dot{\mathbf{r}}_1^+ = \frac{1}{\lambda(1+\lambda)}\sqrt{\frac{\mu(1+x)}{2s^3(\ell+x)}}\left[(\mathbf{r}_2-\mathbf{r}_1)+\frac{s(1+\lambda)^2(\ell+x)}{1+x}\frac{\mathbf{r}_1}{r_1}\right],$$

$$\dot{\mathbf{r}}_2^+ = \frac{1}{\lambda(1+\lambda)}\sqrt{\frac{\mu(1+x)}{2s^3(\ell+x)}}\left[(\mathbf{r}_2-\mathbf{r}_1)-\frac{s(1+\lambda)^2(\ell+x)}{1+x}\frac{\mathbf{r}_2}{r_2}\right],$$

and the two individual impulses are calculated as,

$$\Delta\dot{\mathbf{r}}_1 = \dot{\mathbf{r}}_1^+ - \dot{\mathbf{r}}_1^-,$$

$$\Delta\dot{\mathbf{r}}_2 = \dot{\mathbf{r}}_2^+ - \dot{\mathbf{r}}_2^-.$$

Thus, the total Lambert ΔV_L, given a transfer time, initial and final position is

$$\Delta V_L = |\Delta\dot{\mathbf{r}}_1| + |\Delta\dot{\mathbf{r}}_2|. \tag{8.10}$$

The full derivation and variable definitions are found in Silva[3] and Battin[2]. Upon initial inspection, the algorithm proved to be exactly what was needed for the Paradox guidance law, and could be used for the trade study to accurately determine the expected ΔV capability needed for the mission. However, simulations realized the limitations of the guidance algorithm which would subsequently limit the mission. Note the difference between the separate ΔV calculations $\Delta V_L - \Delta V_{CW}$, depicted in Fig. 8.3

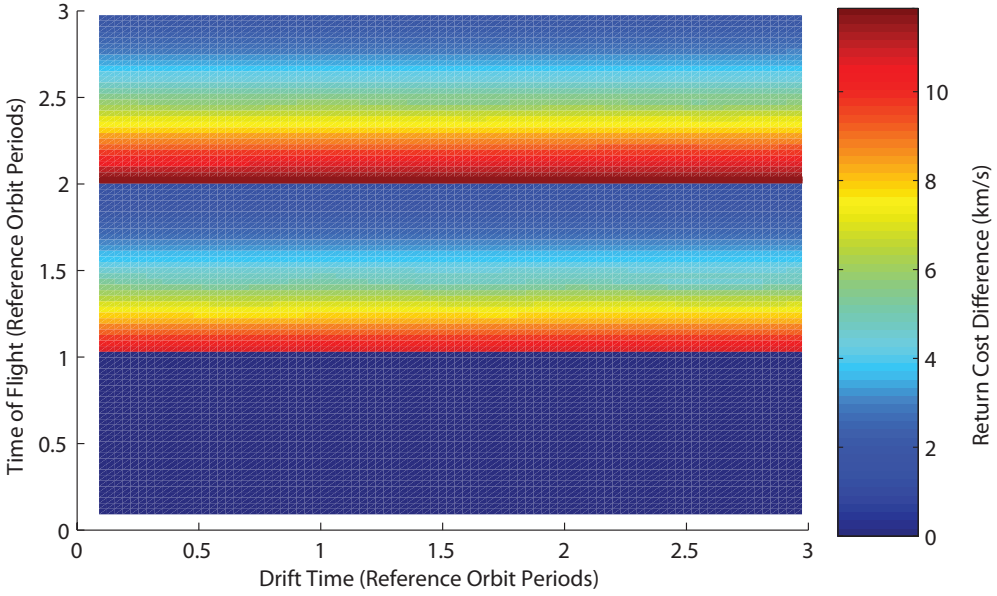

Figure 8.3: Difference between ΔV_L and ΔV_{CW} with respect to drift and transfer times.

The primary limitation is the ΔV_L for transfer times greater than one period of the reference or target orbit. The calculated ΔV_L is much greater than the analytical ΔV_{CW}. In fact, the actual ΔV_L for transfer times longer than a single orbit period are beyond the technological capability of picosatellites.[8]

8.4.1.3 Multi-Revolution Lambert Targeting

For transfer times greater than one orbit period, the difference between the analytical ΔV_{CW} and the Lambert ΔV_L diverge as seen in Fig. 8.3. Lambert solutions with a transfer time greater than one period do not take into account multiple revolutions transfers. Thus, the ΔV_L increases past the capa-

bilities of the mission. Therefore, the guidance algorithm suggested by Silva[3] must be modified to include multiple revolution transfers, subsequently lowering the required ΔV_L and increase flexibility during mission operations. Loechler[4] developed a Lambert targeting algorithm to include multiple revolution transfers. Equation 8.2 is augmented to include multiple revolution transfers,

$$\frac{1}{2}\sqrt{\frac{\mu}{a^3}}(t_f - t_0) = N\pi + E - e\sin E$$

For each multiple revolution, $N > 0$, two solution branches exist, an upper branch and a lower branch. For cases where $\theta \leq 180°$, the upper branch corresponds to large eccentricity transfer orbits, and the lower branch corresponds to small eccentricity transfer orbits. For cases where $\theta \geq 180°$ as defined in Fig. 8.2, the upper branch corresponds to small eccentricity transfer orbits, and the lower branch corresponds to large eccentricity transfer orbits.[5] Thus, two separate algorithms are required to determine the small eccentricity transfer orbit because the large eccentricity transfer orbit is not feasible for small satellites.

For the upper branch, the algorithm is initiated with a guess for x,

$$x_0 = 1 + 4\ell.$$

The rest of the algorithm (Eqns. 8.6 to 8.10) is implemented just as the regular Lambert targeting algorithm in Section 8.4.1.2 except the free parameter (Eqn. 8.4) used to flatten Gauss' cubic form of Kepler's transfer time equation.

Equation 8.4 now becomes,

$$\begin{aligned} h_1 &= \frac{\ell + x}{4x^2(1+2x+\ell)}\left(3(1+x^2)\frac{N\frac{\pi}{2} + \tan^{-1}\sqrt{x}}{\sqrt{x}} - (3+5x)\right), \\ h_2 &= \frac{m}{4x^2(1+2x+\ell)}\left[\frac{[x^2-(1+\ell)x-3\ell]\left(N\frac{\pi}{2} + \tan^{-1}\sqrt{x}\right)}{\sqrt{x}} + (3\ell + x)\right]. \end{aligned}$$

However, the lower branch algorithm is substantially different. The initial guess $x_0 \neq 0$, but must be sufficiently small. Thus, a value of $x_0 = 1\times10^{-4}$ is shown to work well. The equation for the free parameter is also modified as

$$\begin{aligned} h_1 &= \frac{(\ell + x)(1+2x+\ell)}{2(\ell - x^2)}, \\ h_2 &= \frac{m\sqrt{x}}{2(\ell - x^2)}\left[\frac{(\ell - x^2)\left(N\frac{\pi}{2} + \tan^{-1}\sqrt{x}\right)}{\sqrt{x}} - (\ell + x)\right]. \end{aligned}$$

The variable definition for B changes but u remains the same as in Eqn. 8.6,

$$\begin{aligned} B &= \frac{27h_2}{4[\sqrt{x}(1+h_1)]^3}, \\ u &= -\frac{B}{2(\sqrt{B+1}+1)}. \end{aligned}$$

The solution to the cubic equation is only slightly modified from Eqn. 8.8 and because of the change in the definition of y the solution to x is different as well.

$$\begin{aligned} y &= \left(\frac{\sqrt{x}(1+h_1)}{3}\right)\left(2 + \frac{\sqrt{B+1}}{1-2uK^2(u)}\right), \\ x &= \frac{1}{2}\left[\left(\frac{m}{y^2} - (1+\ell)\right) - \sqrt{\left(\frac{m}{y^2} - (1+\ell)\right)^2 - 4\ell}\right]. \end{aligned}$$

Again, the algorithm is iterated until the change in the solution to x is below a desired tolerance. The desired initial and final velocities can be calculated

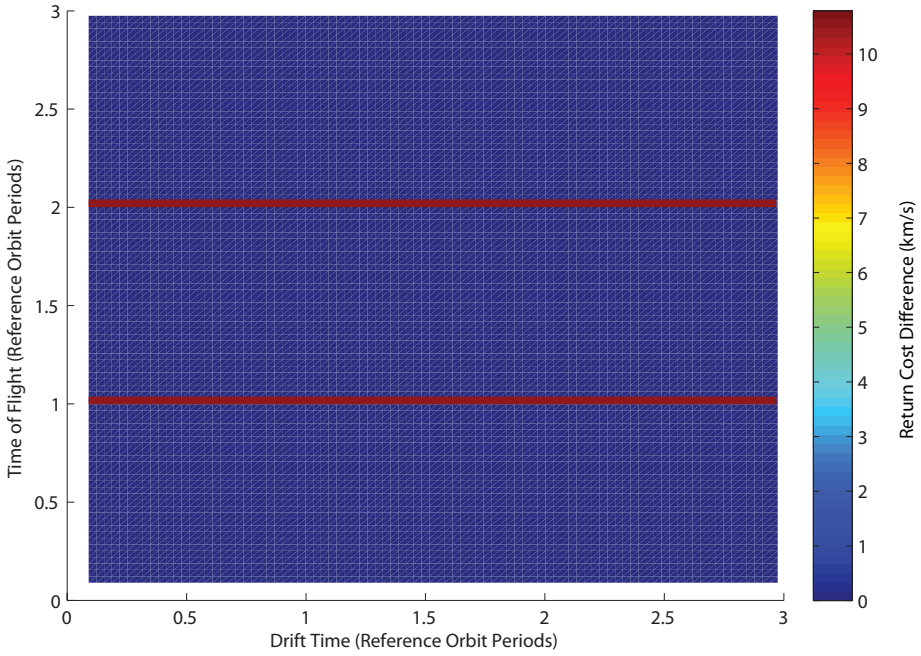

Figure 8.4: Difference between the ΔV_L and ΔV_{CW} with respect to drift and transfer times.

from Eqn. 8.10, and then the total ΔV_L from Eqn. 8.10. Replacing the regular Lambert targeter in the guidance algorithm with the multiple-revolution Lambert targeting algorithm significantly improved the calculated ΔV_L for transfer times greater than one period. Using the difference scheme $\Delta V_L - \Delta V_{CW}$, the two separate ΔV calculations are compared in Fig. 8.4. Notice that there is still a significant difference at integer values of the period of the reference orbit. In this case, the multiple-revolution Lambert algorithm still has the singularity seen in the regular Lambert algorithm derived by Battin.[2] Battin's method removes the singularity at $\theta = 180°$, which exists in Gauss' method, but one still remains at $\theta = 2\pi N$ or, for the problem considered in satellite rendezvous,

every orbit period.

8.4.1.4 Separation Conditions

A major design driver is the assumption that the launch vehicle and the separation profile are currently unknown. Thus, the vehicle must have the ΔV capability to handle any separation condition. Fig. 8.5 defines the separation dynamics between two spacecraft. Note that the second spacecraft has been omitted for clarity.

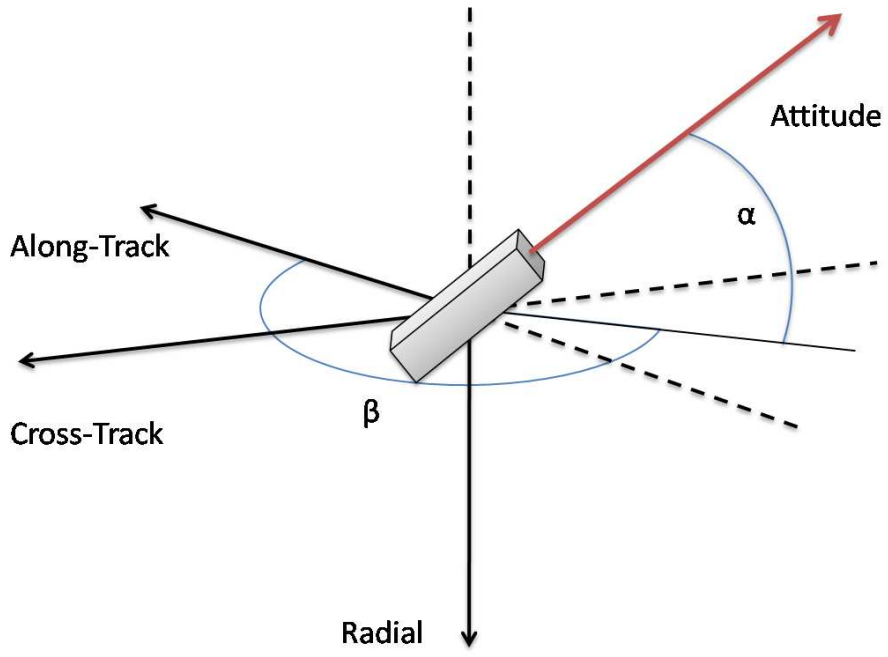

Figure 8.5: Separation dynamics between two spacecraft.

Silva[3] investigated the ΔV cost with respect to both the in-plane and out-of-plane separation angle. Fig. 8.6 showcases the relationship.

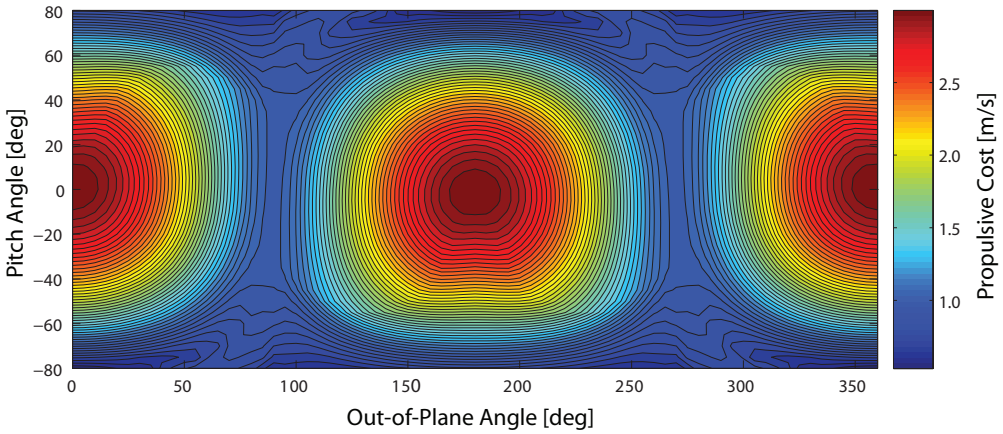

Figure 8.6: Optimal ΔV with respect to separation conditions between two objects.[3]

The analysis indicates that the highest ΔV and subsequently the worst separation case is at the in-plane angle of 0° and an out-of-plane angle of 0° or 180° for a fixed maximum total rendezvous time.

8.4.1.5 Result

The trade study results are two-pronged. First, the results directly derive a performance requirement for the parent functional requirement, "SAT-10: The vehicle shall perform translational maneuvers." Second, the trade study evaluated and refined the design of the guidance algorithm to provide optimal rendezvous trajectory regardless of separation conditions or total rendezvous time less than 24.0 hours.

Decision. The performance requirement is ultimately a conservative bound on the problem posed by the mission. Given the worst case separation conditions of α = 0° and β = 0° or 180° and total rendezvous time is

less than 24.0 hours, then there shall be enough propulsive capability to perform the rendezvous. The technical analysis gives a ΔV of 9.7 m/s. Adding approximately a 50% margin for mission assurance defines the performance requirement,

> PRP-1.1: The propulsion subsystem shall have a total ΔV capacity of 15.0 m/s.

The performance requirement is well within the feasibility of current technology.[8]

System Sensitivity Analysis. The result must be assessed against the current state of the design by performing a sensitivity analysis at each subsystem. Table 8.1 shows the impact of the results on each satellite subsystem.

Table 8.1: Satellite system sensitivity analysis of ΔV trade study result.

Subsystem	**Effect**
CAM	No effect.
CDH	No effect.
COM	No effect.
EPS	No effect.
GNC	The trade study essentially updated the guidance algorithm design.
INT	No effect.
PRP	The result effectively imposed a constraint on the propulsion subsystem design. Further propulsion design work will assess the feasibility of meeting the requirement.

Iterate. At the moment, the sensitivity results do not indicate the need for

an immediate iteration. The trade study directly affects the propulsion subsystem, and designers now have a lower bound to design. As the propulsion team converges on a feasible design, the ΔV trade study will need to be revisited if testing show that the requirement may not be satisfied.

8.4.2 Rotational Actuation

To perform the mission outlined by mission goals and objective, regardless of the choice of the actuator, an analysis must be performed to understand how well the vehicle must make a series of rotational maneuvers. The questions to be answered is, "What is the minimum rotational capability (rad/s) necessary to successfully satisfy the mission?" Answering the question will directly derive a performance requirement that the design of the attitude actuation subsystem must satisfy to provide the necessary rotational actuation.

8.4.2.1 Problem Definition

Objective. Given the mission objectives of rendezvous and to evaluate the actuator suite, system functional requirements dictate, "SAT-11: The satellite shall perform rotational maneuvers." The next step is to determine quantitatively what extent the vehicle will perform rotational maneuvers to accomplish a rendezvous and reject environmental disturbance torques.

Assumptions and Constraints.

1. The three-unit CubeSat standard.[10]

2. The center-of-mass is a distance less than 2 cm from the geometrical center.[10]
3. Earth orbit altitude [300 - 800 km].
4. Relevant total rendezvous time $0.5 < t_{rendez} < 24$ hours.

Evaluation Criteria. Determine the minimum rate (degrees/s) at which the satellite must rotate to accomplish a rendezvous. Determine the minimum torque (N-m) required to rendezvous or reject external torques given the assumptions and constraints.

Alternative Designs. For the trade study, again, there are less obvious alternative designs. Instead, the trade study will determine performance requirements.

Analysis The full analysis is detailed in the following sections.

8.4.2.2 Necessary Rotation Actuation for Rendezvous Sequence

The current design for the guidance algorithm implements two major impulsive maneuvers and a series of four intermediate course corrections to rendezvous. Thus, the satellite must have the capability to rotate to the desired attitude states before performing each impulsive maneuver. With a total rendezvous time, $0.5 < t_{rendez} < 24.0$ hours, the satellite may have to perform a rendezvous within 30 minutes. Thus, the satellite must have the capability to rotate the satellite to the desired impulsive vectors for each maneuver. If the last maneuver concludes the rendezvous, then the satellite has five minutes to rotate to the desired attitude in between each maneuver. Assuming a worst

case scenario where the desired attitude is 180° from the current attitude, then the vehicle must be able to rotate at 0.6°/s.

8.4.2.3 Environment Disturbance Torques

In addition to the rotational maneuvers between propulsive burns, the satellite must be able to hold a desired attitude before and during the impulsive maneuver. The three major environmental torques expected in low earth orbit are atmospheric drag, gravity gradient and residual magnetic dipole. The atmospheric drag force model is

$$\mathbf{f}_d = -\frac{1}{2} C_d A_p \rho v_{rel} \mathbf{v}_{rel}$$

where C_d is the satellites coefficient of drag, A_p is the profile area of the satellite with respect to the relative velocity vector, $\mathbf{v}_{rel}$, the atmospheric density is ρ and m is the mass of the satellite. The relative velocity vector is defined as,

$$\mathbf{v}_{rel} = \mathbf{v}_{sat} - \boldsymbol{\omega}_{Earth} \times \mathbf{r}_{sat}$$

The atmospheric torque on the satellite is the given by,

$$\mathbf{T}_{atmos} = \mathbf{r}_p \times \mathbf{f}_d$$

where $\mathbf{r}_p$ is the center of pressure with respect to the center of mass of the satellite and $\mathbf{f}_d$ is the atmospheric drag force acting through the center of pressure. The analysis chooses the satellite at a 300 km altitude orbit, since the relative velocity and density will bound the actual conditions expected from the assumptions. The maximum atmospheric drag torque is 1×10^{-17} N-m and essentially zero. However, if the center of mass is not at the geometric

center of the satellite, the atmospheric drag torque linearly increases. Thus, if the center of mass is at the maximum for picosatellite standards, ±2.0cm, then the maximum expected torque is 2.0×10^{-5} N-m. Figures 8.7 and 8.8 show the sensitivity of the atmospheric drag torque with respect to the satellite attitude and altitude, respectively.

Figure 8.7 depicts how the atmospheric torque changes with the attitude of the spacecraft. Since the drag force is a function of the projected area in the direction of the relative velocity vector, the drag torque is maximum when the attitude of the vehicle has a maximum projected area. As the vehicle is a rectangular prism, the maximum projected area is at an approximately attitude of out-of-plane angle of 45° and a pitch angle of 12°.

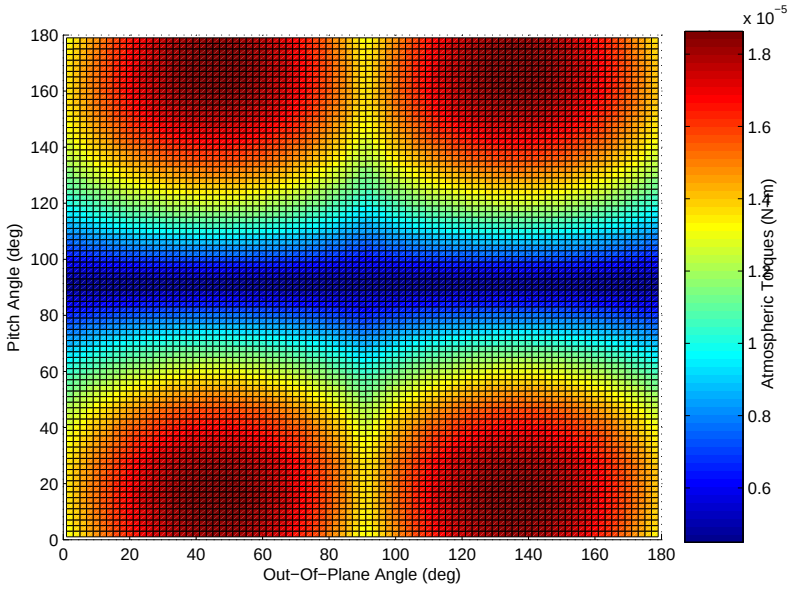

Figure 8.7: The atmospheric drag torque with respect to the attitude of the satellite.

Subsequence occurrences of the same projected area are at 135° out-of-plane and 168° as the vehicle is symmetric. The minimum atmospheric drag torque is at an attitude of 90° and 90°, which is when the smallest face of the prism is in the direction of the velocity vector. Figure 8.7 shows the sensitivity of the atmospheric torque over the assumed orbit altitude range. Thus, this internal sensitivity analysis on the trade study assumptions prevents rework if the trade study must be revisited once the orbit altitude is determined.

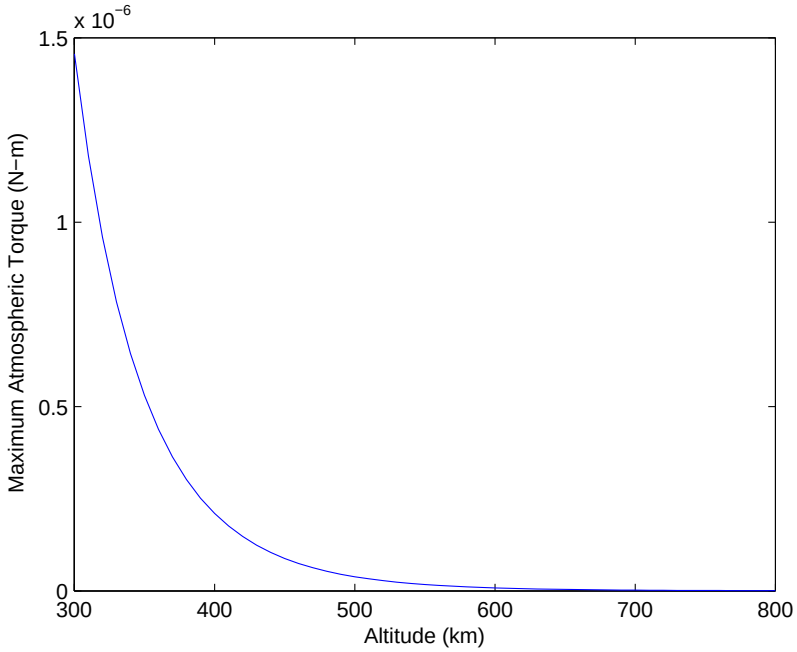

Figure 8.8: The atmospheric drag torque with respect to altitude given the worst case attitude.

The residual magnetic dipole torque is given by,

$$\mathbf{T}_{magnetic} = \mathbf{M}_r \times \mathbf{B}_{Earth}$$

where $\mathbf{M}_r$ is the residual magnetic dipole of the satellite, which is assumed to

be constant, and $\mathbf{B}_{Earth}$ is the magnetic field of the Earth. The typical residual magnetic dipole for a small-sized, uncompensated vehicle is 0.1 A-m^2.[9], in which case the worst case residual magnetic torque would be 4.0×10^{-6} N-m. Figure 8.9 shows the sensitivity of the torque to the expected altitude range.

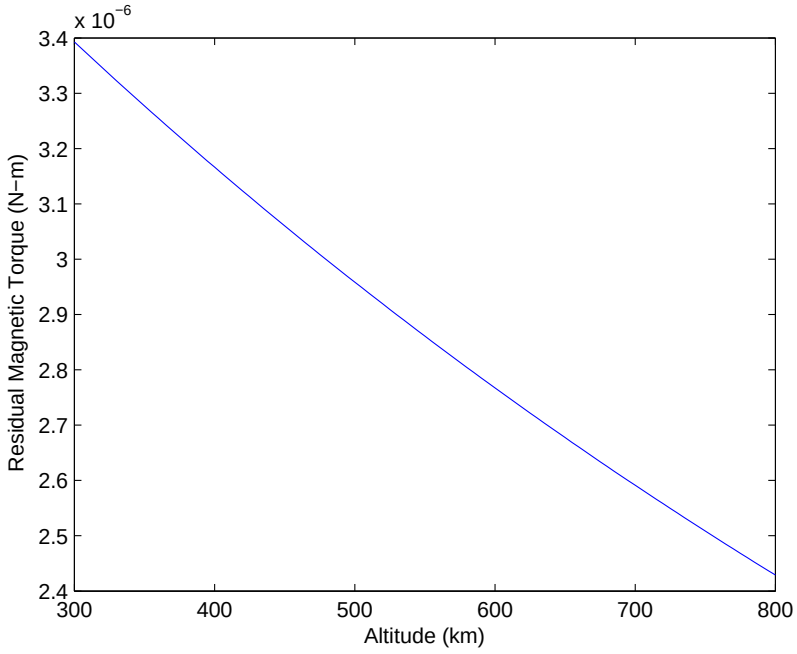

Figure 8.9: The residual magnetic dipole torque with respect to altitude given the worst case attitude.

Last, the effect of the gravity gradient torque, which can be represented as,

$$T_{gg} = 3n^2 \hat{\mathbf{r}}_e^{\times} \mathbf{I} \hat{\mathbf{r}}_e$$

where n is the satellite mean motion, $\mathbf{r}_e$ is the unit vector of the satellite's center to Earth's center expressed in the body frame, and $\mathbf{I}$ is the inertia matrix of the satellite.[11] Figure 8.10 and 8.11 show the sensitivity of the

gravity gradient torque with respect to the satellite's attitude and altitude, respectively. The expected worst case gravity gradient torque is 4.5×10^{-8} N-m.

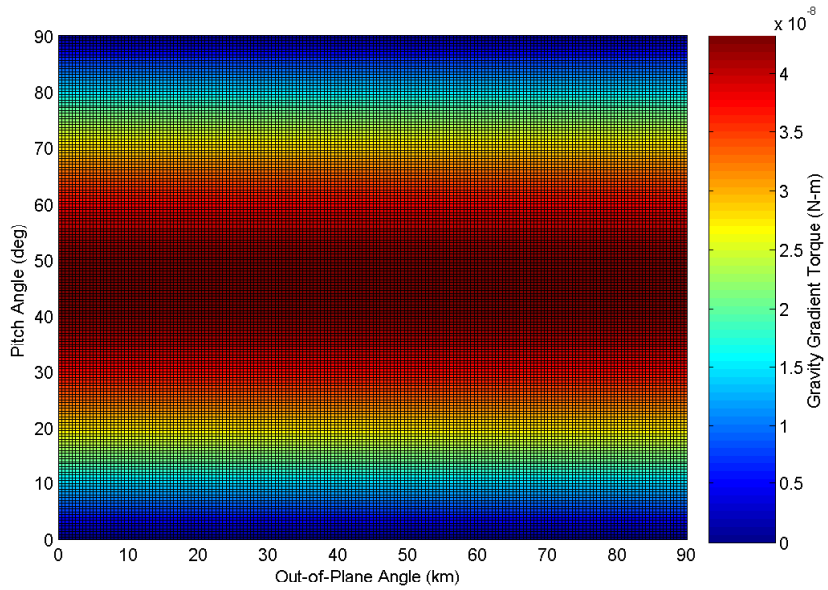

Figure 8.10: The gravity gradient torque with respect to the attitude of the satellite.

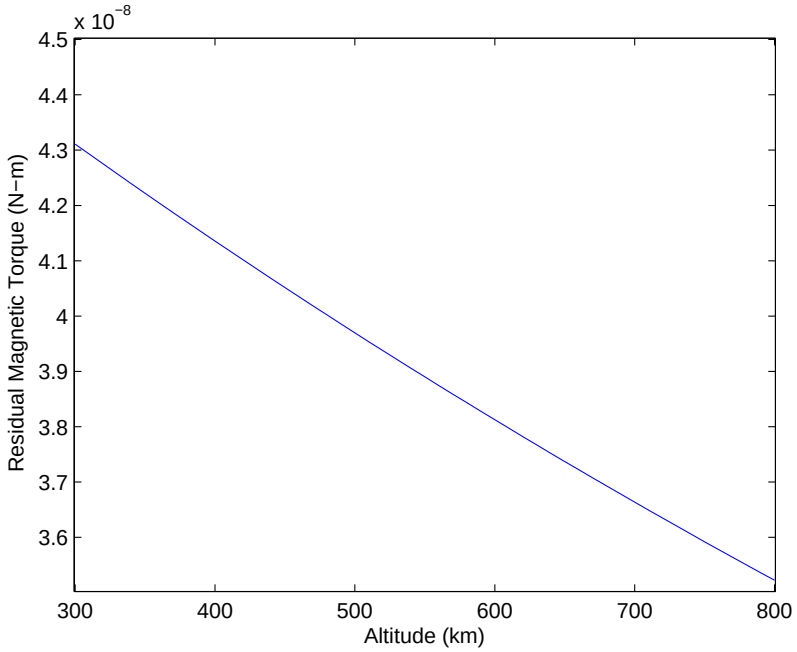

Figure 8.11: The gravity gradient torque with respect to altitude given the worst case attitude.

8.4.2.4 Result

The trade study results directly derive a two performance requirements to satisfy the parent functional requirement, "SAT-11: The vehicle shall perform rotational maneuvers."

Decision.

> GNC-1.1: The GNC subsystem shall perform rotational maneuvers at a minimum rate of 0.011 rad/s (0.6 deg/s).
>
> GNC-1.2: The GNC subsystem shall reject environmental disturbance torques of at least 2×10^{-5} N-m.

The performance requirements are ultimately a conservative bound on the problem posed by the mission. Given the worst case instantaneous torque of 2×10^{-5} N-m, which is the sum of the three environmental torques investigated, then the attitude actuator design shall the capability to perform the rendezvous at the minimum rate and overcome the disturbance torques required. However, the result must be assessed against the current state of the design by performing a sensitivity analysis as in Table 8.2.

System Sensitivity Analysis.

Table 8.2: System sensitivity analysis on the rotational actuation trade study result.

Subsystem	**Effect**
CAM	No effect.
CDH	No effect.
COM	No effect.
EPS	No effect.
GNC	Selection of the attitude actuators must be able to overcome environmental torques and rotate the satellite faster than 0.6 deg/s. In addition, momentum storage and rejection capabilities need to be evaluated.
INT	No effect.
PRP	No effect, unless used as a momentum rejection device.

Iterate. At the moment, the sensitivity results do not indicate the need for an immediate iteration. The trade study directly affects the GNC subsystem, and designers now have performance metrics to select the attitude actuators. As the GNC team converges on a design, the trade study will need to be revisited if the requirement is not successfully verified.

8.5 Example: Attitude Actuator Selection

Objective. Select or design attitude actuator hardware to satisfy the derived performance requirements,

> GNC-1.1: Perform rotational maneuvers at a minimum rate of 0.011 rad/s (0.6 deg/s).
>
> GNC-1.2: Reject environmental disturbance torques of at least 2×10^{-5} N-m.
>
> GNC-1.3: Maintain an attitude with a maximum error of 0.087 rad (5.0 deg).

Although not discussed explicitly in this thesis, a separate analysis derived the performance requirement for GNC-1.3 as the level of pointing accuracy required in the rotational maneuvers.

Assumptions and Constraints. The major constraints for selecting attitude actuators is in the extended objective for a three-unit CubeSat standard (SAT-18).[10] Any other assumptions and constraints on the actuator hardware are built into the performance requirements, such that if the performance requirements are satisfied the assumptions and constraints are as well.

Evaluation Criteria. Hardware specifications are defined either as individual hardware components or integrated as a complete attitude actuator hardware solution. The following criteria are ranked in order of importance:

1. *Volume*. The volume will be assessed based on the integrated design of the actuator hardware solution within the CubeSat form factor.
2. *Power*. Both nominal power consumption at the component level will be used as criteria.
3. *Performance*. The ability of the system to store momentum caused by disturbance torques, or reject disturbance torques real time is necessary. The primary metrics will be the nominal momentum storage, the solution's ability to unload momentum, and the nominal torque.
4. *Flight Heritage*. Whether hardware components have been flown successfully in space before.
5. *Mass*. The mass of the individual hardware components.
6. *Cost*. The total cost of the attitude actuator hardware solution.

Alternative Designs. For this trade study there are a variety of designs to consider. First, the general architecture must be decided. There currently are three different technology architectures to actively control a satellite's attitude, which can either be considered as independent solutions or as combinations.

1. *Reaction Wheels*. Momentum exchange devices, like reaction or momentum wheels, provide the level of accuracy required (GNC-1.3), but are not a viable independent solution due to a momentum saturation problem.
2. *Propulsion*. Propulsion systems do not have the momentum saturation problems as momentum exchange devices. However, active

attitude control will require large amounts of propellant and thus a larger volume than allowable within the CubeSat form factor. In addition, the accuracy of a CubeSat-sized propulsion system is not well known.

3. *Magnetorquer.* Similar to the propulsion solution, magnetorquer does not have a saturation problem. However, magnetorquers do not need an expendable (like propellant) to perform attitude control. However, due to the lack of knowledge of the Earth's magnetic field, the level of accuracy (GNC-1.3) will not make torque coils/rods a viable independent solution.
4. *Reaction Wheels and Propulsion.* Momentum exchange devices can provide the level of accuracy needed, while a propulsion system can provide relief to the saturation problem. Nevertheless, the propulsion system will require a significant volume and mass to be used to unload momentum from the reaction wheels.
5. *Reaction Wheels and Magnetorquer.* The coupled system provides the level of accuracy needed with the reaction wheels while alleviating the momentum saturation problem without adding significant mass and volume to the overall satellite design. Therefore, given the desired performance and constraints, a reaction wheel and magnetorquer combination should satisfy all of the requirements and constraints better than any other option.

Through this initial pro/con analysis based on the innate capabilities of each design architecture, the alternatives were down-selected

to a single architecture without spending time on detailed designs.

The reaction wheels and magnetorquer architecture design must be analyzed in detail. Due to the lack of in-house expertise and the level of complexity of developing reaction/momentum wheels, commercial-off-the-shelf options will only be considered. However, with an in-depth look at the commercial magnetorquer hardware, the students determined that a similar product could be reproduced to meet the requirements of the mission for a significantly lower cost. The current student design idea for the magnetorquer is similar to the Clyde Space 3U Solar Panel with Magnetorquer Printed Circuit Board. Nevertheless, the following commercial hardware solutions currently exist for the reaction wheels and meet requirements,

1. Sinclair Interplanetary: 30 mN-m-s Reaction Wheel (SI-30)
2. Sinclair Interplanetary: 10 mN-m-s Reaction Wheel (SI-10)
3. Astro-und Feinwerktechnik: Reaction Wheel Type A (AF-A)
4. Astro-und Feinwerktechnik: Reaction Wheel Type B (AF-B)
5. IntelliTech Microsystems, Inc.: IMI-100 Attitude Control and Determination System. (IMI-100)

The IMI-100 system is a stand-alone, off-the-shelf integrated reaction wheel and magnetorquer solution with accompanied sensors. The other commercial reaction wheels will have to be integrated with the student magnetorquer design to form a comparable solution to the IMI-100.

Analysis. First, review specifications on each product in Table 8.3 and 8.4

Table 8.3: Commercial Attitude Control Actuators Non-Performance Specifications.

Hardware	**Dimensions (mm)**	**Power (W)**	**Mass (g)**	**Cost ($)**
SI-30	50 × 50 × 40	1.5	185	~ 25,000
SI-10	50 × 50 × 30	0.7	120	~ 20,000
AF-A	⌀ 21 × 12	0.72	20	~ 18,000
AF-B	⌀ 21 × 12	0.72	12	~ 18,000
IMI-100	102 × 102 × 79	4.32	907	~ 70,000

Table 8.4: Reaction Wheel Performance Specifications.

Hardware	**Nominal Torque (mN-m)**	**Angular Momentum (mN-m-s)**	**Flight Heritage**
SI-30	2.0	30	Yes
SI-10	1.0	10	No
AF-A	0.023	1.1	Yes
AF-B	0.004	0.2	No
IMI-100	0.635	1.1	No

Based on these specifications, the Astro-und Feinwerktechnik reaction wheels are smaller and have significantly less mass for a similar power consumption level to the Sinclair wheels. In addition, both have flight heritage. The SI-30 are flight proven onboard the CanX-2 satellite, which launched April 2008, and the AF-A/B are successfully flying on BeeSat, launched in September 2009. However, the AF-A/B performance is significantly less than either of the Sinclair Interplanetary wheels.

Nevertheless, the key metric for the trade study is volume, or

rather how much usable space is consumed by a hardware solution. Usable space can be defined as the amount volume for which a hardware component can be fully integrated and assembled within the structure. Each set of reaction wheels was modeled and designed to mount directly to the structure with the intent to minimize the impact in volume while ensuring machinability and ease in assembly. For example, Fig. 8.12 shows an integrated solution using the SI-30 wheels. Similar design solutions were developed for each reaction wheel option and compared.

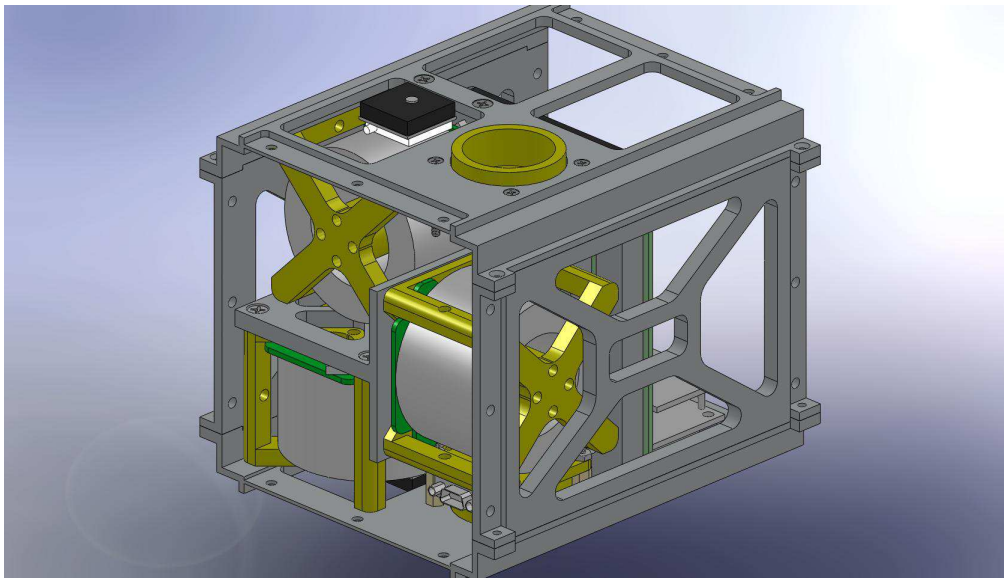

Figure 8.12: A custom configuration of three SI-30 mounted in a custom satellite structure.

To properly compare volumes with the IMI-100, the Sinclair and Astro-Fien reaction wheel solutions had to be modeled with the selected attitude sensors and accompanied electronics boards. Table 8.5 showcases the volumetric comparison of the innovative integrated attitude

actuator design solutions to compare with the IMI-100.

Table 8.5: Volumetric Comparison of Integrated Hardware Solutions

Hardware	**Volume (cm^3)**
SI-30	655
SI-10	507
AF-A	105
AF-B	105
IMI-100	813

Decision. First, the IMI-100 requires significantly more volume, even after taking into account the integrated electronics and sensors, while providing a medium level of performance. In addition, the IMI solution does not have flight heritage.

Between the two Sinclair Interplanetary wheels, there is enough savings in volume, power and mass without sacrificing performance to choose the SI-10 wheels over the SI-30 wheels. Similarly, the major difference between the two Astro-und Feinwerktechnik reaction wheels is in performance, thus the AF-A, with better performance, is the better candidate.

The resulting trade is choosing between the SI-10 and the AF-A wheels. A classic trade between performance and technical resources. Preliminary analysis confirms that both wheels satisfy the performance requirements, but the SI-10 wheels have a higher momentum saturation limit than the AF-A. Therefore, in the presence of significant disturbing torques on-orbit, a solution with the AF-A wheels will have to desaturate

more frequently using the magnetorquers. In this case, the SI-10 reaction wheels provide a more robust solution to environmental perturbations.

Nevertheless, the current baseline decision is the AF-A wheels because of the significant savings in volume and mass.

System Sensitivity Analysis. The reaction wheel hardware selection must be analyzed with respect to the other subsystems in the satellite. Further design and analysis with respect to the full system may provide more information to the trade study. Table 8.6 shows the effects of the result on the other subsystems.

Table 8.6: System sensitivity analysis from the attitude actuator trade study.

Subsystem	**Effect**
CAM	No effect.
CDH	No effect.
COM	No effect.
EPS	Must define power interface to selected hardware. Further power budget analysis will need to confirm power consumption level adequate.
GNC	The trade study selected the optimal hardware components to meet mission requirements based on relevant data. Must define data interface to selected hardware.
INT	Must define physical interfaces, both mounting the selected hardware to the structure and the necessary wiring harness.
PRP	The propulsion subsystem does not have to provide active attitude actuation or momentum desaturation.

Iterate. The comparison between the AF-A and SI-10 is difficult because of the positive and negative attributes of each. Further analysis is needed

to determine the sensitivity of desaturation frequency with respect to the range of expected on-orbit disturbance torques. A questions that remains is whether the desaturation frequency using the AF-A wheels adversely affects on-orbit operations. The trade study will need to be revisited once the analysis has been performed.

Chapter 9

Resource Management

A program/project resource is a physical entity, such as mass, volume and money, which is often limited or constrained. Technical resources, then, pertain to the resources utilized by the technical design to achieve the mission, such as mass and volume. Thus, money or a project's financial budget and schedule is not considered a technical resource and will not be covered here. Nevertheless, for a system being designed with constrained technical resources, those resources must be properly allocated to the subsystems, elements and components that ultimately makeup the larger system for which the resource constraint is imposed.

Resource allocation first begins with engineering estimates, or current best estimates (CBE), for each respective sub-element of a system. From a current best estimate and given the point within the project life cycle, a future projection of the resource is made to take into account expected growth as the design matures. This metric is considered the maximum expected estimate of the resource and is the amount of the resource actually allocated to the sub-element. That is, a subsystem design team provides the CBE based on "everything that can be currently accounted for", but the team requests an allocation of the resource because the design is not expected to grow past this "maximum expected value".

Therefore, at any point in the project life cycle there is a maximum possible, maximum expected and current best estimate for every technical resource. In general, the current best estimate of a resource changes as the development team improves the design, but the allocated amount would not change unless aspects of the system design requires a re-allocation of the resource.

The difference between the current best estimate and the allocated resource value is considered *contingency*. Contingency is often held at the subsystem level and the amount of contingency is based on the design maturity and so subsequently the project life cycle. Many definitions of contingency exist. The SDL uses the following definition[17],

$$\text{Percent Contingency } (\%) = 100\frac{\text{Contingency}}{\text{Current Best Estimate}}.$$

On the other hand, *margin* is the difference between the design limit (maximum possible value) and the allocated resource value. Margin is different from contingency in that it accounts for unknown unknowns which occur unexpectedly during the design development. One definition for percentage of margin is[17],

$$\text{Percent Margin } (\%) = 100\frac{\text{Margin}}{\text{Allocated Resource Value}}.$$

Normally margin is held at the system level, but it could be allocated to subsystems if necessary. In a student-based design laboratory, margin is

actively managed by the student leadership. Student-based design laboratories have no guidelines to how the margin is rationed over the life cycle and the decisions on rationing should be justified on technical analysis and data. Figure 9.1 depicts the relationship between a resource design limit, margin and contingency.

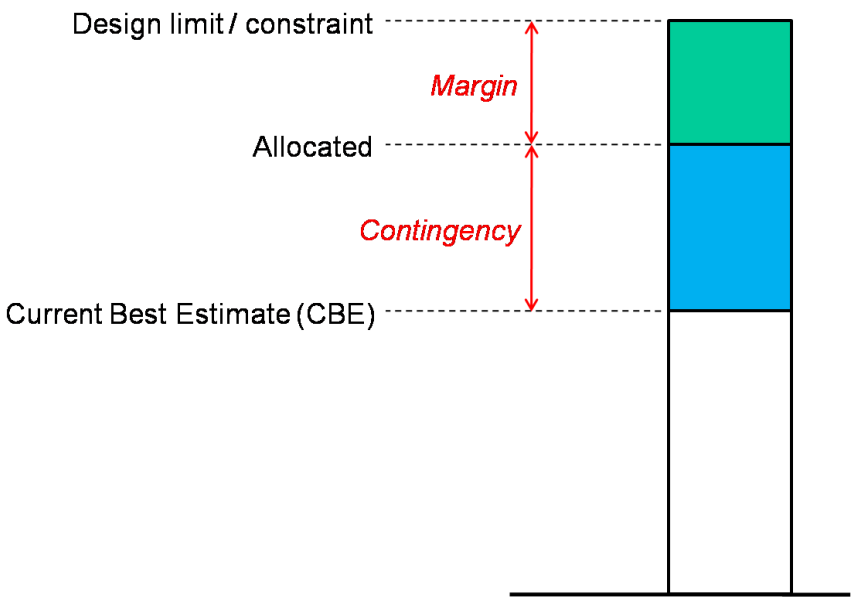

Figure 9.1: Margin and contingency with respect to any resource constraint.

NASA recommends levels of contingency and margin depending on the phase of the life cycle. Table 9.1 provides guidelines for margin.

Table 9.1: NASA Goddard Technical Resource Margin Recommendations based on Project Life Cycle.[17]

Resource	**Pre-A**	**A**	**B**	**C**	**D**
Mass	≥30%	≥25%	≥20%	≥15%	≥0%
Power (EOL)	≥30%	≥25%	≥15%	≥15%	≥10%
Propellant	≥30%	≥30%	≥20%	≥10%	≥10%
Pointing Accuracy	2×	2×	1.5×	1.5×	1.0×
Pointing Knowledge	2×	2×	1.5×	1.5×	1.0×
RF Link	6dB	6dB	6dB	4dB	4dB
Data Storage	≥40%	≥40%	≥40%	≥30%	≥30%
Data Throughput	≥30%	≥30%	≥20%	≥15%	≥15%

9.1 Example: Paradox Mass Budget

For all spacecraft, mass is an essential technical resource to monitor. With launch costs directly tied to mass, heavy constraints are imposed based on launch vehicles or separation devices. Other typical spacecraft resources include volume, power, average and peak data rate, propellant, and data storage. Mass can be considered a static resource that is demonstrated prior to launch. Other resources such as power can be very dynamic and require much more analysis and management to ensure a successful mission.

To demonstrate how the Paradox mission is managing mass as a technical resource, a series of tables show the allocation of mass through the Paradox satellite system. The design limit for the 3U CubeSat standard is 4.0 kg.[10] Table 9.2 demonstrates a component level mass budget where the current best estimate for the electrical power subsystem is determined.

Table 9.2: Electrical Power Subsystem Mass Budget

Component	**CBE** (g)
Power Board	86
Solar Panels	200
Battery Board	80
Battery Daughter Board	124
Total	490

Contingency is then applied to the individual subsystem based on the maturity of the design. The EPS subsystem is currently the most defined subsystem primarily because the subsystem is based on heritage hardware from PARADIGM; thus, a 5% contingency is applied. The propulsion subsystem is the least known and given a 20% contingency. As each subsystem design matures, the mass estimates are expected to erode into the original contingency allocated, as seen in Table 9.3.

Table 9.3: Paradox Mass Budget

Subsystem	**CBE** (g)	**Contingency** (%)	**Allocated** (g)
CAM	100	15	115
CDH	100	15	115
COM	200	15	230
EPS	490	5	515
GNC	950	10	1045
INT	500	10	550
PRP	500	20	600
Total (dry)	2840		3170

The allocated mass for the subsystems are then integrated into the allocated dry mass for the satellite. Adding the mass of the propellant based

on the required ΔV (PRP-1.1) and then the mass margin is calculated against the design limit for the entire satellite seen in Table 9.4. A margin of 19% for a system in Phase A of the project life cycle is considered low compared to Table 9.1. However, considering that much of the hardware has been selected and the masses are known, such as in the EPS, GNC and INT, the 19% is considered a healthy margin to act as a buffer for future unknown unknowns.

Table 9.4: Paradox Mass Budget Overview

	Mass (g)	
Satellite Dry Mass	3170	
Propellant	200	
Loaded Mass	3370	
Margin	630	19%
Design Limit	4000	

Chapter 10

Risk Management

Risk is a measure of the inability to achieve overall program objectives within constraints and has two components: (1) the probability of failing to achieve a particular outcome, and (2) the consequences of failing to achieve that outcome.[1]. There are many types of risks associated with any program, all which must be managed to some extent, even if all are not actively managed within a student-based design laboratory. Two general types of risk include technical and programmatic. Student-based design laboratories will naturally handle programmatic risks, such as cost and schedule risk, but should actively manage technical risk and personnel risk. Risk management involves analysis to identify and characterized risks and then the mitigation of those risks.

10.1 Risk Analysis

Technical risk analysis can take many forms, ranging from qualitative risk identification such as Failure Modes and Effects Analysis (FMEA), to highly quantitative methods such as Probability Risk Assessment (PRA). Risk analysis characterizes the risks in terms of two independent variables: probability of occurrence and consequence of the occurrence. Based on the probability and consequence, each risk to the project is then ranked to establish a plan of attack; focusing efforts and resources to mitigate risks to mission

success.

Often undergraduate students do not have enough background in probability and statistics to warrant the use of PRA. The amount of time a student would take to learn probability and statistics compared to the results that would directly impact the design suggests the time would be well spent on other tasks. Qualitative methods on the other hand provide a simple, yet effective way to identify risks and directly suggest mitigation strategies to implement into the design.

The premier qualitative method for risk analysis suggested for student-based design laboratories is Failure Modes Effects Analysis (FMEA), primarily because of the direct feedback from and into the design. FMEA is a methodology to identify failure modes, assess the risk for each mode, rank the causes of failure and identify mitigation strategies.[1] There must be a level of design prior to performing a FMEA. Nevertheless, FMEA is useful during the early design phases in order to affect the design. Sometimes mitigation strategies are solved in operational procedures, but most often it influences the system design. The tailored process for FMEA for a student-based design laboratory is outlined below[16],

1. **Function/Item**: First, identify a function or item, either software or hardware, to assess failures that may impact mission success.
2. **Failure Modes**: Identify potential failure modes for the function or item. A failure mode is defined as the manifestation of a failure. For example, loss of communications.

3. **Effects**: Identify the effects or consequences of the failure modes. A consequence is defined as the result of the failure mode. Thus, an effect from loss in communications may be mission failure.
4. **Severity Rating**: Rate the severity of each consequence. Qualitatively rate the severity with respect to mission success on a 1 - 5 scale where: 1 - Minor, 2 - Marginal, 3 - Moderate, 4 - Critical, 5 - Catastrophic.
5. **Causes**: Identify possible causes of the failure mode. The cause of a failure is defined as what induces the failure mode, such as the failure to deploy an antenna causes a loss in communications.
6. **Probability Rating**: Rate the probability of occurrence for each cause. Similarly, the probability rating is on a 1 - 5 scale, with the following qualitative assessment: 1 - Extremely unlikely (<20%), 2 - Unlikely (20% - 40%), 3 - Moderate (40% - 60%), 4 - Likely (60% - 80%), 5 - Very likely (>80%)
7. **Mitigation**: Determine mitigation strategies to decrease either the probability of the cause, consequence of the effect or both.
8. **Recommendation**: State the recommended mitigation strategy and assign responsibility.

Each function can have multiple failure modes and each failure mode may have a series of effects and causes. Each valid combination of an effect and cause represents a single risk. In addition, each cause or effect may be mitigated by a number of different strategies. A FMEA can very quickly become a "tree" of risks. Often, the FMEA process captures methods of detection that are already built into the design and suggest fault detection as

mitigation solutions. However, fault detection methods are really out of the scope for student-based design laboratories and not included in the tailored FMEA.

Once the risks have been assessed through a FMEA, the analysis needs to be easily communicated. Risk matrices are an easy way to manage and communicate risk. A standard risk matrix used by NASA is depicted in Fig. 10.1. The risk matrix is not an assessment tool, but it does communicate individual project risks and track the status of those risks.[1] The rating system described in the FMEA directly maps to the risk matrix as a communication tool.

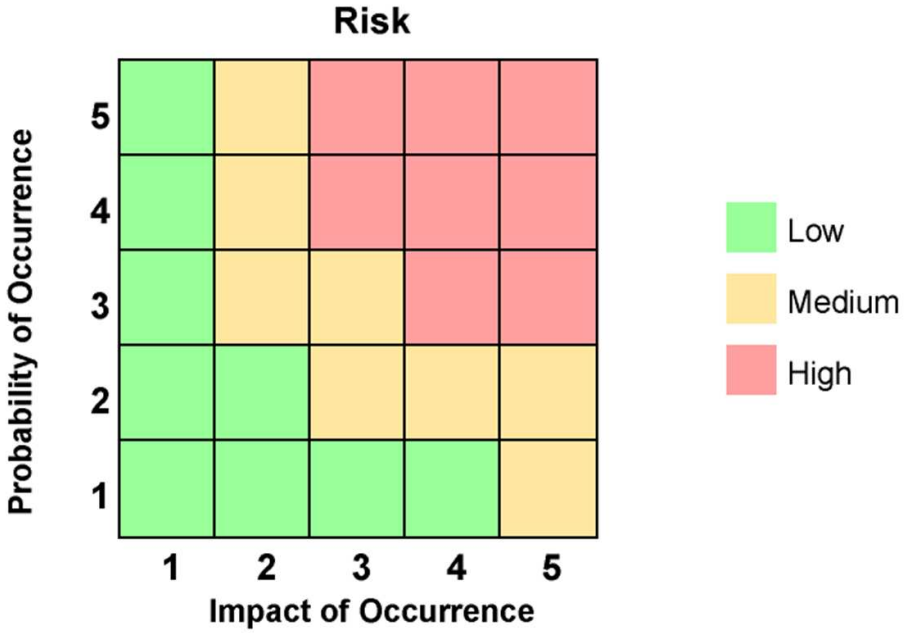

Figure 10.1: Risk matrix used to communicate project risks.[1]

Once the analysis is performed and the risks are populated in a risk matrix, the matrix identifies a level priority between separate risks. Thus, the

identified risks can be ranked and mitigated in a sequence that make sense to ensure mission success. An example is shown in Section 10.2.

10.2 Example: Failure Mode Effects Analysis on the Rendezvous Guidance Algorithm

The design of the rendezvous guidance algorithm is to the fidelity that a FMEA can be carried out to recognize and mitigate potential failures in the autonomous flight software on Paradox. Often a FMEA is represented as a chart stepping through the sequence outlined in the previous section. Only a single failure mode is considered in the example, but many possible causes and effects are represented.

In this case, many aspects of the preliminary design of Paradox are planned to be modified based on the FMEA in Table 10.1. The design of the flight software will be supplemented with autonomous verification software and with operational modes specifying impulsive maneuvers will not occur in an eclipse. In addition, a change to the concept of operations will provide "human-in-the-loop" checks verifying the autonomy of the system while ensuring mission success. To mitigate the severity of the risks would have required adding fuel reserves and would not have been possible due to volume constraints.

To communicate the assessment of the current risks from the FMEA, each risk is represented on a risk matrix shown in Fig.10.2. Briefly, AA (E-A and C-A) is the effect of partial loss of propellant caused by navigation errors. AB is partial loss of propellant caused by the Lambert singularity in

Table 10.1: FMEA on the Rendezvous Guidance Algorithm

Function/ Item	Rendezvous guidance algorithm
Failure Mode	The flight software executes an incorrect impulsive maneuver.
Effects	***E-A***: Partial loss of propellant. ***E-B***: Complete loss of propellant.
Severity Ratings	Effect ***E-A*** receives a rating of 3 because only a single objective, MO-3, may not be satisfied, but the other objectives would have already been accomplished considering the concept of operations (Section 5.5). Effect ***E-B*** receives a rating of 4 because only a single mission objective is not satisfied, but all other objectives would have already been accomplished.
Causes	***C-A***: Navigation errors produce incorrect estimates of the current satellite state. ***C-B***: The optimal time of flight is within the singularity of the Lambert targeter algorithm.
Probability Ratings	Cause ***C-A*** receives a likelihood of 2 because high errors in the state occur when the satellite is eclipsed by the Earth, which occurs approximately 30% of an orbit. Cause ***C-B*** receives a probability of 1 because simulations produce incorrect calculation due to the Lambert singularity <20%.
Mitigation	***M-A***: Implementation of verification software to autonomously check the validity of the impulsive maneuver prior to execution. (Decreases the probability of Cause C-A and C-B) ***M-B***: Modify the concept of operations to require all impulsive maneuvers to occur within sunlight. (Decreases the probability of Cause C-A)

Continued on the next page.

Mitigation	***M-C***: Modify the concept of operations to require ground passes and checks from ground controllers before satellite executes each impulsive maneuver. (Decreases the probability of the failure mode)
Recommendation	All three mitigation strategies should be implemented primarily because Paradox is in the early design phase and changes are easy to implement now.

the rendezvous guidance algorithm. BA is complete loss of propellant caused by navigation errors and BB is risk associated with complete propellant loss caused by the Lambert singularity in the rendezvous guidance algorithm.

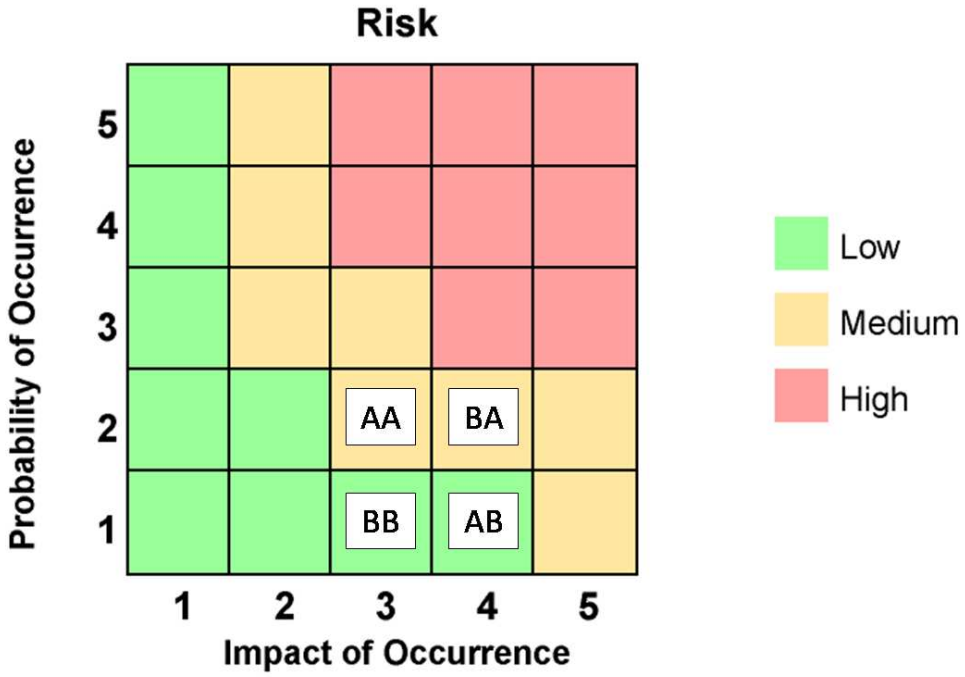

Figure 10.2: Risk matrix used to communicate risks from FMEA on rendezvous guidance algorithm.

The risk matrix allows for a ranking of the risk with respect to the order in which each should be mitigated: first BA, second AA, third AB and fourth BB. Thus, a focus on preventing complete or partial loss of propellant due to navigation errors should be first priority due to the medium risk indicated in Fig. 10.2.

10.3 Risk Mitigation

Mitigation of identified technical risks is the next logical step. Risk mitigation is defined as the process of reducing the severity or probability of a risk. Steps should be taken to eliminate or mitigate the risk if it is well understood and the benefits realized. Not all risks must be mitigated. For example, if the consequence of a risk is small and the mitigation is not justified in cost or schedule, then the risk may have to be accepted. On the other hand, if a risk is not well understood, that is, the probability and or the consequence is highly uncertain, then the risk must be further investigated until a level of certainty exists to properly mitigate.[1] Risk research includes testing, and if testing demonstrates that the probability or severity of a failure mode are negligible, then the risk is often considered mitigated through testing.

Mitigation for student-based design laboratories lies primarily in changing the design, either hardware or software, to affect the probability or consequence for any given risk. Thus, as designs are crafted, risk analysis identifies and characterizes the risk. Mitigation strategies then become integrated into the design. However, another primary way for student-based design laboratory to mitigate risk in a project is through heritage.

10.3.1 Heritage

Project or mission heritage is characterized by the use of hardware or software in a current design which has been utilized successfully in a previous mission. For a student-based design laboratory, heritage can greatly mitigate risk. To mitigate risk, a student-based design laboratory may choose to implement an entire subsystem that they have demonstrated on a previous, but similar, application. Conversely, a student-based design laboratory could implement a series of components with heritage from other applications as a way to mitigate the risk of the new integrated subsystem. In addition, heritage can come from previous student projects, industry missions or commercial-of-the-shelf components that have heritage on previous missions.

For example, flight heritage was used as a trade study evaluation criteria between the various commercial reaction wheels (see Section 8.5). Flight heritage is characterized as a product successfully flown in space, as opposed to ground-qualified heritage, which is qualified for flight but either has not flown yet or was not successful in the application. An example of ground-qualified heritage is from FASTRAC and PARADIGM. Heritage from either of the missions can only be considered ground-qualified because FASTRAC has not flown yet and PARADIGM could not successfully demonstrate most of its systems.

There are varying degrees to which heritage can be applied. If major modifications are required, the hardware or software can probably only be considered "experience", not heritage because the application is completely different. On the other hand, if only minor modifications are necessary, then

partial heritage can be claimed. Last, if a system is identical to a previous system it can be claimed as full heritage. Nevertheless, the following quote provides a reality check when implementing heritage into a mission,

> All use of heritage flight hardware shall be fully qualified and verified for use in its new application. This qualification shall take into consideration necessary design modifications, changes to expected environments, and differences in operational use.[17]

Thus, heritage can be used to mitigate risk on a project, not necessarily as a means to reduce the amount of work on a project.

In the end, student-based design laboratories are, by their very nature, high risk institutions due to the lack of experience and spirit of innovation. Risk analysis and mitigation must be part of a student-based design laboratory, but only insofar that the laboratory is not overwhelmed with activities not directly supporting the design and development effort.

Chapter 11

System Verification

Verification proves that a realized product for any system within a larger system structure conforms to the requirements. The objective is to generate evidence to confirm that the end products, from the lowest level of the system structure to the highest, conform to the specified requirements.[1]

While verification is the process of identifying that the system was produced correctly, validation determines if the system is the "right" system satisfying stakeholder expectations. Verification maps directly back to the requirements, whereas validation maps directly to customer expectations. Although two distinct processes, for a student-based design laboratory validation and verification should be performed simultaneously due to the considerable overlap between the two. Specific validation processes are not covered in this thesis.

There are four methods to verify a requirement: analysis, demonstration, inspection and test. A requirement can be verified by more than one method. Analysis uses mathematical models and analytical techniques to predict the capability of a design meeting the requirements. Demonstration is a qualitative assessment of a design meeting a specified requirement often functional requirements and system characteristics. Inspection is a visual examination of the end product. Testing is a technical process of gathering

data using special equipment and instrumentation to evaluated performance requirements.

A student-based design laboratory is expected to utilize each method based on the capabilities of the laboratory. Often within a student-based design laboratory state-of-the-art test equipment is unavailable; thus, students may rely on analysis, demonstration or less relevant test environments to prove aspects of a mission.

11.1 Configuration Management

To verify a design or product, both the design and what the design is being verified against must be documented. To document the design and the design verification, the documentation must be configured and available to the team. Thus, a key element to verification is configuration management and the documentation of the design and systems processes. Configuration management is a discipline applied over the product life cycle to provide visibility and control for documentation of a product.[1] Essentially, configuration management enables all stakeholders in the technical effort (ie students, faculty, external organizations), at any given time in the life of a product, to use identical data for development activities and decision making.[1]

The importance of configuration management increases with the number of personnel directly requesting or accessing information on a project. Thus, the larger a team, the more necessary is a well defined configuration management strategy. This is the critical reason why industry requires a significant configuration management plan for technical data.

While configuration management may not be a responsibility for a dedicated student engineer, it is still needed. The primary importance for configuration management is that documentation and technical data must be easily usable for all members of a product team. If configuration management is not done properly, then work is often lost and rework is performed causing delays. Additionally, if version control is not kept up-to-date, then hardware could be fabricated incorrectly, assumptions could be made based on old data or the system could be integrated incorrectly due to an old schematic. Therefore, configuration management is key to verifying the most current product with the most recent verification criteria.

Recently software products have been developed to assist in configuration management. One that has been extremely helpful is Subversion, which is available online. Subversion (SVN) has been shown to work exceptionally well for student laboratories as a version control and documentation repository. A documentation numbering structure must be developed and enforced by the student leadership. Subversion is used for all current projects within the SDL for flight software, simulations, hardware component specification documentation, systems processes documentation and CAD designs. The open source product handles most aspects of configuration management for the lab and is easy to use. To use SVN properly,

1. Update the repository prior to a work session.
2. Perform work related to the project during a work session.
3. Commit new files and changes to the repository before ending a work session.

4. Comment on new documents or modifications made during the work session to inform team members of pertinent additions and changes.

11.2 Documentation

Documentation within a student-based design laboratory is in constant conflict with students interests. The great debate as to whether time should be spent on design and development or on documentation is unending. External organizations, customers and faculty are always providing pressure on the student-based design laboratory to provide documentation.

Viewed as a heavily bureaucratic task, documentation is not something most students are eager to do after a level of design and development. Proper documentation of the design and a few key systems processes can suffice for an extensive amount of other documentation normally required by industry standards.

The crucial reason for documentation in a student-based design laboratory is due to the high turn-over rate. Rework is inevitable if some level of the design is not documented. In fact, students have a strong tendency to "make the design their own" even if a design from a previous student is documented, but not rationalized. The primary reason for that is because the constraints and assumptions built within a design are not always obvious to new students. Therefore, assumptions and constraints must be documented within design drawings or CAD, and in the form of comments for software. As mentioned in Section 6.3, documenting the design alleviates the need for separate specific documentation such as interfaces. However, specific types of

documentation may be required by an external organization. For example, Texas 2-STEP had to produce material lists, component lists, assembly procedures, test procedures, certificates of compliance and quality control on top of the following documentation required for a student-based design laboratory for the UNP-5 competition. It is important to keep in mind what is required by partners and customers.

In addition to documenting the design itself, certain systems processes need to be properly documented to ensure a level of continuity of the design and discipline in the project. Often a level of visibility of the systems documents within a student-based design laboratory serves as a constant reminder to think about the "big picture" and develops a systems engineering culture within the laboratory.

Specifically, the needs goals and objectives derived from customer expectations need to be documented and visible to the entire student-based design laboratory or project team. The concept of operations should be extensively documented in a similar form as presented in Fig. 5.3 and accessible to the team. Additionally, it is suggested to make the various pictorial concept of operations visually accessible within the lab as a quick reference during discussions. Similarly, the system hierarchy is an excellent tool to display in the laboratory as indicated in Fig 6.3. The systems hierarchy summarizes very quickly the entire system and helps define tasks to students. Trade studies, particularly ones that define the mission scope, must be well documented, while individual or small hardware selection trade studies do not necessarily need to be documented. All resource budgets, such as mass, volume, power,

telemetry, etc. should be recorded and maintained through the project life cycle. Lastly, any detailed risk analysis that significantly impacts the design should be documented to provide a rationale for the design change and mitigation strategies. A project risk matrix can be displayed to communicate project technical risks, similarly as the system hierarchy, in the laboratory.

The main student-based design laboratory systems process to document are requirements. A requirements document will contain an array of meta data stored in a single place within one document accessible to the project team. Some of the meta-data is detailed in Sections 7.4 and 7.5. The SDL uses Microsoft®Excel to capture all of the requirements information. When writing requirements, basic data to include are the identification number, rationale, parent requirement IDs, references and the student engineer responsible for the requirement. In addition, a requirements document can simultaneously include a verification matrix.

A requirements verification matrix is a tool to define how a requirement is verified with respect to the end-product and subsequently if the requirement has been verified or not. For a student-based design laboratory the verification matrix can easily be combined with a requirements document to reduce the number of documents that must be managed. Thus, within the requirements document the following meta-data should be included,

Identification (ID). A unique abbreviated form to quickly and easily reference specific requirements without stating the full requirement.

Requirement. The binding "shall" statement that is the requirement on the system.

Rationale. A brief explanation of the requirement including assumptions and constraints.

References. Documents that provide more supporting information that could not be covered in the brief rationale. Not every requirement shall have supporting references.

Parent Requirement. The ID(s) of the parent requirement(s) to provide traceability all the way back to the mission objectives. Every requirement has a parent requirement except for the mission need or goals.

Verification Method. The way in which the requirement will be verified either through Analysis, Demonstration, Inspection, Test or any combination of the four.

Success Criteria. The criteria for which the system will be evaluated against during the verification method(s). Often the success criteria is embedded in the requirement, especially if it is a performance requirement.

Verification Date. The date for which the requirement was successfully verified.

Responsibility. The student engineer(s) responsible for design, development and verification with respect to the requirement.

11.3 Example: Paradox Requirements Verification Matrix

As mentioned, for a student-based design laboratory the requirements verification matrix supplements the requirements; thus, only a single docu-

ment must be managed. The level of maturity in the Paradox project has not allowed any of the requirements to actually be verified to-date. However, the verification matrix is developed at the same time as the requirements themselves. For clarity of this example, much of the meta-data not directly related to the verification matrix is omitted here. The verification methods are abbreviated in the following manner: Analysis (A), Demonstration (D), Inspection (I) and Test (T). For a subset of the Paradox satellite system requirements, Table 11.1 shows the verification matrix meta-data. Refer to the functional system requirements in Table 7.3 for the actual requirement statements.

Table 11.1: A subset of the Paradox satellite system requirements' verification matrix.

ID	Method	Success Criteria
SAT-1	A	Given a final hardware design with expected electrical loads, analysis is successful if the system can stay power positive during all mission operations.
	T	System-level ground tests shall be successful if the system can provide power throughout expected mission operations.
SAT-2	A	If the link budget analysis proves the system has the recommended link margin (Table 9.1), the verification is successful.
	T	High altitude balloon tests shall be successful if expected mission data is transmitted to the UT ground station.
SAT-3	A	If the link budget analysis proves the system has the recommended link margin between the two satellites on-orbit (Table 9.1), the verification is successful.
	T	High altitude balloon tests shall be successful if expected mission data is transmitted between the two satellites.

Continued on the next page.

SAT-4	A	Simulations with flight software are successful if the satellite system rendezvous with a target within 200 meters under expected orbit conditions.
SAT-5	A	Simulations shall demonstrate the satellite's capability to determine its relative position on the order of tens of meters to be successful.
	T	Ground testing within a GPS simulator of sensors used to determine position shall the above accuracies to be successful.
SAT-8	A	Simulations shall demonstrate the satellite's capability to determine its attitude on the order of degrees for success.
	T	Ground testing of sensors used to determine attitude on an air-bearing platform shall show the above accuracies.
SAT-10	A	Simulations prove the system can perform the ΔV translational maneuvers required for rendezvous in expected on-orbit conditions to be successful.
	D	Demonstrate the system's ability to translate on an air-bearing platform.
SAT-11	A	Simulations shall prove the system can perform rotational maneuvers required for rendezvous in expected on-orbit conditions for success.
	D	Demonstrate the system's ability to execute rotations on an air-bearing platform.
SAT-12	D	Embedded hardware testing with the flight software shall successfully show the handling of mission data.
SAT-13	D	Embedded hardware testing with the flight software shall successfully show the execution of commands.

11.4 Testing

Of the four methods to verify requirements, testing is by far the most crucial. System verification testing relates back to the approved requirements set and can be performed at all stages of the project life cycle and at all levels of

the design.[1] Tests are often the only way to verify performance requirements. With unlimited time and money, each element would be extensively tested at every level of the system. Unfortunately, such an opportunity is unlikely in any industry, much less in a student-based design laboratory.

Even though the project life cycle is more sequential going from Phase A to F, within each phase there is an extensive amount of iteration. In life cycle phases A and B, there is an iteration on the design including some level of prototyping, but in phase C, a student-based design laboratory must make it a focus of their effort to do extensive prototyping and component or subsystem level testing to verify subsystem level requirements. If the requirements are unable to be verified, the requirement may be relaxed, changed or removed, if possible. Consequences of requirement modification later in the life cycle must be considered. Note that the level of iteration in Phase C can still be considered part of the design, as designs are being tested to meet the requirements before a final design configuration is chosen.

In phase D, the product is integrated together and system-level testing, often termed "end-to-end" testing, ensures verification of system-level requirements and customer expectations. This means assembling the system in its realistic configuration, subjecting it to a flight-like environment and then operating it through all of the expected operational modes.[1] The subtlety, however, is whether the test was performed in a "realistic environment".

A recent example in the experience of the SDL was the deployment and separation of PARADIGM. Although the most tested aspect of the system, PARADIGM failed to properly separate from the Texas A&M satellite

on-orbit. The deployment was tested on a two-dimensional air-bearing table multiple times during development. According to the failure report, it is likely that the deployment tests on the ground did not accurately take into account the environmental conditions and their effect on the separation of the two satellites. Although the ability to do a separation test in a space-like environment was unlikely, simply adding thermal vacuum tests may have independently uncovered the problem(s).

As with any flight project, more testing is encouraged to ensure mission success. For a student-based design laboratory, it is critical to stress that the work does not end with a paper design and that time must be allocated to test hardware at the component, subsystem and system level before delivery. Planning tests early is especially necessary if the tests require equipment and facilities from external organizations, which is likely for system-level tests such as thermal vacuum and vibration tests. In the current schedule for Paradox in Section 12.4, approximately a year is currently allocated for system integration and testing.

Chapter 12

Technical Reviews

12.1 Waterfall and Spiral Project Life Cycle

The NASA style project life cycle is classically the "waterfall" or linear organization of phases to decompose the process into more manageable pieces, recall Section 1.2. The rationale for a sequential approach to the development of a project is because, nominally, a concept comes before design, design comes before fabrication and fabrication comes before operation. The transitions between each of these distinguished phases are decision points or control gates. Thus, a project naturally steps through this linear sequence of events interrupted by reviews of work to ensure proper progression into the next major phase. However, it is within these specific phases, that a non-linear life cycle approach is needed for student-based design laboratories.

The spiral project life cycle is an alternative approach towards product development. The primary difference between the spiral and the waterfall is that the spiral approach cycles through design, development, build and test for successive refinement. The spiral project life cycle is most often utilized in the software industry where construction and preliminary testing are not as expensive as hardware fabrication and testing. However, within a student-based design laboratory a level of spiral development even in hardware is extremely useful within Phases B and C of the linear approach. Not only

does it serve to motivate the final design, but the cyclical nature of designing, prototyping and testing serves to constantly verify the system. In addition, in a student-based design laboratory with education as a primary objective, constant prototyping and testing serves to teach students what works and what does not work. Thus, a spiral method embedded within the waterfall approach of the life cycle is justified despite higher costs and more time, because students are more likely to produce design solutions that meets mission objectives.

12.2 NASA Minimum Set of Technical Reviews

As mentioned in the context of the life cycle above, the phases of the waterfall approach to a project life cycle are divided by decision points or control gates. Often referred to as technical reviews, these control gates are the events at which the decision authority (student leadership, faculty or customer) determines the readiness of a program/project to progress to the next phase of the life cycle.[1] Primarily, technical reviews are key development milestones used to measure progress and assess project maturity. If a project does not showcase enough progress or maturity, then the decision may be to remain in the current phase and go through the review at a later date or the decision may be to cancel the project entirely. It depends on demonstrating progress.

NASA outlines a minimum set of technical reviews every project must conduct. For a complete description of the entrance and success criteria please see: NPR 7123.1 NASA Systems Engineering Processes and Requirements.[18] In order the reviews are,

Mission Concept Review (MCR). End of Pre-Phase A. The MCR will

affirm the mission need and examine the proposed mission's objectives and the concept for meeting those objectives.

System Requirements Review (SRR). End of Phase A. The SRR examines the functional and performance requirements defined for the system and ensures that the requirements and the selected concept will satisfy the mission.

Preliminary Design Review (PDR). End of Phase B. The PDR demonstrates that the preliminary design meets all system requirements with acceptable risk and within the cost and schedule constraints and establishes the basis for proceeding with detailed design.

Critical Design Review (CDR). Mid-Phase C. The purpose of the CDR is to demonstrate that the maturity of the design is appropriate to support proceeding with full scale fabrication, assembly, integration, and test, and that the technical effort is on track to complete the flight and ground system development and mission operations in order to meet mission performance requirements within the identified cost and schedule constraints.

System Integration Review (SIR). Beginning of Phase D. A SIR recognizes whether a system's sub-elements, personnel, procedures and facilities are ready for integration.

Test Readiness Review (TRR). Throughout Phase D. A TRR ensures that the test article (hardware/software), test facility, support personnel, and

test procedures are ready for testing and data acquisition, reduction, and control.

Flight Readiness Review (FRR). End of Phase D. The FRR examines tests, demonstrations, analyses, and audits that determine the system's readiness for a safe and successful flight/launch and for subsequent flight operations. It also ensures that all flight and ground hardware, software, personnel, and procedures are operationally ready.

Operational Readiness Review (ORR). End of Phase D. The ORR examines the actual system characteristics and the procedures used in the system or product's operation and ensures that all system and support (flight and ground) hardware, software, personnel, procedures, and user documentation accurately reflects the deployed state of the system.

Post-Launch Assessment Review (PLAR). Beginning of Phase E. After launch and deployment, a PLAR assesses the systems status and confirms the decision to move forward into nominal operations.

Critical Event Readiness Review (CERR). Throughout Phase E. A CERR is held prior to any critical event outlined in the operations to ensure the system, ground facilities, support personnel and procedures are ready for the critical event.

Decommissioning Review (DR). Beginning of Phase F. The purpose of the DR is to confirm the decision to terminate or decommission the system and assess the readiness for the safe decommissioning and disposal of system assets.

12.3 Technical Reviews for Student-based Design Laboratories

For a student-based design laboratory, the minimum set of reviews is a few less than NASA requires, primarily because of the unusual time constraints on the students working in the laboratory. Although these are only the necessary technical reviews, it does not suggest that these are the only reviews that could be held. While the listed technical reviews can coincide with mandated reviews by the customer, the customer may request additional reviews for the project.

For each student technical review, all stakeholders (students, participating faculty, customers and "non-advocates") should be invited, unless otherwise indicated by the customer. A non-advocate is considered to be a person independent from the project, but they should have a similar technical background, and they should be invited for their skill and experience. Thus for student-based design laboratories, non-advocates often include non-participating faculty and student leaders apart from the project but recognized for their previous experience working in a student-based design laboratory.

If not mandated by the customer, the project student leadership is expected to set-up and lead reviews. Student leaders will be the main presenters, but will often be supported by other student members for specific technical material or demonstrations. Additionally, each review requires a different level of formality unless otherwise indicated by the customer. Informal reviews often consist only of presentations and discussions, whereas formal reviews require supporting documentation, prototypes or flight hardware to be provide ahead

of a scheduled presentation. During formal reviews, hardware demonstrations or simulations are expected. The required reviews for student-based design laboratories are as follows,

Mission Concept Review (MCR). The MCR is an informal review. For a successful MCR, mission objectives must be clearly defined and are unambiguous and internally consistent, the preliminary set of requirements shall flow from the mission objectives, and a conceptual solution has been identified which is technically feasible.

Preliminary Design Review (PDR). PDR is the first formal review. At PDR students must show the following: a complete flow down of verifiable requirements, a preliminary design that is expected to meet the requirements and demonstrate all required technology development is complete or valid back-up options exists. Prototype hardware and demonstrations of new technologies are expected at the review.

Critical Design Review (CDR). Again another formal review, passing CDR indicates a level of maturity to begin purchasing expensive flight hardware. To pass CDR, the detailed design is expected to meet the requirements, exhibit adequate technical and programmatic margins and resources and the product verification plans are complete. Refined hardware demonstrations and simulations are expected.

Test Readiness Review (TRR). A TRR is only needed if testing involves external organizations. Before performing the test, the TRR ensures

that adequate test plans are completed and approved, adequate identification and coordination of required test resources is completed and previous component, and subsystem test results form a satisfactory basis for proceeding into planned system-level tests.

Flight Readiness Review (FRR). A formal review performed immediately before delivery for the flight system. Often characterized as a safety review or a series of safety reviews. At FRR, students must show the flight vehicle is ready and verified for flight with a high probability for achieving mission success, flight and ground software elements are ready to support flight and flight operations and interfaces are checked out and found to be functional.

Operational Readiness Review (ORR). For missions requiring very active ground operations due to a time critical series of events, a formal review is required; if not, the review can be informal. The ORR must ensure all operational supporting and enabling products (facilities, equipment, documents, updated databases, etc) that are necessary for the nominal and contingency operations have been tested and delivered to support operations; as well as, training has been provided to the users and operators on the correct operational and contingency procedures for the system.

Critical Event Readiness Review (CERR). A CERR is needed only if mission operations includes a critical event and is held similarly to the ORR.

Decommissioning Review (DR) An informal review to confirm the decision between all stakeholders to decommission the system.

12.4 Example: Paradox Project Schedule

The Paradox project schedule is based on a January 2009 initialization and a projected January 2012 launch. The project life cycle phases and technical reviews are mapped over the expected three-year development timeline. At this time, an extra review, SRR, is required by the NASA sponsor and is noted on the schedule. Multiple TRRs are expected for environmental testing such as thermal vacuum and vibration tests since UT does not have those facilities. In addition, due to the time critical mission operations a formal ORR is expected prior to launch and separation. At the publishing of this thesis, Paradox was approaching the end of Phase A and had not gone through the SRR as indicated in Fig. 12.1.

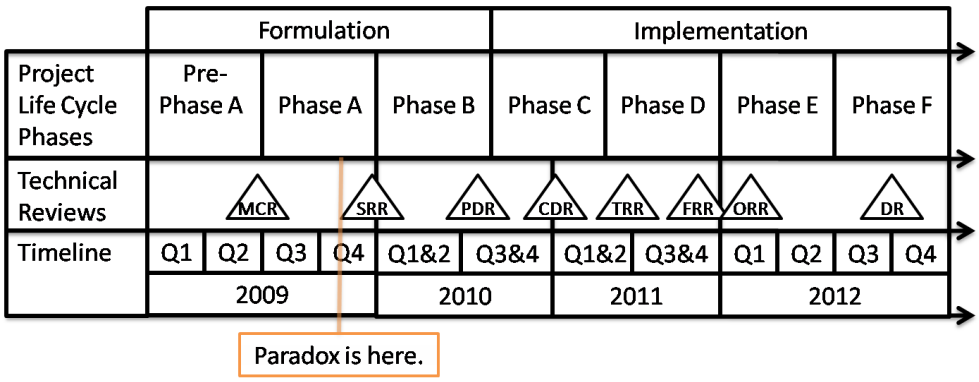

Figure 12.1: The Paradox Project Schedule.

Note that student-based design laboratory projects are often one of several secondary payloads on a launch vehicle; thus, there is an uncertainty

in both the launch vehicle to carry the mission and the expected launch date for the launch vehicle once manifested. Therefore, in light of the schedule in Fig. 12.1, an FRR and ORR can be a moving target for student-based design laboratory to prepare. Nevertheless, student-based design laboratory must be flexible and make preparations for when a launch vehicle becomes available and ready to launch.

Chapter 13

Iteration

The systems engineering process is iterative. As shown in Fig. 4.1 indicates, systems engineering feeds into the design, the design and development feeds into the systems effort and the cycle continues until the system is in operation.

The constant iteration applies within the project life cycle and at each level of a system. Although the project life cycle is linear from Pre-Phase A to Phase F, within each phase there is an extensive amount of iteration. As Chapter 12 indicated, iteration in the design for a student-based design laboratory is an important process, but iteration in the systems engineering is just as important to ensure a valid and successful design solution. The systems engineering and the design and development are intimately linked and dependent on one another.

Additionally, the systems engineering process is recursive. Each subelement of a larger system can itself be broken into elements and the systems engineering process outlined here can be applied to each element. The recursion is continued until the component level is reached within the system.

For the Paradox example, NASA started the process by outlining general expectations for which the SDL scoped the Paradox mission and generated the foundation for the system hierarchy. The high level mission objectives were

then identified and a concept for operations developed. The second iteration broadened the system hierarchy and functional requirements were allocated to each system. Mission analysis and trade studies were initiated and the concept of operations were further defined. The third iteration defined the subsystems within the satellite system, and functional and performance requirements were then derived to the subsystem level. At this stage, hardware trade studies and technology development was initiated and resource budgets were developed for all of the constrained technical resources and preliminary mission risks were assessed. The iteration continues until a final design solution is realized.

For a more specific example, the previous SDL project, Texas 2-STEP (refer to Section 2.2 for mission description), underwent a significant level of iteration between the Mission Concept Review and Preliminary Design Review. At the outset, the Texas 2-STEP mission architecture involved two "smart" satellites. A smart satellite can determine its position, velocity, acceleration, attitude and angular rotation on-board and in real-time. The only initial difference between the "target" satellite and "chaser" satellite, was that the chaser satellite had a requirement to perform orbit changing maneuvers to rendezvous with the target. Work ensued under the mission architecture and a concept of operations, systems hierarchy and set of requirements were derived to the subsystem level. In addition, hardware trade studies, technology development and design work supported the mission architecture.

An informal peer review prior to PDR allowed students and faculty to reflect on the mission concept. Invited, independent faculty took an in-depth look at the mission objectives and challenged the built-in complexities of the

mission architecture. The mission objectives only asked for a rendezvous to be performed with a physical target. The two "smart" satellite architecture levied unnecessary requirements on the target satellite. From a rescope of the mission architecture, functional allocation to the target satellite was significantly trimmed and most of the requirements were eliminated. Subsequently, the concept of operations, system hierarchy and design of the target radically changed according to the new requirements. The architecture change scaled the target design considerable and freed up much of the technical resources for the chaser design.

The iterative nature of systems engineering cannot be overly emphasized. The Texas 2-STEP example showcases an extreme case where a single iteration significantly changed the design. Paradox is already undergoing significant design iteration and just like design and development of a product, systems engineering must constantly be refined. As previous projects within the SDL have suggested (Chapter 2), performing systems engineering early is not sufficient for the success of a mission. The systems processes must continue to be practiced at every level of the system and throughout the project life cycle.

Chapter 14

Conclusion

NASA and other organizations have systems engineering practices that are tailored to their needs and to the organization itself. However, implementation by a student-based design laboratory of NASA's processes overwhelms students. This thesis presented a tailored NASA systems engineering "engine" for the unique environment of a student-based design laboratory.

Specific NASA systems engineering techniques are highlighted due to their relevance to student-based design laboratories while others are omitted. Additionally, each systems process is explained in the context of a student laboratory and implemented in the current project, Paradox, within the student-based Satellite Design Lab (SDL) at UT Austin. The examples successfully showcase the need and extent to which the NASA flavor of systems engineering can be tailored to a student laboratory. The processes are complete, providing discipline to a student project within the unique environment, and manageable for the student volunteer workforce with unusual time constraints and high turn-over rate.

The NASA systems engineering processes relevant to student-based design laboratories are: scope, system hierarchy, interfaces, requirements, trade studies, resource management, risk management, system verification and technical reviews. To summarize the tailored processes: Scope identifies the

unique relationship between a student-based design laboratory and its customer, which allows a student-based design laboratory flexibility in developing the goals and objectives for a mission and subsequently the architecture and concept of operations. A systems hierarchy simultaneously defines the structure of the end-product and the tasks for students to perform. Interfaces must be centralized into specific subsystem designs giving interface design responsibility to a select number of students. Requirements are a critical aspect to any project. However, for students the requirement flow down ends once the functional and performance requirements are fully developed. There is no need to write design specifications; instead document the design.

A direct link between systems engineering and the design, trade studies are a fundamental tool for student-based design laboratories to make correct decisions about the design with respect to mission scope. Resource management provides a system-level analysis for constrained technical resources to ensure a student mission is meeting constraints. Risk management is a qualitative way for student to analyze technical mission risks and develop design solutions to mitigate risks. System verification ensures the end-product meets requirements and subsequently meets stakeholder expectations. Last, a limited number of technical reviews provide a forum for the students, faculty, customer and other stakeholders to review progress and ensure the design and development will ultimately meet customer expectations.

The students within the SDL have enjoyed the freedom to challenge the very nature of the satellite business. Through this experience and with a deep desire, the SDL wishes to be more than just a sandbox for ideas, but to

be a center for mission success. The very motivation for the thesis has been to employ systems engineering practices to provide discipline and continuity for the projects within the SDL to ensure mission success. The continued challenge is to develop a systems engineering culture and utilize these processes in Paradox and future satellite missions developed by the dedicated students in the SDL.

Appendices

Appendix A

Target Centered Relative Reference Frame

The relative coordinate frame describes the position of the chaser spacecraft relative to the target spacecraft. The target centered rotating (TCR) reference frame is defined by three unit vectors, the first pointing in the direction of motion $\hat{\mathbf{u}}_{AT}$, the second perpendicular to the orbit plane in the direction of angular momentum $\hat{\mathbf{u}}_{CT}$, and the third radially outwards $\hat{\mathbf{u}}_r$. The unit vector are defined as:

$$\begin{aligned}
\hat{\mathbf{u}}_{AT} &= \frac{(\mathbf{r}_T \times \mathbf{v}_T) \times \mathbf{r}_T}{\|(\mathbf{r}_T \times \mathbf{v}_T) \times \mathbf{r}_T\|} \\
\hat{\mathbf{u}}_{CT} &= \frac{\mathbf{r}_T \times \mathbf{v}_T}{\|\mathbf{r}_T \times \mathbf{v}_T\|} \\
\hat{\mathbf{u}}_r &= \frac{\mathbf{r}_T}{\|\mathbf{r}_T\|}.
\end{aligned}$$

where $\mathbf{r}_T$ is the target spacecraft position vector in Earth Centered Inertial (ECI) coordinates, and $\mathbf{v}_T$ is the target spacecraft velocity.

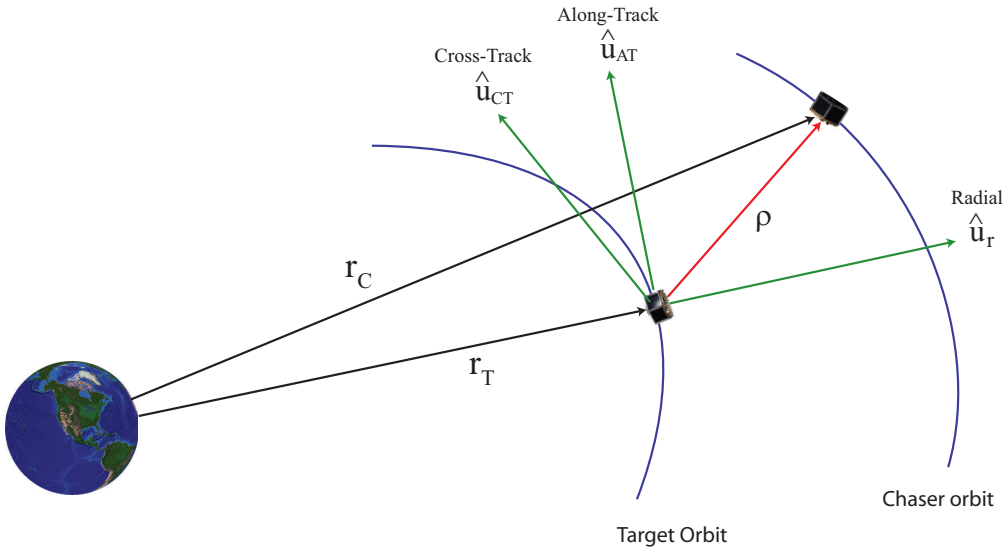

Figure A.1: Description of positioning in relative reference frame.[3]

Furthermore, transformations of relative positions and relative velocities from the inertial frame to the relative reference frame are realized using the transformation matrix $\mathbf{T}_{ECI}^{TCR}$:

$$\mathbf{T}_{ECI}^{TCR} = \begin{bmatrix} \hat{\mathbf{u}}_{AT} & \hat{\mathbf{u}}_{CT} & \hat{\mathbf{u}}_r \end{bmatrix}^T .$$

Bibliography

[1] NASA Headquarters, *System Engineering Handbook*, NASA/SP-2007-6105 Revision 1, Washington, D.C., December 2007.

[2] Battin, Richard H., *An Introduction to the Mathematics and Methods of Astrodynamics*, Revised Edition, AIAA Education Series, 1999.

[3] Silva, Elvis D., *Development of an Autonomous Guidance Algorithm for Nanosatellite Rendezvous Applications*, Masters Thesis, The University of Texas at Austin, May 2007.

[4] Loechler, Laura, *An Elegant Lambert Algorithm for Multiple Revolution Orbits*, Masters Thesis, Massachusetts Institue of Technology, May 1988.

[5] Shen, Haijun and Tsiotras, Panagiotis, *Using Battin's Method to Obtain Multiple-Revolution Lambert's Solutions*, AAS/AIAA Astrodynamics Specialists Conference, Big Sky, MO, August 3-7, 2003.

[6] Shen, Haijun and Tsiotras, Panagiotis, *Optimal Two-Impulse Rendezvous Using Multiple-Revolution Lambert Solutions*, Journal of Guidance, Control, and Dynamics, Vol. 26, No. 1, Jan-Feb 2003.

[7] Greenbaum, Jamin S., *Flight Unit Fabrication of a University Nanosatellite: The FASTRAC Experience*, Masters Thesis, The University of Texas at Austin, May 2006.

[8] Hinkley, David, *A Novel Cold Gas Propulsion System for Picosatellites and Nanosatellites*, Small Satellite Conference, Aug. 11-13 2008.

[9] Larson, Wiley J. and Wertz, James R., *Space Mission Analysis and Design*, Third Edition, Space Technology Series, 1999.

[10] Munakata, Riki, *CubeSat Design Specification*, Revision 12, The CubeSat Program, Cal Poly SLO, August 2009.

[11] Tsiotras, Panagiotis and Shen, Haijun *Satellite Attitude Control and Power Tracking with Energy/Momentum Wheels* Journal of Guidance, Dynamics and Control, Vol. 24, No. 1, Jan-Feb 2001.

[12] *Guide for the Preparation of Operational Concept Documents* ANSI/AIAA G-043-1992, American National Standard, January 22, 1993.

[13] Mankins, John C., *Technology Readiness Levels* Advanced Concepts Office, Office of Space Access and Technology, NASA, April 5, 1995.

[14] K. Sarda, S. Eagleson, E. P. Caillibot, C. Grant, D. D. Kekez, F. Pranajaya, and R. E. Zee, "Canadian Advanced Nanospace Experiment 2: Scientific and Technological Innovation on a Three-Kilogram Satellite," Acta Astronautica. Vol. 59, 2006, pp. 236-245.

[15] Guerra, Lisa A., Fowler, Wallace. *Space Systems Engineering for Aerospace Undergraduates* 46th AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada. 7 - 10 January 2008.

[16] B.E. Goldberg, K. Everhart, R. Stevens, N. Babbitt III, P. Clemens, and L. Stout. *System Engineering Toolbox for Design-Oriented Engineers* NASA Reference Publication 1358. December 1994.

[17] Goddard Space Flight Center. *Rules for the Design, Development, Verification and Operations of Flight Systems* GSFC-STD-1000 Revision A. May 30, 2005.

[18] NASA Office of the Chief Engineer. *NASA Systems Engineering Processes and Requirements* NASA Procedural Requirements NPR 7123.1. March 13, 2006.

[19] Dieter, George E., Schmidt, Linda C. *Engineering Design* Fourth Edition. McGraw-Hill Higher Education. 2009.

Vita

Michael Dax Garner[⋆] was born in Houston, Texas on March 11, 1986, the son of Ronald W. Garner and Priscilla C. Garner. Dax married the intelligent and beautiful Ashlea D. Majors on August 11, 2007, and they were blessed with the birth of their son, Azden Zayde Garner, on October 24, 2009.

Dax received his Bachelor of Science degree in Aerospace Engineering from The University of Texas at Austin in May 2008. As an undergraduate student, Dax worked as a co-op for L-3 Communications: Integrated Systems in Greenville, Texas. Quickly realizing his interests lie more in spaceflight, he sought an undergraduate research position in the Satellite Design Lab (SDL) at UT. In the fall of 2007, Dax unknowingly became the "systems engineer" for the student satellite project Texas 2-STEP and spent the rest of that year figuring out what the job entailed. Luckily in the spring of 2008, a piloted course entitled Space Systems Engineering did exactly that. The course uniquely provided him with useful tools on the student satellite project.

During his career as a graduate student, Dax became the teaching assistant (TA) for the Space Systems Engineering course. In his spare time, Dax worked as the Program Manager for the Texas 2-STEP until the satellite competition ended in January 2009. After Texas 2-STEP, Dax moved his experience to a fledgling small satellite program, Paradox, and became a research assistant (RA). As an RA, he supported Paradox in the development of the rendezvous guidance algorithm and ensuring the SDL properly implements the

systems engineering practices necessary for mission success.

Upon receiving his Masters of Science in Engineering Degree from The University of Texas at Austin in Aerospace Engineering, Dax plans to expand his knowledge of GNC and systems engineering in his career at Odyssey Space Research in Houston, Texas.

Permanent address: 16215 Madewood
Cypress, Texas 77429

This thesis was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.