Application of a Safety-Driven Design Methodology to an Outer Planet Exploration Mission

Brandon D. Owens, Margaret Stringfellow Herring, Nicolas Dulac, and Nancy G. Leveson
Complex Systems Research Laboratory
Massachusetts Institute of Technology
77 Massachusetts Avenue, Building 33-407c
Cambridge, MA 02139-4307
617-253-4519
owensbd@mit.edu, sapphire@mit.edu, ndulac@mit.edu, leveson@mit.edu

Michel D. Ingham and Kathryn Anne Weiss
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive, Mail Stop 301-225
Pasadena, CA 91109
818-354-5718
Michel.D.Ingham@jpl.nasa.gov, Kathryn.A.Weiss@jpl.nasa.gov

Abstract—Traditional requirements specification and hazard analysis techniques have not kept pace with the increasing complexity and constraints of modern space systems development. These techniques are incomplete and often consider safety late in the development cycle when the most significant design decisions have already been made. The lack of an integrated approach to perform safety-driven system development from the beginning of the system lifecycle hinders the ability to create safe space systems on time and within budget. To address this need, the authors have created an integrated methodology for safety-driven system development that combines four state-of-the-art techniques: 1) Intent Specification, a framework for organizing system development and operational information in a hierarchical structure; 2) the STAMP model of accident causation, a system-theoretic framework upon which to base more powerful safety engineering techniques; 3) STAMP-based Hazard Analysis (STPA); and 4) State Analysis, a model-based systems engineering approach. The iterative approach specified in the methodology employs State Analysis in the modeling of system behavior. STPA is used to identify system hazards and the constraints that must be enforced to mitigate these hazards. Finally, Intent Specification is used to document traceability of behavioral requirements and subject them to formal analysis using the SpecTRM-RL software package. In this paper,[1,2] the application of this methodology is demonstrated through the specification of a spacecraft high gain antenna pointing mechanism for a hypothetical outer planet exploration mission.

TABLE OF CONTENTS

1. INTRODUCTION

Conservative design has led to tremendous success in space exploration. As the complexity of spacecraft increases, however, and the use of new technology, particularly computers and software, increases, serious problems and even mission losses have resulted [1]. To handle this added complexity, new methodologies that design safety[3] into spacecraft systems from the beginning of the design process and that use model-based techniques to find logical design errors early are needed.

In this paper, a new methodology for safety-driven model-based systems engineering is presented and demonstrated through the top-down specification and analysis of a deep space exploration mission, focusing on the details of communications antenna pointing. The specification encompasses all aspects of the mission (i.e., spacecraft, launch vehicle, ground communications network, etc.) that influence this focus area.

Through the safety-driven specification process, the

[1] 1-4244-1488-1/08/$25.00 ©2008 IEEE.
[2] IEEEAC paper #1279, Version 8, Updated December 14, 2007

[3] Note that the term safety is used here in the broad sense of reducing the risk of mission failure.

spacecraft-to-Earth communication function was allocated to a high gain antenna (HGA) located on the end of a deployable boom and subject to disturbances from other spacecraft subsystems and the spacecraft's operating environment. Through the early consideration of safety in this process, the hazards associated with such HGA pointing operations were addressed by selecting a design option that reduced both complex coupling of spacecraft pointing functions and the relevance of orbital debris generation. Additionally, the final specification produced in this study documents the traceability between many of the traditional systems engineering artifacts produced in a design specification effort as well as those that were previously unique to one or more of the systems engineering frameworks described in the next section.

2. BACKGROUND

The methodology used in this study draws from four state-of-the-art systems engineering techniques: MIT's accident causation model STAMP, Intent Specification, STAMP-based Hazard Analysis, and JPL's State Analysis. In this section, each of these frameworks is described individually. For more details on this methodology and its relation to these systems engineering techniques, refer to [2].

Intent Specifications

Intent Specification is a specification and development framework to provide support for system design and other system engineering activities and to provide more readable and reviewable specifications. Intent specifications are based on research in human problem solving and on basic principles of system theory and systems engineering [3].

An intent specification differs from a standard systems engineering specification primarily in its structure, which is designed to 1) facilitate the tracing of system-level requirements and design constraints down into detailed design and implementation and the documentation of design rationale, 2) assist in the assurance of various system properties (such as safety) in the initial design and implementation, and 3) reduce the costs of implementing changes and subsequent re-analysis when the system is changed, as it inevitably will be. Intent specifications contain the same information as would be found in traditional requirements artifacts; no extra specification is involved (assuming that projects produce the usual specifications). Intent specifications simply use a different structuring and linking of the information so that the specifications provide more assistance in the design and evolution process.

As shown in Figure 1, there are seven levels in an intent specification. These levels do not represent refinement, but completely different models of the same system that support a different type of reasoning about the system (i.e., each model or level presents a complete view of the system, but from a different perspective). The model at each level is described in terms of a different set of attributes or language. Refinement and decomposition occurs within each level of the specification. In addition to intra-level refinement, the levels are organized in a "Means-Ends" hierarchy. In such a hierarchy, the information at a level acts as the goals (the ends) with respect to the model at the next lower level. In other words, the next lower level is where the means to the ends of the current level are implemented [3].

	Environment	Operator	System and Components	V&V
Level 0	Project management plans, status information, safety plans, etc.			
Level 1: System Purpose	Assumptions Constraints	Responsibilities Requirements I/F Requirements	System Goals, High-Level Requirements, Design Constraints, Limitations	Hazard Analysis
Level 2: System Design	External Interfaces	Task Analyses Task Allocation Controls/displays	Logic Principles, Control Laws, Functional Decomposition and Allocation	Validation Plans and Results
Level 3: Blackbox Models	Environment Models	Operator Task and HCI Models	Blackbox Functional Models, Interface Specifications	Analysis Plans and Results
Level 4: Design Representation		HCI Design	Software and Hardware Design Specifications	Test Plans and Results
Level 5: Physical Representation		GUI and Physical Controls Designs	Software Code, Hardware Assembly Instructions	Test Plans and Results
Level 6: Operations	Audit Procedures	Operator Manuals Maintenance Training Materials	Error Reports, Change Requests, etc.	Performance Monitoring and Audits

Figure 1. Intent Specification Hierarchy.

The top level (Level 0) provides a project management view and insight into the relationship between the plans and project development. Level 1 of an intent specification is the customer view and assists system engineers and customers in agreeing on what should be built and whether that has been accomplished. It includes system goals, high-level requirements, design constraints, hazards, environmental assumptions, and system limitations.

Level 2, System Design, is the system engineering level and provides the structure and content needed for engineers to reason about the system in terms of the physical principles and laws upon which the system design is based. It documents the basic system-level design decisions made to satisfy the requirements and constraints at Level 1.

The third level, or Blackbox Behavior level, enhances reasoning about the logical design of the system as a whole and the interaction among the components as well as the functional state without distractions from implementation issues. This level acts as an unambiguous interface between systems engineering and component engineering to assist in communication and review of component blackbox behavioral requirements and to reason about the combined behavior of individual components using informal review, formal analysis, and simulation. The models at this level are formal and can be both executed and subjected to formal

analysis (for example, completeness and consistency analyses).

The next two levels (4 and 5) provide the information necessary to reason about individual component design and implementation issues. Finally, the sixth level provides a view of the operational system. The study described in this paper has predominantly focused on Levels 0-3 of the Intent Specification. Levels 4 and 5 represent the standard component documentation used on most any engineering project.

Figure 2 shows an example of intent specification traceability between Levels 0, 1, and 2 through partial specification of a spacecraft capable of landing on a planet surface. Traceability is captured through hyperlinks denoted by arrows and the specification item tag (for example, ↓H1). Traceability links denote different relationships between specifications based on their direction. An up arrow (↑) denotes that the current specification item is involved in the implementation of the intent of a specification item at a higher level in the "means-ends" hierarchy denoted by the tag after the arrow. A down arrow (↓) points to a specification item at a lower level in the "means-ends" hierarchy that is involved in the implementation of the intent of the current specification item. Left and right arrows denote relationships between specification items at the same level in the "means-ends" hierarchy that affect the items' relationship to items on other levels. The direction of the arrow for this type of relationship depends on the physical location of the specification item in the intent specification document. A left arrow (←) points to a specification item at the same level that appears earlier in the specification than the current specification item. Conversely, a right arrow (→) points to another specification item at the same level that appears later in the current specification document. Thus, in Figure 2, the hazard (H1 at Level 1) will show an upwards link pointing to an accident (ACC1 at Level 0). This relationship shows 'why' the hazard is of concern: it can lead to the accident ACC1 shown in Level 0. The accident has a downward arrow pointing to H1 showing 'how' the accident could occur. Similarly, H1 points across the level to a safety constraint (SC1) derived from the hazard. The safety constraint has downward pointing links to Level 2 where that safety constraint is enforced with system design decisions. Lastly, the relationship between the design decisions is captured through traces across Level 2.

Intent information represents the design rationale upon which the specification is based and is integrated directly into the specification. Each level also contains information about underlying assumptions upon which the requirements, design, and validation is based. Assumptions are especially important in operational safety analyses. When conditions change such that the assumptions are no longer true, then a new safety analysis should be triggered. In the traditional system engineering specification approach, these assumptions may be included in a safety analysis document, but are not usually traced to the parts of the implementation they affect. Thus, even if the system safety engineer knows that a safety analysis assumption has been changed, it is a very difficult and resource-intensive process to figure out which parts of the design used that assumption.

In summary, intent specifications allow a seamless transition from system to component (including software) specifications and the integration of formal and informal aspects of system and software development. The specification structure: 1) facilitates the tracing of system-level requirements and constraints into the design and the assurance of various system properties (such as safety) in the initial design and implementation and 2) reduces the costs of implementing changes and re-analysis.

Level 0
Accidents (ACC):
ACC1. The spacecraft is damaged or destroyed. (↓H1)

Level 1
Hazards (H):
H1. The spacecraft comes in contact with the planet surface at a speed greater than 10 m/s. (↑ACC1), (→SC1)

Safety Constraints (SC):
SC1. The spacecraft must not contact the planet surface at a speed greater than 10 m/s. (←H1), (↓DD1, DD2)

Level 2
Design Decisions (DD):
DD1. A vertical velocity sensor with a range of -15 m/s to 15 m/s measures the spacecraft velocity during descent to the planet surface. (↑SC1), (→DD2)

DD2. If a vertical velocity greater than 10 m/s is detected, the spacecraft transitions to "Landing Contingency Mode". (↑SC1), (←DD1)
…

Figure 2. An example of intent specification traceability.

STAMP

STAMP, which stands for System-Theoretic Accident Modeling and Process, is an accident causality model in which accidents are conceived as resulting not from component failures, but from inadequate control or inadequate enforcement of safety-related constraints on the design, development, and operation of the system [4]. Instead of viewing accidents (as is traditional in engineering) as the result of an initiating, or root cause, event in a series of events leading to a loss, accidents are viewed as resulting from interactions among components that result in a violation of system safety constraints. In STAMP, safety is viewed as a control problem. Accidents occur when component failures, external disturbances,

and/or dysfunctional interactions among system components are not adequately handled or controlled. The control processes that enforce the safety constraints must limit system behavior to the safe states implied by the safety constraints.

Figure 3 shows a generic (example) control structure to enforce safety constraints. Each hierarchical level of the control structure represents a control process and control loop with actions and feedback. Two control structures are shown in Figure 3, system development and system operations, both of which have different responsibilities with respect to enforcing safe system behavior.

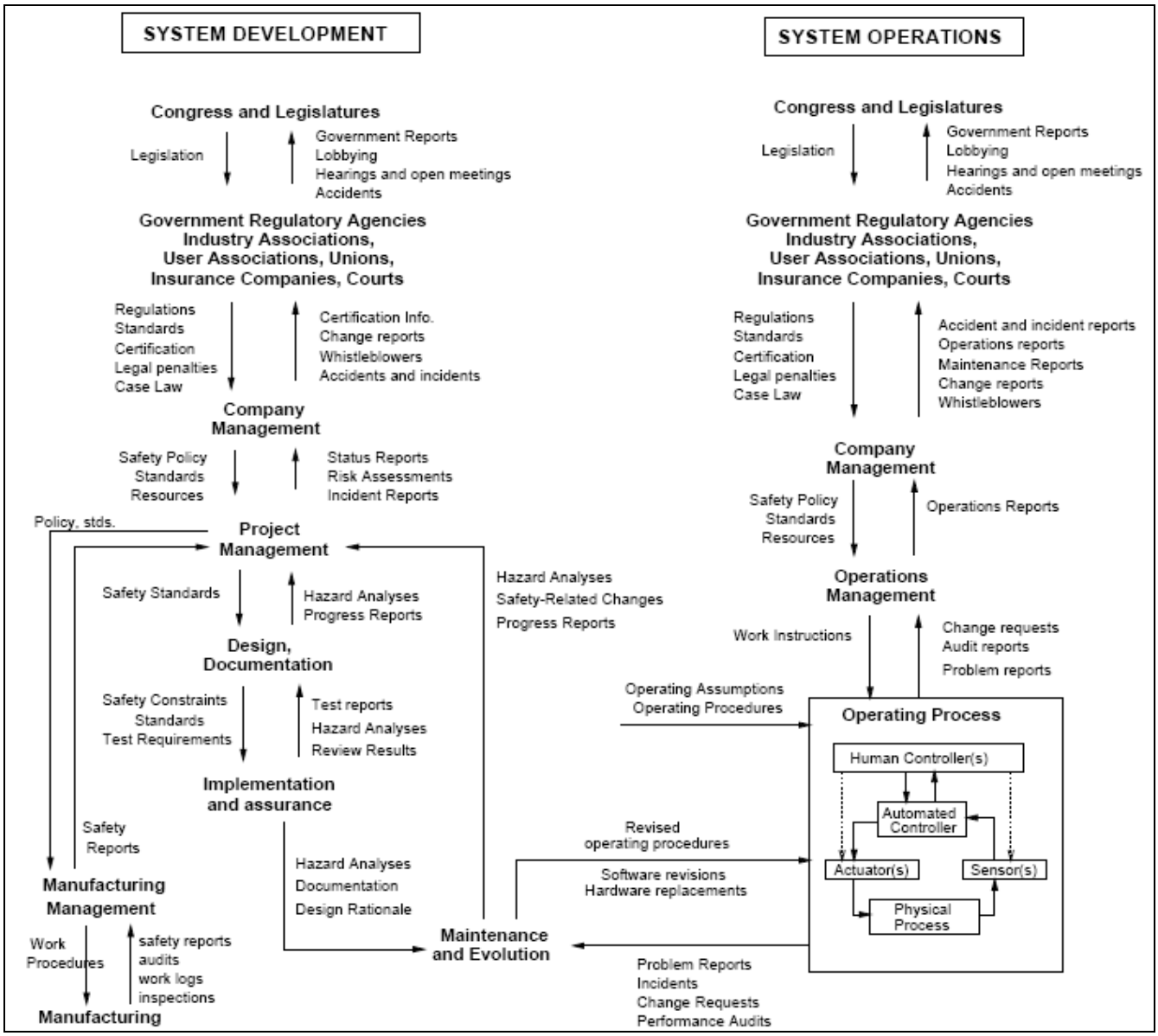


Figure 3. Generic System Control Structure [4].

STAMP treats a system not as a static design, but as a dynamic process that is continually adapting to achieve its ends and to react to changes in itself and its environment. The original design must not only enforce appropriate constraints on behavior to ensure safe operation, but the system must continue to operate safely as changes and adaptations occur over time. Furthermore, any controller—human or automated—must contain a model of the system being controlled. The model of the process (the plant, in control theory terminology) at one extreme may contain only one or two variables (such as that required for a simple thermostat) while at the other extreme it may require a complex model with a large number of state variables and transitions (such as that needed for air traffic control). Whether the model is embedded in the control logic of an automated controller or in the mental model of a human controller, it must contain the same type of information: the current state (the current values of the system variables), the ways the process can change state (the system dynamics), and the desired relationship among the system variables (the

control laws). This model is used to determine what control actions are needed and it is updated through various forms of feedback. When the model does not match the controlled process, accidents can result [5, 6].

STPA

The objectives of STPA (STAMP-Based Hazard Analysis) are the same as that of a traditional hazard analysis (as described in [6]): 1) to identify the system hazards and the safety-related constraints necessary to ensure acceptable risk and 2) to accumulate information about how the safety constraints may be violated and use this information to eliminate, reduce, and control hazards in the system design and operation [7]. Although the first steps of the STPA are similar to those performed in other hazard analysis techniques, the later steps either deviate from traditional practice or provide a guiding framework for doing what is traditionally done in an ad hoc manner.

In essence, STPA starts with a hazard and its related requirement or constraint. The STPA taxonomy, described below, is used to identify instances of inadequate control and the control flaws and/or inadequate control executions that lead to inadequate control. From there, engineers refine requirements or constraints and create new design until all hazards are mitigated, eliminated, or controlled. Engineering judgment is used to determine when the design is "safe and complete enough."

Underlying the STPA process is the notion that the presence of hazards is eliminated or controlled through system design. Figure 4 presents a generic low-level process control loop in STPA. Each item in the STPA taxonomy, upon which the hazard analysis is based, relates to an element of the control loop. As seen in the figure, the control input is a reference signal. The controller uses the control input in conjunction with received measurements to generate commands. Continuing along the loop, the command is sent to the actuator, which implements the command through the arrow labeled U. The U vector refers to actions of the actuator that influence the controlled process. The control algorithm used by the controller is based on an internal process model of the controlled process. The controlled process, or plant, is subject to process inputs and disturbances. The process output may become an input into another linked process control loop. The sensors measure the output resulting from the actuator's actions and disturbances to generate measurements that are then fed into the controller.

Depending on the particular system, the control input, usually a set point, is often referred to as a goal, plan, sequence, or directive in spacecraft engineering parlance. The controller may send directives to a lower-level controller rather than an actuator in order to affect control on that process. Similarly, a controller may receive measurements (e.g., health status) from a lower-level controller, rather than a sensor.

STAMP is based on the concept of controlling hazards rather than eliminating component failures (which are only one cause of hazards). When a safety constraint is violated, the hazard can occur and accidents can happen. Returning to the example in Figure 2, the relevant hazard is "The spacecraft comes in contact with the planet surface at a speed greater than 10 m/s." The spacecraft could be inadequately controlled and the hazard state could occur if, for example, the controller commands the thrusters such that the spacecraft speed is 11 m/s when the spacecraft reaches the surface. A control flaw, such as an incorrectly calibrated velocity sensor, could contribute to inadequate control of the landing process. The concepts of inadequate control and control flaws are discussed below.

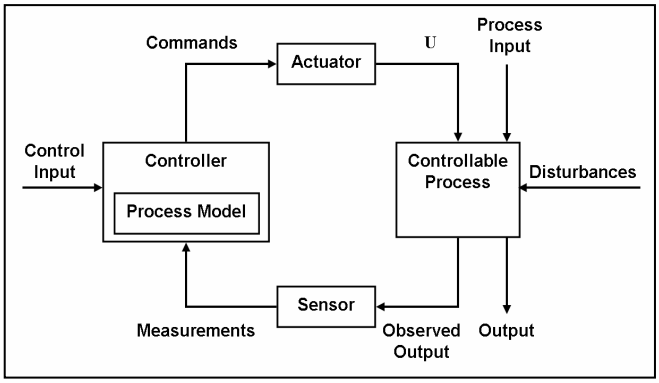


Figure 4. Generic STPA Low-Level Process Control Loop.

Each hazard and related safety constraint is analyzed using STPA. First, inadequate control actions that could violate the safety constraint and result in a hazardous state are identified. In general, there are four types of inadequate control [6]:

1. A required control action is not provided or is inadequately executed.
2. An incorrect or unsafe action is provided.
3. A potentially correct or adequate control action is provided too late or at the wrong time.
4. A correct control action is stopped too soon or continued too long.

Next, the identification of control flaws is conducted, using knowledge of the system design developed so far. *Control flaws* are the mechanisms that could instantiate or lead to the inadequate control actions. Typically, the following are the general ways in which an inadequate control action could occur and are used to guide the hazard analysis process:

1. Design of the control algorithm does not enforce constraints
    a. Flaw(s) in creation process
    b. Process changes without appropriate change in control algorithm (asynchronous evolution)
    c. Incorrect modification or adaptation
2. Process models are inconsistent, incomplete, or incorrect
    a. Flaw(s) in creation process

b. Flaw(s) in updating process
c. Inadequate or missing feedback
    i. Not provided in system design
    ii. Communication flaw
    iii. Time lag
    iv. Inadequate sensor operation
d. Time lags and measurement inaccuracies not accounted for
e. Expected process inputs are wrong or missing
f. Expected control inputs are wrong or missing
g. Disturbance model is wrong
    i. Amplitude, frequency, or period is out of range
    ii. Unidentified disturbance

3. Inadequate coordination among controllers and decision makers

In the early stages of Intent Specification, few design decisions have been made and control flaws may not yet be identified. However, performing STPA early will allow the results of the hazard analysis to inform the design process.

Inadequate control execution is another cause of inadequate control (violation of safety constraints). Inadequate control execution pertains to physical component failures such as motor failures, or sluggish response of the motor (perhaps an indicator that the motor will soon fail) or communication line failures between components. Inadequate control execution can occur when the process model is correct and the correct control action is selected and applied, but the control action is still not successfully applied due to inadequate actuator operation, time lags, or communications flaws beyond the scope of the process model. For example, if the controller sends the command to increase thrust of the landing spacecraft, then the inadequate control action "Spacecraft provides too little thrust" will occur due to inadequate actuator operation. In general, inadequate execution of control actions can take the following forms:

1. Communication flaw
2. Inadequate actuator operation
3. Time lag

Once control flaws and inadequate control executions have been identified, some combination of the following three actions are taken to eliminate, mitigate, or control the related hazard:

1. Refine the related requirement or safety constraint.
2. Create a new design to eliminate, prevent, or mitigate the effect of the control flaw.
3. Record the rationale that the design has been accepted as is.

Note that one new design decision can address several control flaws or inadequate control executions. Also note that one control flaw can lead to several inadequate control actions.

STPA should be performed iteratively and opportunistically. Engineers can either drill down into one particular hazard they wish to control or apply STPA more broadly across several hazards. The results of STPA are documented in the hazard log in Level 1 of the intent specification.

SpecTRM & SpecTRM-RL

In this study, a commercial systems engineering toolset called SpecTRM (Specification Tools and Requirements Methodology) was used to capture intent specifications [5]. SpecTRM focuses on the early stages of system development, where the foundation is set for later implementation, operations, and maintenance activities. The SpecTRM toolset includes support for requirements development and management, hazard analysis, requirements tracing (within the document and to external documents), recording of design rationale, and modeling and analysis of blackbox logic requirements.

SpecTRM includes a formal modeling language, SpecTRM-RL (SpecTRM Requirements Language) to model the system blackbox behavior described at Level 3 of the intent specification. SpecTRM-RL has a formal foundation so models built in this language can be executed and subjected to formal analysis while still being readable with minimal training and expertise in discrete math. In addition, the models are analyzable, and tools have been developed to check for completeness, consistency, and robustness.

SpecTRM-RL was designed to satisfy two objectives: to be easily readable enough to serve as part of the official specification of the blackbox behavioral requirements and, at the same time, to have an underlying formal model that can be executed and subjected to mathematical analysis. This underlying formal model, which we call the requirements state machine (RSM), is very low-level and not appropriate as a specification language for complex systems. Instead, SpecTRM-RL acts as the specification language (or visualization of the underlying model) that overlies the low-level model. As long as the mapping from SpecTRM-RL to the RSM is unambiguous and well-defined, formal analysis is possible on both the underlying RSM formal model as well as the higher-level SpecTRM-RL specification itself.

The conditions under which an output is triggered (sent) can be specified by a predicate logic statement over the various states, variables, and modes in the specification. In our experience in specifying complex systems, however, we found that the triggering conditions required to accurately capture the requirements are often extremely complex. We also found propositional logic notation did not scale well to complex expressions in terms of readability and error-proneness. To overcome this problem, we developed a tabular representation of disjunctive normal form (DNF) that we call **AND/OR** tables.

Figure 5 illustrates an example of a SpecTRM-RL **AND/OR**

table specification; it defines the criteria for the transition of a spacecraft camera state into an *Idle* mode. The far-left column of the **AND/OR** table lists the logical phrases of a predicate logic statement. Each of the other columns is a conjunction of those phrases and contains the logical values of the expressions. The rows of the table represent **and** relationships while the columns represent **or** relationships. The state variable takes the specified value (in this case, *Idle*) if any of the columns evaluate to true. If one of the columns evaluates to *true*, then the entire table evaluates to *true*. A column evaluates to true if all the rows have the value specified for that row in the column. An asterisk denotes "don't care" while 'T' and 'F' denote true and false, respectively. Underlined variables represent hyperlinks. For example, clicking on Camera-State would show how the 'Camera-State' state variable is defined in the intent specification.

= Idle

Previous Value of Camera-State in state Off	*	F	F
Turn-On-Camera-Command was Received	T	*	*
Previous Value of Power-Bus-State in state Powered	T	T	T
Time Since Camera-State Last Entered Ready >= 10 seconds	F	T	F
Previous Value of Camera-State in state Ready	F	T	T
Go-Idle-Camera-Command was Received	*	F	T

Figure 5. **AND/OR** Table for Camera State Transition Logic.

In the example described in Figure 5, the camera is only able to transition into *Idle* mode if: 1) the Camera was previously *Off*, the 'Turn-On-Camera-Command' was received, and the power bus is delivering power to the camera; or 2) the camera has been in *Ready* mode for at least 10 seconds and the power bus is delivering power to the camera; or 3) the camera has been in *Ready* mode for less than 10 seconds, the 'Go-Idle-Camera-Command' was received, and the power bus is delivering power to the camera.

The **AND/OR** tables used in the blackbox models describe the conditions for transitioning between states and the values of inputs and outputs. Visualizations for black box models can also be created in SpecTRM (refer to Figure 23 in the methodology section of this paper for an example). These visualizations show all of the inputs, outputs, control modes, inferred state variable values, and other controllers or devices necessary for the control of the relevant process. Each state variable, input, and output has a model described in part by an **AND/OR** table in Level 3 of the intent specification. Level 3 formally defines the control system behavior and includes the transitions between values of an inferred state variable and control modes. It also includes timing constraints, descriptions, state variable macros, and functions for the value calculation of continuous state variables.

State Analysis

A novel, model-based systems engineering methodology, called *State Analysis*, has been developed to complement traditional functional decomposition approaches and better address the challenges of increasingly complex systems [8]. It provides a methodical and rigorous approach for:

- Modeling behavior in terms of system state variables and the relationships between them (*state-based behavioral modeling*);
- Capturing mission objectives in detailed scenarios motivated by operator intent (*goal-directed operations engineering*); and
- Describing the methods by which objectives will be achieved (*state-based software design*).

For the study described in this paper, we have focused primarily on the state-based behavioral modeling and state-based software design aspects of State Analysis[4]. The state-based behavioral modeling aspect provides an iterative process for *identifying state variables of the system under control* and for *incrementally constructing the model of the physical behavior of the system under control* associated with these state variables. The steps in this process, which is referred to as "State Discovery," include the following:

1. Identify needs – define the high-level objectives for controlling the system.
2. Identify state variables that capture what needs to be controlled in order to meet the objectives and define their representation.
3. Define models for the identified state variables—these may uncover additional state variables that affect the identified state variables.
4. Identify measurements needed to estimate the state variables and define their representation.
5. Define measurement models for the identified measurements—these may uncover additional state variables.
6. Identify commands needed to control the state variables and define their representation.
7. Define command models for the identified commands—these may uncover additional state variables.
8. Repeat steps 2-7 on all newly discovered state variables until all relevant variables and effects are accounted for.
9. Return to step 1, this time to identify additional objectives and proceed with additional iterations of

[4] Future work is needed to address the goal-based operations aspect of State Analysis and how that can be incorporated with the intent specification approach.

the process until the scope of the mission has been covered.

This modeling process can be used as part of a broader, iterative incremental system and software development process, in which cycles of the modeling process can be interwoven with concurrent cycles of software implementation.

The state-based software design aspect of State Analysis involves using these behavioral models to develop requirements and specify algorithm-level designs for the fundamental control system functions required to achieve the specified objectives. These control system functions map to the state-based control architecture, shown in Figure 6.

Each state variable, estimator, controller, and hardware adapter represents a component of the control system. State Analysis defines an interconnection topology among the components in each control loop according to canonical patterns and standard interfaces; furthermore, the causal effects between state variables captured in the behavioral models can be used to specify appropriate interfaces between the corresponding control loops.

There are significant software assurance benefits to using a development methodology and architecture that share the same basic structure, namely an unprecedented level of coordination and control of the systems and software development processes. For example, requirements are cleanly partitioned and traceable directly to implementation, making it easy to track and manage each step in the development process. Verification and validation exploits the same explicit structure, as well as the objective specification of each system element and the overt declaration of success criteria at all levels of operation.

State Analysis produces and compiles information that is traditionally documented in a variety of systems engineering artifacts, including Hardware Functional Requirements, Failure Modes and Effects Analyses, Command Dictionaries, Telemetry Dictionaries, and Hardware-Software Interface Control Documents.

In summary, the State Analysis approach is novel and unique in three ways:

1. It is based on a state-based control architecture (see Figure 6) and leverages a core set of underlying architectural principles.
2. When coupled with software that adopts the state-based control architecture, State Analysis provides a common vocabulary for systems and software engineers to communicate, and a common set of architectural elements such that the gap between the requirements provided by the systems engineer and the software developed by software engineer can be minimized. More specifically, it defines *direct mappings* from the system requirements (expressed in the form of behavioral models) to software specifications, and from these software specifications to implemented software artifacts.
3. It considers the full breadth of system state variables (e.g., dynamics, environmental states, device status and health, parameters, resources, etc.) and allows for documentation of models using whatever representation is most appropriate (differential equations, state charts, tables, pseudo-code, textual descriptions, etc.).

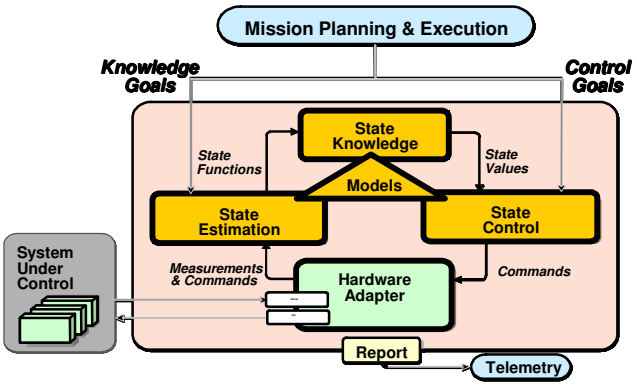


Figure 6. Conceptual View of the State-Based Control Architecture.

3. METHODOLOGY

The integration of Safety-Driven Design based on STAMP and STPA, Intent Specification, and State Analysis has the potential for powerful synergy as a systems engineering methodology. It assists engineers in 1) generating requirements aimed at hazard elimination and mitigation *concurrently* with generating functional requirements and spacecraft specification and 2) designing safety into the spacecraft from the beginning of the design process rather than adding on fault protection after the fact.

A roadmap for progressing through the integrated methodology is provided in Table 1. In the remainder of this section, each step of the integrated methodology is presented in the context of an example of a hypothetical NASA mission to explore an icy moon of an outer planet in our solar system. Hereafter, this mission will be referred to as the Outer Planets Explorer (OPE) mission. For more details on this methodology and other examples, refer to [2].

Table 1. Roadmap for Methodology Progress and Output

Step 1: Identify Mission Goals, Requirements, and Constraints.
Products: Level 1 intent specification of mission goals and constraints

Step 2: Define System Accidents or Unacceptable Losses.
Products: Level 0 intent specification documenting the accidents.

Step 3: Define High-level Hazards.
Products: Level 1 intent specification documenting high-level hazards.

Step 4: Define High-level Safety-Related Constraints.
Products: Level 1 intent specification documenting safety constraints.

Step 5: Identify Environment and Customer Constraints.
Products: Level 1 intent specification of environmental constraints and environmental assumptions, customer-derived system design constraints, and customer programmatic constraints.

Step 6: Perform High-level Functional Decomposition.
Products: Level 1 intent specification documenting the functional decomposition.

Step 7: Design High-level System Control Structure.
Products: Level 1 intent specification documenting the high-level control structure.

Step 8: Perform Preliminary Hazard Analysis using STPA and Create Hazard Log.
Products: Level 1 intent specification documenting STPA hazard analysis.

Step 9: Define System Element Specifications.
Products:
- Level 1 intent specification documenting goals, requirements, design constraints, and safety constraints for each subsystem or functional element (including subsystems and/or functional elements defined both before Step 9 and during the iterative sub-steps of Step 9).
- Level 2 intent specification documenting design decisions made to implement the requirements and constraints in Level 1.
- Level 3 intent specification documenting the formal design of the control system.

Step 10: Perform Validation Tests.
Products: Test results.

Step 11: Generate Designs and Software Code.
Products: Design specifications and software code

Step 1: Identify Mission Goals, Requirements, and Constraints

The starting point for the methodology is the definition of *mission goals* that we want or require the mission to achieve and *constraints* on the mission that must not be violated in the efforts to achieve the mission goals. Both mission goals and constraints can be found in a number of standard information sources such as: prior architecture studies, reused mission requirements, governmental mandates, corporate policies, laws, and standard system safety practices.

For example, a set of science goals for a Europa orbiter mission is described in [9]. The goals listed below in Figure 7 are generalized from those goals in [9] for our example of the exploration of icy moons of outer planets.

Mission goals and constraints are documented in Level 1 of the intent specification. Figure 7 contains examples of mission goals with links across Level 1 pointing to high-level requirements using the OPE example (refer to Appendix A for a description of the notation used in the OPE Intent Specification).

G1. Characterize the presence of a subsurface ocean on an icy moon of an outer planet.
(↑ACC4, ACC5), (→HLR3, HLR4), (↓SV-81)

G2. Characterize the three-dimensional configuration of the icy crust of the icy moon of an outer planet, including possible zones of liquid. (↑ACC4, ACC5), (→HLR1, HLR2, HLR3)

G3. Map organic and inorganic surface compositions of the icy moon of an outer planet, especially as related to astrobiology. (↑ACC4, ACC5), (→HLR2, HLR3)

G4. Characterize surface features of the icy moon of an outer planet and identify candidate sites for future exploration. (↑ACC4, ACC5), (→HLR1, HLR2, HLR3)
…

Figure 7. A Sample of Level 1 Mission Goals for OPE.

Step 2: Define System Accidents or Unacceptable Losses

To derive the safety requirements and design constraints, the engineer first defines system accidents or unacceptable losses. These losses may include loss of life, mission, or damage to the environment. Figure 8 contains several accident definitions for this example. System Accidents are documented in Level 0 of the intent specification.

ACC1. Humans and/or human assets on earth are killed/damaged. (↓PC1, H5, SV-77, SV-78, SV-79)

ACC2. Humans and/or human assets off of the earth are killed/damaged. (↓PC1, H6, SV-77, SV-78, SV-79)

ACC3. Organisms on any of the moons of the outer planet (if they exist) are killed or mutated by biological agents of Earth Origin. (↓H4)

ACC4. The scientific data corresponding to the mission goals are not collected. (↓G1, G2, G3, G4, G5, G6, G7, H1, SV-80)

ACC5. The scientific data corresponding to the mission goals is rendered unusable (i.e., deleted and/or corrupted) before it can be fully investigated. (↓G1, G2, G3, G4, G5, G6, G7, H2, H3, SV-80)

…

Figure 8. A Sample of Defined System Accidents for OPE.

Step 3: Define High-Level Hazards

Next, the engineer defines the high-level hazards that could lead to system accidents, prevent mission goals and requirements from being met, or violate the mission constraints. Figure 9 lists some of the high-level hazards that were derived using these mission objectives in combination with standard system safety engineering principles and analysis of the accidents identified above. System-level hazards are documented in Level 1 of the intent specification. Then, from these high-level hazards, the high-level safety constraints are defined.

H1. Inability of Mission to collect data. (↑ACC4), (↓SV-85)

H2. Inability of Mission to return collected data. (↑ACC5), (↓SV-86)

H3. Inability of Mission scientific investigators to use returned data. (↑ACC5), (↓SV-87, SV-88)

…

Figure 9. A Sample of High-Level Hazards for OPE.

Step 4: Define High-Level Safety-Related Constraints

The safety constraints are requirements that eliminate or mitigate the hazard. For instance if the hazard is of the form "Hazardous state occurs," it involves a simple (and often trivial but important) translation from the hazard into an engineering goal. For example, if the hazard is "Spacecraft comes in contact with planet surface at a speed greater than 10 m/s," the corresponding safety constraint could be "The spacecraft must not contact the planet surface at a speed greater than 10 m/s." An example from the OPE Intent Specification is shown below in Figure 10. High-level safety constraints are documented in Level 1 of the intent specification.

H1. Inability of Mission to collect data. (↑ACC4), (↓SV-85)

SC1. The mission must have the necessary functionality for data acquisition at the required times. (←H1), (→MOC-G1, MOC-G2, MOC-G3, MOC-G4), (↓2.1, 2.2, 2.4, SV-85)

H2. Inability of Mission to return collected data. (↑ACC5), (↓SV-86)

SC2. The mission must be able to return data at the required times. (←H2), (→MOC-G1, MOC-G2, MOC-G3, MOC-G4), (↓2.1, 2.3, 2.4, 2.5, SV-86)

…

Figure 10. Sample High-Level Hazards and Safety Constraints for OPE.

Step 5: Identify Environmental and Customer Constraints

Next, the engineers identify:

1. environmental constraints and environmental assumptions,
2. customer-derived system design constraints, and
3. customer programmatic constraints (e.g., budgets, etc.)

Environmental descriptions, constraints, and assumptions describe and constrain the environment of the system and are design independent. If the goal of the mission is to explore an outer planet, information regarding temperature, atmospheric pressure, and gravity would be documented in this section. In addition, this kind of information would be documented for other environments that the mission will encounter before and after the primary mission in the orbit of the outer planet moon. An example of an environmental description, constraint, and assumption can be found in Figure 11. Environmental constraints are documented in Level 1 of the intent specification.

Magnetic Flux: The magnetic field surrounding the icy outer planet moon to be studied is poorly understood and different from the magnetic field in Low Earth Orbit.

EC.1. The mission elements must withstand a solar flux of TBD Watts/m^2, the solar flux in the orbit of the outer planet moon. (↓SV-1, SV-83)

EA.1. The translation and rotation of the moon of study with respect to the Sun and relevant outer planet will be relatively stable over the mission and thus predictable.

…

Figure 11. Sample Environmental Description, Constraint, and Assumption for OPE.

Customer-derived system design constraints are constraints on the design of the system that are technical in nature. Typically, they involve how the system must interact with existing resources or engineering mandates or initiatives the customer wishes to implement. An example of customer-derived design is shown in Figure 12. Customer-derived design constraints are documented in Level 1 of the intent specification.

DC1. The mission must be carried out with existing technologies and space exploration infrastructures as needed (i.e., technologies rated at Technology Readiness Level TBD as defined by NASA). (↓2.1)
Rationale: While technology development is expected to be an ongoing activity of NASA, it is assumed to be beyond the mandate of the mission.

DC1.1. The mission must utilize the Deep Space Network (DSN) for any communications beyond earth orbit. (→S/C-R5, S/C-R6), (↓2.3)
Rationale: The DSN is a proven resource for ground communication with spacecraft operating beyond earth orbit. The capabilities that it provides were created at great expense and funding will not be provided to duplicate them.
…

Figure 12. Sample Customer-Derived Design Constraints for OPE.

Customer programmatic constraints are those programmatic decisions that will influence the design of the entire system. Conflicts between safety and customer programmatic constraints should be investigated, so it is critical to document programmatic constraints in the intent specification. These constraints are documented in Level 0 of the intent specification. An example of this type of constraint is provided in Figure 13.

PC1. Whenever the mission utilizes space exploration infrastructure that other space exploration missions make use of, it must do so without directly interfering with the successful completion of those missions. (↑ACC1, ACC2, ACC7), (→S/C-C3)
Rationale: It is possible for this mission to interfere with the completion of other missions through denying the other mission access to the space exploration infrastructure (e.g., over-use of limited DSN resources, damage to launch pad during this mission results another mission missing its launch window, etc.)
…

Figure 13. Customer-Derived Programmatic Constraint.

Step 6: Perform High-Level Functional Decomposition

After the goals and external constraints are defined and documented, the next step in the methodology is to perform a high-level functional decomposition to define the system functions and assign those functions to high-level system components.

The assignment of functions to components can be viewed as system-level design decisions and should involve an analysis to assist in making safety-related decisions. These choices have a huge impact on safety decisions. For example, if, for business reasons, management decides to use radio-isotopic thermoelectric generators (RTGs), this decision will potentially introduce a hazard of contaminating Earth by inadequately preventing the dispersion of radioactive materials into Earth's atmosphere. In order to incorporate safety from the beginning of system development we recommend following a risk-based architectural approach as described in [10]. While a functional analysis can be performed is various ways, our example uses several Design Structure Matrices (DSMs)[5] to document and aid in the analysis [11]. As functions necessary to meet the system's requirements and constraints are identified, physical and informational interactions between these functions (e.g., energy exchanges, information exchanges, and/or material exchanges) are also identified and recorded in the DSMs. Figure 14 below shows a portion of the DSM used in the initial functional decomposition of the spacecraft for the Outer Planet Explorer mission.

The rows and columns of the matrix represent the functions to be performed. Each function is assigned a number that appears in the row next to the function name, the column corresponding to the function, and the diagonal where the function's row and column intersect. The "1's" in the rows of the matrix represent physical and informational interactions between the functions represented by the row and columns. A "1" in a row indicates that the function represented by that row receives a physical or informational input from the corresponding column function (e.g., the spacecraft translation function in Row 10 requires an order to execute a MOC Directive, electrical power, and a reorientation or pointing of the spacecraft so that the appropriate thrust vector can be obtained). Note that all physical interactions were considered equal in this analysis; it is possible to weight the interactions based on a number of criteria (e.g., scale, complexity, etc.), however, for the purpose of this analysis, such weighting was deemed beyond the scope of this study and left to future work.

Physical and informational coupling across functional components of complex systems has often been cited as a major factor in the accidents that occur in these systems [12]. Therefore, once all functions and physical interactions identifiable at this point of the analysis are recorded in the DSM, the functions are clustered by manually reordering the

[5] Design Structure Matrices are also referred to as N-Square Diagrams, Dependency Structure Matrices, Incidence Matrices, Dependency Maps, Interaction Matrices, Design Precedence Matrices, etc. in the literature.

matrix rows and columns in order to assign the functions to individual functional elements in a manner that minimized coupling across the functional elements. The major functional elements defined in the DSM of Figure 14 are denoted by distinct colors that are explained in the key for the matrix. Once these functional elements where identified, requirements and constraints are derived for them. The process for identifying lower-level functional elements necessary to satisfy these new requirements and constraints is described later in step 9.6.

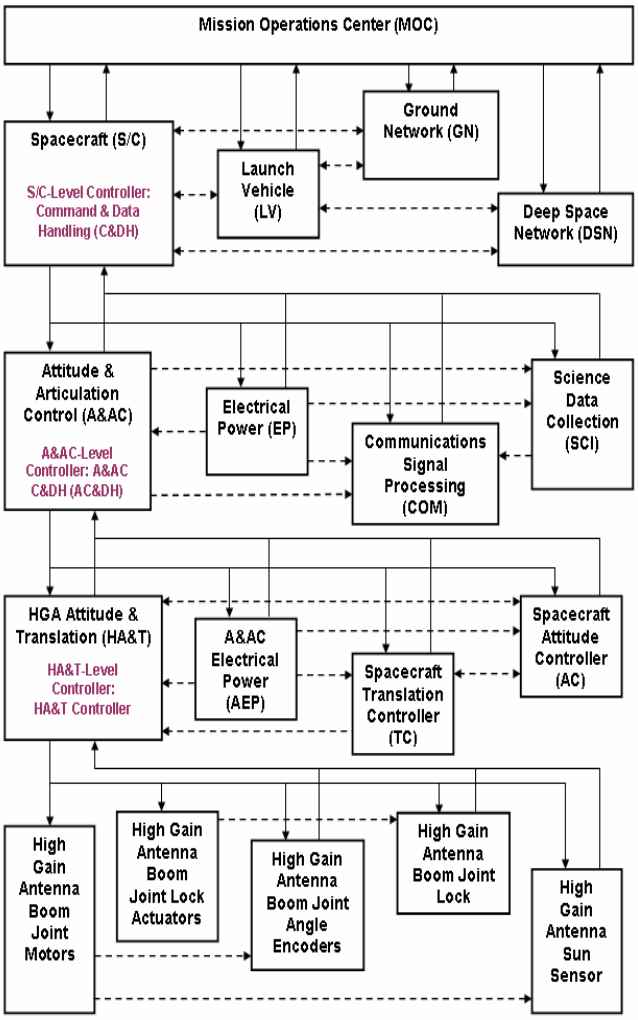
Name		1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
Spacecraft MOC Directive Execution	1	1	1	1						1						
Spacecraft MOC Directive Storage	2		2							1						
Spacecraft H&S Data Evaluation	3			3	1		1			1						
Spacecraft H&S Data Collection	4	1			4					1						
Spacecraft H&S Data Storage	5	1		1	1	5				1						
Spacecraft H&S Data Retrieval from Storage	6	1				1	6			1						
Spacecraft H&S Data Packetization	7	1					1	7		1						
Spacecraft Power Generation	8	1							8							
Spacecraft Power Distribution	9	1							1	9						
Spacecraft Translation	10	1								1	10	1				
Spacecraft Pointing	11	1								1	1	11	1	1		
Spacecraft Antenna Deployment	12	1								1			12			
Spacecraft Antenna Pointing	13	1								1	1	1	1	13		
Spacecraft Science Data Collection	14	1								1	1	1			14	
...																

Key
Command & Data Handling (C&DH)
Electrical Power (EP)
Attitude & Articulation Control (A&AC)
Science Data Collection (SCI)

Figure 14. A Portion of the Design Structure Matrix.

Step 7: Design High-Level System Control Structure

As can be seen in Figure 14, it is not always possible to define functional elements of a system such that no physical and informational interactions across elements are necessary. To address this issue, we turn to a hierarchical control structure [13]. At each level of functional decomposition, each functional element is assigned responsibility for the control of the functional interactions within the element while one hierarchically superior element is assigned responsibility for control of the interactions across elements. Note that this is not a description of the architecture, but a representation of the functions the system must perform and how the functions are related to each other. In the example, interactions between spacecraft functional elements are controlled by the spacecraft command and data handling functional element (C&DH) while interactions between functional elements of the Attitude and Articulation Control (A&AC) functional element are controlled by the A&AC command and data handling functional element (AC&DH). The system control structure is provided in Figure 15 below. The iterative evolution of the control structure is discussed later in step 9.6.



Control Structure Legend	
Diagram Item:	Description:
↓	Control in the form of Directive(s) or Command(s)
↑	Control Feedback in the form of State Information or Sensor Measurements
⇢ ⇠ ⇢	Physical and Informational Interaction other than Control and Control Feedback Interactions
Functional Element Name (Functional Element-Level Controller (if applicable))	Functional Element with the controller of its internal interactions (i.e., the functional element-level interactions)

Figure 15. Outer Planets Explorer Control Structure.

Step 8: Perform Preliminary Hazard Analysis

Using the information obtained in steps 2 to 5, a preliminary

hazard analysis using STPA is performed and the system hazard log is created.

STPA is performed as described in the background section of this paper. The results of STPA are recorded in the hazard analysis section. It is worth reiterating that in the initial phase, not much of the design information is known and that the STPA process will focus on safety constraint refinement and architecture selection. Later in the process STPA may involve more design refinement to enforce the safety constraints. A partial result of an STPA hazard analysis for OPE is shown below in Figure 16.

Inadequate Control Actions and Control Flaws	Relevant High-Level Hazard(s)	Associated Design Principle	Resulting Safety Constraint(s) or Requirements
A&AC-ICA1. The HGA rotates relative to the spacecraft at the wrong time. (←S/C-SC1, S/C-SC2, S/C-SC5, S/C-SC7)	H1, H2, H5, H6, H7	A&AC-2.1	
A&AC-CF1.1. The HGA rotates relative to the spacecraft while the spacecraft is within the payload fairing. *Rationale: Because space is constrained within the payload fairing, articulation of the HGA relative to the spacecraft could cause damage to the HGA, launch vehicle, and/or other parts of the spacecraft. The effects of such damage could range from degradation in HGA functionality to partial break up of the spacecraft and launch vehicle.*			A&AC-SC1

Figure 16. Partial STPA Hazard Analysis.

Creation of the hazard log follows STPA. For each high-level hazard, the system element pertaining to the hazard is listed as well as the relevant operation or mission phase. The causal factors shown in the hazard log are pointers to the control flaws identified in the STPA hazard analysis. The level and effect is information pertaining to hazard severity and the categorization of the resulting loss, if desired. A portion of the OPE hazard log is shown below in Figure 17. The hazard log and hazard analysis are documented in Level 1 of the intent specification.

H1. Inability of Mission to collect data. (↓SV-85)

System Element: Spacecraft (C&DH, SCI, EP, and A&AC), Launch Vehicle, and Mission Operations Center

Operation/Phase: Pre-Launch, Post-Launch/Pre-Icy Moon Orbit, Icy Moon Orbit, Disposal

Causal Factors: Spacecraft loses functionality to collect data, (←C&DH-CF1.1, C&DH-CF2.1, C&DH-CF8.1, C&DH-CF9.1, C&DH-CF10.1, A&AC-CF1.1, A&AC-CF2.1, A&AC-CF2.2, A&AC-CF2.3, A&AC-CF3.1, A&AC-CF4.1, A&AC-CF4.2, A&AC-CF4.3, A&AC-CF10.1, A&AC-CF10.2, A&AC-CF10.3, A&AC-CF10.4, A&AC-CF10.5, A&AC-CF10.6, A&AC-CF10.7, A&AC-CF10.8, A&AC-CF11.1)

Level and Effect: Potential loss of data collection opportunities (↑ACC4)

Safety Constraints:
The mission must have the necessary functionality for data acquisition at the required times. (→SC1)

The spacecraft must have the necessary functionality for data acquisition at the required times. (→S/C-SC1)
…

Figure 17. Partial Hazard Log for OPE.

Step 9: Define System Element Specifications

Steps 9.1 through 9.6 are used to

1. Define goals, assumptions, requirements, design constraints, and safety constraints for each subsystem or functional element at Level 1.

2. Make design decisions at Level 2 to implement the requirements and constraints. Refer to Figure 21 for an example.

3. Create a formal model of the control system design at Level 3. Refer to Figure 23 for an example.

Steps 9.1 through 9.6 are performed iteratively until the design is set.

Step 9.1: Define Subsystem or Functional Element Goals, Requirements, and Constraints

Once the mission-level goals, requirements, design constraints, and safety constraints are defined, it is then possible to allocate them to the subsystem or functional elements identified in the functional decomposition through the definition of goals, requirements, and constraints (both safety-related and non-safety-related) for these functional elements or subsystems. Refer to Figures 18 and 19 for an example of functional elements goals, requirements, and constraints. The subsystem and/or functional element goals,

requirements, and constraints are documented in Level 1 of the intent specification.

Spacecraft Attitude and Articulation Control (A&AC) Design Constraints

A&AC-C1. All attitude and articulation control components must fit within TBD% of the space beneath the payload fairing of a Delta-IVH. (←S/C-C2), (→AC&DH-C1), (↓A&AC-2.2.1)
Rationale: Space is a limited resource inside the payload fairing of a launch vehicle and thus, space for each component of the spacecraft must be carefully budgeted. The space allocation process involves a number of architectural tradeoffs beyond the scope of this study. Therefore, we will use a TBD% space allocation to the attitude and articulation control functional elements.

A&AC-C2. The A&AC functional element must be able to receive and execute directives from the C&DH functional element. (←A&AC-G2), (→AC&DH-G2), (↓C&DH-2.1.6)

Spacecraft Attitude and Articulation Control (A&AC) Safety-Related Design Constraints

A&AC-SC1. The HGA must not rotate or translate with respect to the main spacecraft structure while the spacecraft is inside the payload fairing of the launch vehicle. (←H1, H2, H5, H6, H7), (→A&AC-ICA10, A&AC-ICA11, AC&DH-G1, HA&T-G1, HA&T-G2, HA&T-G3), (↓A&AC-2.2.1.4, A&AC-2.2.1.8, A&AC-2.3, A&AC-2.4)

A&AC-SC2. HGA translation and rotation with respect to the main spacecraft structure must be restrained during periods of spacecraft velocity changes so that the HGA does not deform and/or detach from the spacecraft due to inertial loads. (←H1, H2, H4, H5, H6, H7), (→AC&DH-G1, HA&T-G1, HA&T-G2, HA&T-G3), (↓A&AC-2.2.1.4, A&AC-2.3, A&AC-2.4)

A&AC-SC3. While translating and/or rotating, the HGA and the radiation it emits must maintain minimum separation from other parts of the spacecraft. (←H1, H2, H4, H5, H6, H7), (→A&AC-ICA10, A&AC-ICA11, AC&DH-G1, HA&T-G1, HA&T-G2), (↓A&AC-2.2.1.5, A&AC-2.3, A&AC-2.4, AC&DH-2.1.2, A&AC-2.2.1.9)
…

Figure 18. Example functional element constraints for OPE.

Spacecraft Attitude and Articulation Control (A&AC) Goals

A&AC-G1. To provide the spacecraft velocity changes necessary for orbit insertion about the icy moon of the outer planet, maintenance of/changes to that orbit, and spacecraft disposal. (←S/C-R1), (→A&AC-R1, A&AC-R2, A&AC-R3, A&AC-R4, A&AC-R5), (↓2.2, S/C-2.3, C&DH-2.1.6, SV-1, SV-2)

A&AC-G2. To point the spacecraft and spacecraft elements in accordance with science data and communications needs for the mission. (→A&AC-R6, A&AC-R7, A&AC-R8, A&AC-R9, A&AC-R10), (↓S/C-2.3, C&DH-2.1.6, SV-1, SV-2)
Rationale: The pointing of the spacecraft and spacecraft elements are both allocated to the A&AC functional element because the rotation and translation of spacecraft elements with respect to the main spacecraft structure will affect spacecraft attitude.

Spacecraft Attitude and Articulation Control (A&AC) Requirements

A&AC-R1. After release from the launch vehicle, the A&AC shall provide spacecraft velocity changes for spacecraft transit from the release point to the orbit of the outer planet. (←A&AC-G1), (↓A&AC-2.6, SV-1, SV-2)

A&AC-R2. Upon arrival to the orbit of the outer planet, the A&AC shall provide spacecraft velocity changes necessary for spacecraft capture in the orbit of the outer planet. (←A&AC-G1), (↓A&AC-2.6, SV-1)
…

Figure 19. Example functional element goals and requirements for OPE.

Step 9.2: Develop Models of the System under Control

In this step, the system engineer performs state-based behavioral modeling (per the State Analysis description, above) of the system under control. Starting from the high-level hazards, constraints, and requirements, engineers draw state effects diagrams and develop the state variable, measurement, and command models. These models are a good repository for design information from subsystem engineers, including continuous physics, such as the pointing dynamics of the spacecraft's high gain antenna. The state-based behavioral model artifacts are documented on Level 2 of the intent specification. As described in Section 2 of this paper, State Analysis is used to model the system under control and aides in the design of the control system (estimators and controllers), which is described formally in Level 3 of the intent specification.

It is important that the models of the state variables, etc. are captured concurrently with the drawing of the state effects diagram. Drawing the state effects diagram alone shows all

of the state variables and existence of physical effects between them, but without models that explicitly describe the nature of the physical effects, the state effects diagram is incomplete.

Also, while performing the state analysis modeling, assumptions and rationale are documented and the level of detail to which the subsystem is modeled is under the purview of the system engineer. For example, if unmitigated hazards are traced to the imaging system, then that subsystem will be modeled in greater detail than subsystems that are not associated with high-level hazards.

Below, in Figure 20, is an example of the state effects diagram pertaining to the control of the HGA boom rotation joint angles. To control the angle of Joint i along degree of freedom j, commands are sent to the appropriate motor on Joint i. The angle measurement is used by the control system to send commands to the motors on joint i to control the angle. The lock position also has an effect on the joint motor (i.e., the joint motor is unable to operate nominally with the motor lock engaged).

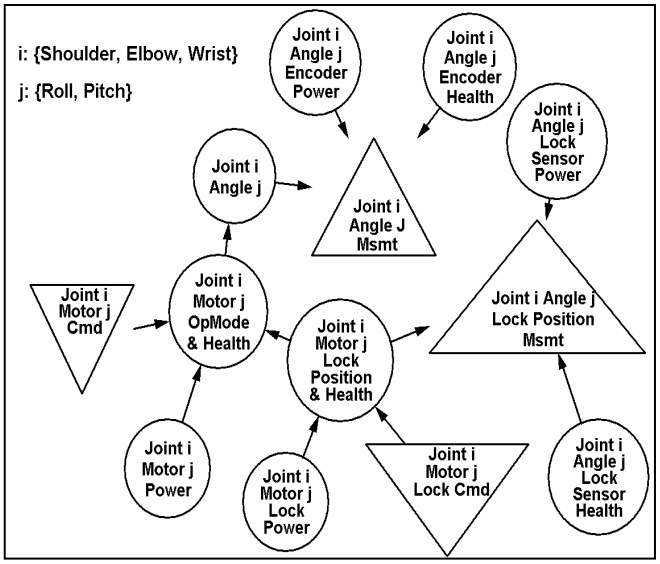


Figure 20. State effects diagram for HGA boom joint rotation on OPE.

Step 9.3: Define and Design Control System Operational Behavior

In this step, the system engineer documents design decisions pertaining to the control system. This information serves as the textual description of the control system and helps in the creation of the blackbox models. The state effects diagram and models provide key insight into physical effects in the state variables of the control system and how to design the control system behavior appropriately.

For example, the design specification of how directives for HGA Attitude and Translation (HA&T) Control are created from C&DH directives is shown below in Figure 21.

AC&DH-2.1.1. The AC&DH derives and sends the 'HGA Boom Rotation Joint Targets' directive to the HA&T. (↑AC&DH-R4, AC&DH-R5, AC&DH-C3), (→AC&DH-2.1.2)

AC&DH-2.1.2. The AC&DH defines an imaginary 3-dimensional surface referred to as the "roll-pitch mask" that marks the allowable boundary of HGA motion and signal radiation relative to the main spacecraft structure and spacecraft appendages. Any C&DH directive that produces an 'HGA Boom Rotation Joint Targets' directive that would lead to a combination of shoulder, elbow, and wrist roll and pitch rotation that causes any part of the HGA, HGA boom, and/or the HGA's radiated signals to breach the roll-pitch mask will not be authorized for execution by the AC&DH. (↑A&AC-SC3, A&AC-SC16, AC&DH-C3), (←A&AC-2.1, A&AC-2.2, AC&DH-2.1.1), (→SV-8, SV-9, SV-12, SV-13, SV-16, SV-17)

AC&DH-2.1.2.1. The roll-pitch mask is derived from the mounting position of the HGA Boom Shoulder Joint on the main spacecraft structure and state information inputs from the controllers of other spacecraft appendage articulation.

…

Figure 21. OPE example of design decisions enforcing Level 1 constraints while implementing goals and requirements.

Step 9.4: Develop Formal Models of the Control System

In this step, the system engineer designs and specifies blackbox models of the control system using SpecTRM-RL. Discrete state variable models documented in Level 2 can often be represented in SpecTRM-RL directly. Many state analysis state variable models describe continuous phenomena, in which case engineers may either discretize the state variables or use SpecTRM *functions* and *macros* to compute state variable values.

State effects models are representations of the physics in the system under control. SpecTRM-RL models, on the other hand, are discrete representations of the control system and the control system's model of the system under control that together specify the system blackbox behavior. Succinctly, the state effects diagram and models describe the behavior of the system under control while SpecTRM-RL models describe the behavior of the control system.

Consequently, in the SpecTRM-RL model of the control system, system state variables will always have an "Unknown" state. If, for example, a measurement was not received the controller may not be able to infer the current value of a state variable and the controller's model of that state variable would transition to "Unknown." This manner of accounting for insufficient control system knowledge of a

system state is consistent with the state-based software design aspect of State Analysis, where the control system specifications are required to consider uncertainty in the state estimates.

Engineers use Level 2 high-level control system design decisions and the control structure to populate the blackbox model. Figure 22 shows part of the formal control system definition, namely the definition of the input, WristRollLockPositionMSMT. This input can be seen in the control system design shown in Figure 23 (note that the "Joint i" notation is used to compactly refer to the wrist, elbow, and shoulder joints).

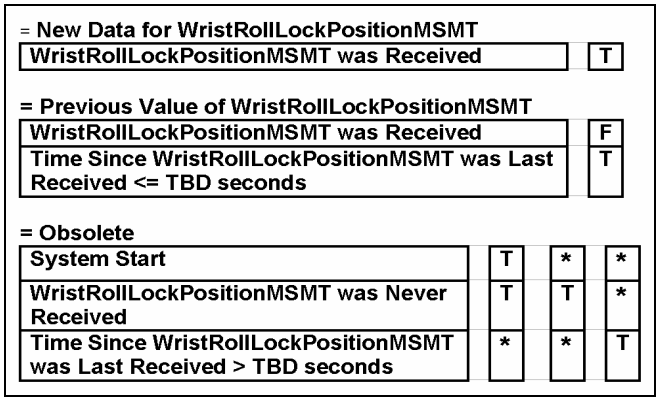


Figure 22. Formal model of the wrist roll lock sensor input to the control system

Engineers also use Level 1 requirements and constraints to populate the blackbox model. For instance, there will often be safety constraints listed in Level 1 for the maximum time allowed between safety-critical measurement readings. This information is used to transition a measurement from valid to obsolete in the SpecTRM-RL process model. Depending on the state variable model, obsolescence of a measurement may cause the transition of a state variable from some value to "Unknown".

Engineers create the process models used by the controllers in SpecTRM-RL through a rigorous mapping process:

1. The hierarchy of control in Level 3 should reflect the control structure listed in Level 2.
2. Control system behavior should reflect designed high-level control behavior described in Level 2.
3. Control system design must enforce constraints and requirements listed in Level 1.
4. Control system design must reflect state analysis artifacts as described below.

The process model for the control of the high gain antenna is shown below in Figure 23. Referring back to the state effects diagram in Figure 20, we see that the process model design is derived from the state effects model from the state analysis. Each affecting state variable that can be classified as a process input to the controlled state variables must be included in the model as well as every affecting state variable for the measurements evaluated by the controller (such as the WristRollLockPositionMSMT, which is labeled "Joint i Angle j Lock Position Msmt" in the state effects diagram). This process model is written in SpecTRM-RL and can be used for traceability, completeness analysis, and automatic code generation.

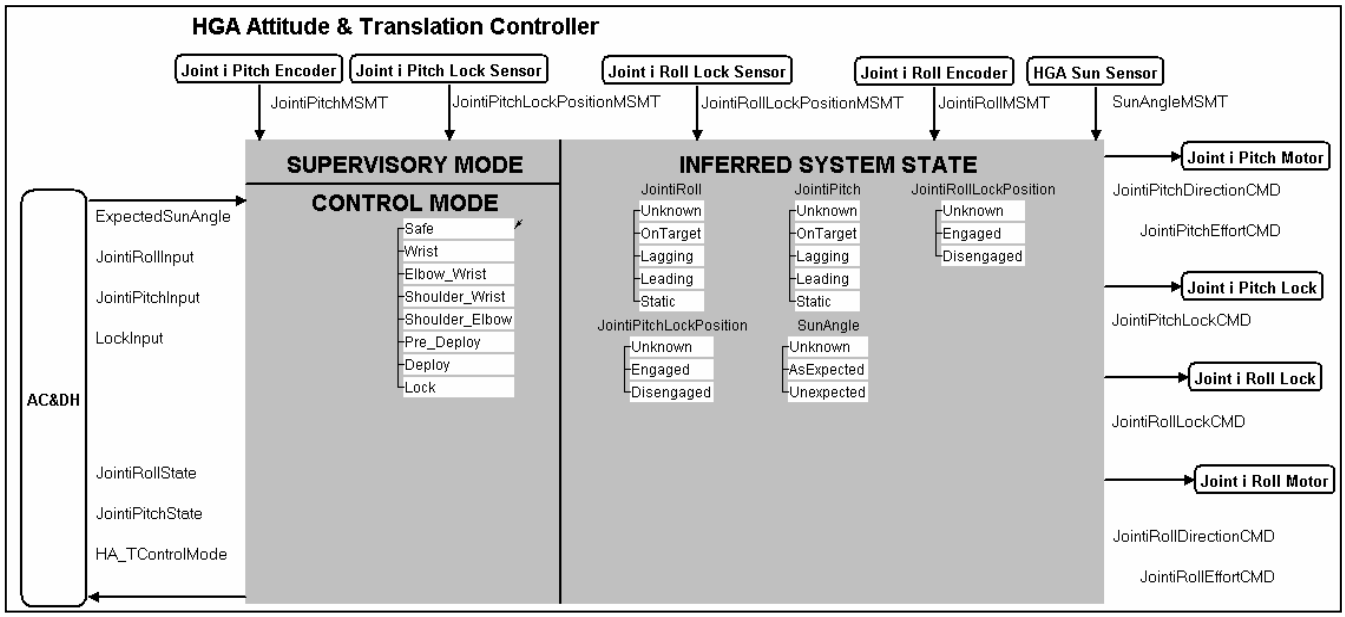


Figure 23. SpecTRM process model of High Gain Antenna Pointing.

Step 9.5: Continue to Perform STPA

STPA is performed in parallel with the creation of design. As new design is created or new control flaws are found, the state analysis products are augmented and used for further hazard analysis. After new design has been captured in the state effects diagram, engineering judgment is used to see if

other important state variables need to be added.

Many inadequate control actions stem from poor controller behavior due to the controller's incorrect process model. The state effects diagram should aid in creating the process model and documenting the actual physics of the system. State analysis can also be used in the discovery of control flaws—state effects that could lead to instances of inadequate control.

The STPA process may also lead the engineer to reevaluate state variable models and discover new state effects. These additional state effects and modified state variable models are recorded in an updated state effects diagram. In addition, the STPA process may inspire changes to the control system and the control system's process model. In such cases, the SpecTRM-RL models should be modified as well.

New design is also created during the STPA process to enforce newly defined constraints or better enforce existing constraints. The new design must also be reflected in the state analysis artifacts.

Step 9.6: Iteratively Refine the System Design

Two products completed at a high level in steps 1 to 8 are iterated on to inform lower-level design: the functional decomposition and the control structure. The identification and decomposition of lower-level functional elements must be performed in tandem with steps 9.1 through 9.5. As new or lower-level requirements and constraints are generated, new and lower-level functions within the elements will be identified in the DSM. Accordingly, the functional elements will each be decomposed with another DSM. Additionally, as steps 9.1 through 9.5 are performed, the control structure must be fleshed out iteratively to capture lower-level interactions and inform the lower-level design.

Repetition of steps 9.1 through 9.6 may also change products from other steps in the methodology. Feedback to the earlier steps of the methodology can occur when engineers create new requirements and constraints as a result of STPA or if the hazard analysis inspires engineers to make Level 2 design changes or Level 1 goal changes.

Iteration through the methodology, both through steps 1 to 8 and 9.1 to 9.6, is complete when the design is set and all hazards are eliminated, mitigated, or controlled.

Step 10: Perform Validation Tests

While only steps 1 to 9 were used in the study described in this paper, validation is an important part of the methodology. SpecTRM has several tools for validation. SpecTRM-RL models are executable and analyses can be performed on them (completeness, robustness, and consistency) to evaluate and identify errors and omissions.

Step 11: Generate Designs and Software Code

The final design of the system under control and the implementation of the control system in software code are the ultimate end-products of the methodology. Physical component designs and software code are generated from the models, either manually or automatically.

4. OPE INTENT SPECIFICATION TRACEABILITY

Traceability between systems engineering artifacts is becoming increasingly important in the design of spacecraft. As stated in the NASA Software Safety Standard [14]:

> *"Because many software safety requirements are derived from hazard analysis, these requirements will also be linked to specific hazard reports…Tracing requirements is a vital part of system verification and validation, and especially in safety verifications. Full requirements test coverage is virtually impossible without some form of requirements traceability. Tracing also provides a way to understand the impact on the system of changing requirements or modification of software elements."*

Given the importance of traceability, it worth noting how traceability is captured in the products generated from the methodology used in this study. Figure 24 below shows traceability between all Level 0 through Level 2 artifacts of the OPE Intent Specification. In this figure, a link between "Customer Programmatic Constraints" and "Component/Functional Element Constraints," for example, indicates that at least one Component/Functional Element is hyperlinked to at least one Customer Programmatic Constraint in the OPE Intent Specification. These links, as mentioned in the Intent Specification background section, represent "means-ends" relationships between linked items. These types of relationships are important to consider in the derivation and modification of specification items. As can be seen in the figure, the methodology in this study led to traceability (and the documentation thereof) of Design Decisions to High-Level Hazards and High-Level Hazard Causal Factors, among other things, through High-Level and Component/Functional Element Safety Constraints. Note the closed loop between Component/Functional Element Safety Constraints, Design Decisions, Inadequate Control Actions, Control Flaws, and High-Level Hazard Causal Factors; this loop represents the concurrent hazard analysis and design work occurring in step 9.5. Moreover, traceability was also captured between design decisions and many other traditional systems engineering artifacts, as well as those that were previously unique to STAMP and STPA, Intent Specification, or State Analysis.

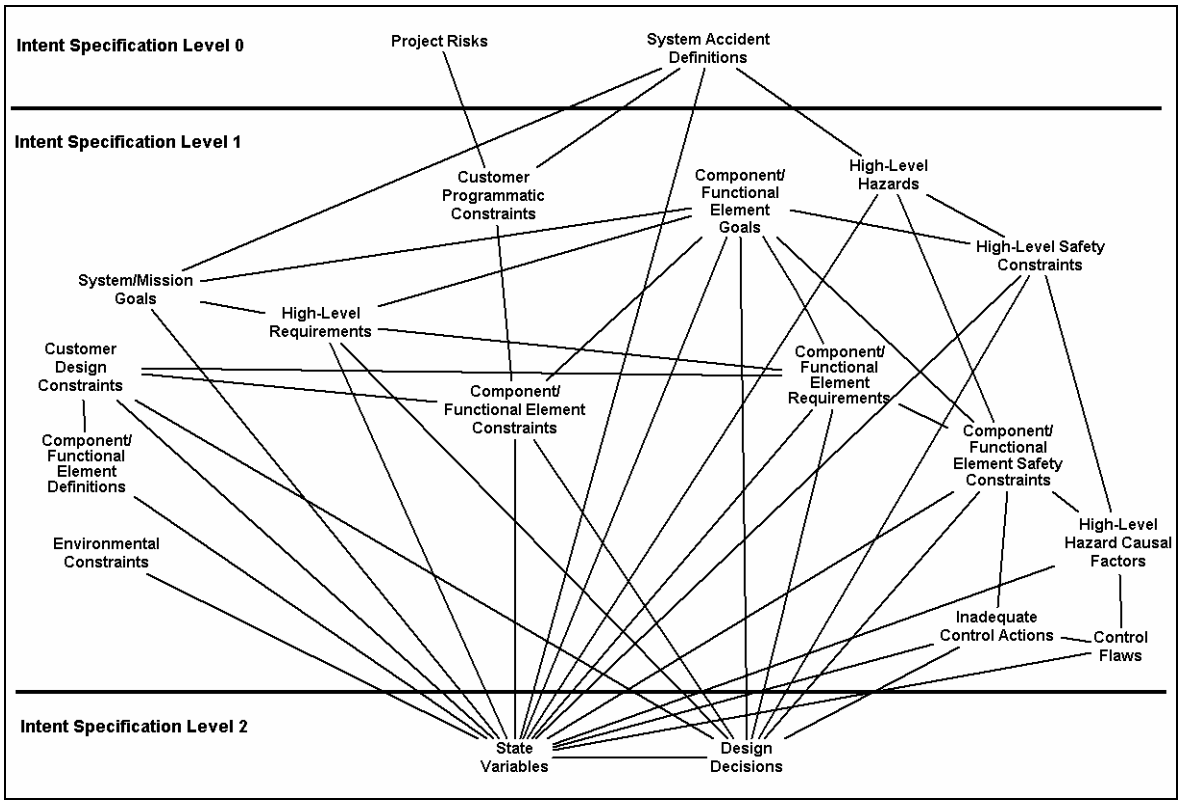


Figure 24. The traceability between systems engineering artifacts in the OPE Intent Specification

5. OPE HGA BOOM TRADE STUDY

Designers typically have many design options for enforcing safety-related constraints and the options that they ultimately choose greatly affect the cost and efficacy of constraint enforcement. In this section, we present a brief, comparative analysis of design options considered for the enforcement of specific safety-related constraints in the OPE Intent Specification.

The basic spacecraft high gain antenna boom configurations (i.e., stowed and deployed) that were derived in this study using the methodology described above are shown in Figure 25. The Level 2 design decision describing this configuration is provided in Figure 26 (refer to Appendix B for a derivation of this design decision).

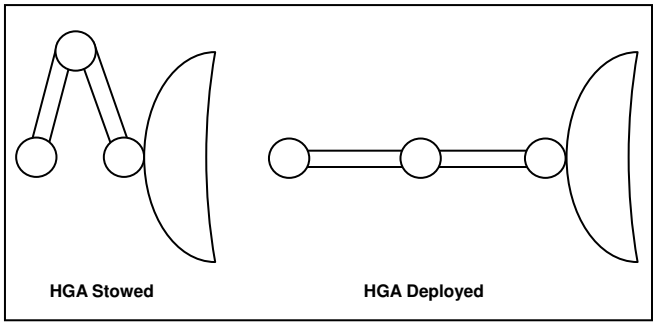


Figure 25. Basic High Gain Antenna (HGA) boom configuration for OPE.

A&AC-2.2.1. The HGA boom consists of 3 rotation joints (shoulder, elbow, and wrist) driven by TBD electric motors and connected by 2 solid (i.e., non-telescoping) boom segments. (↑A&AC-A1, A&AC-A2, A&AC-C1, HA&T-G2), (←A&AC-2.1.1)

Figure 26. Level 2 design decision for the basic HGA boom configuration.

However, further specification of the rotation joints is necessary to evaluate the efficacy of the enforcement of three safety constraints of interest in HGA pointing, shown in Figure 27. The design variables of interest are: the rotational degrees of freedom in each joint, the presence and capability of mechanical locks on each degree of freedom, and the type of actuator used for each degree of freedom. Accordingly, the tradespace that was considered in this study is described in Figure 28 below. All design options include a pitch degree of freedom in each joint and a roll degree of freedom in the wrist joint. Pitch is required in each joint for boom deployment and roll is required in the wrist joint in order to create a hemispherical range of HGA motion. In each design, wrist roll and pitch are the primary degrees of freedom used for HGA articulation after deployment and thus, these degrees of freedom are actuated by motors. In some options, redundant pointing capabilities are provided by pitch and roll degrees of freedom in the elbow and shoulder joints and thus these degrees of freedom are also actuated by motors. In Option 5, no redundancy is provided as the pitch degrees of freedom of the elbow and shoulder joints are only used for deployment and are

actuated by damped springs. Finally, while all options include locks on all of the available degrees of freedom to restrict rotation while the HGA is under the payload fairing, only two include locks that can be re-engaged after HGA boom deployment.

A&AC-SC1. The HGA must not rotate or translate with respect to the main spacecraft structure while the spacecraft is inside the payload fairing of the launch vehicle. (←H1, H2, H5, H6, H7), (→A&AC-ICA10, A&AC-ICA11, AC&DH-G1, HA&T-G1, HA&T-G2, HA&T-G3), (↓A&AC-2.2.1.4, A&AC-2.2.1.8, A&AC-2.3, A&AC-2.4)

A&AC-SC3. While translating and/or rotating, the HGA and the radiation it emits must maintain minimum separation from other parts of the spacecraft. (←H1, H2, H4, H5, H6, H7), (→A&AC-ICA10, A&AC-ICA11, AC&DH-G1, HA&T-G1, HA&T-G2), (↓A&AC-2.2.1.5, A&AC-2.3, A&AC-2.4, AC&DH-2.1.2, A&AC-2.2.1.9)

A&AC-SC7. Mechanical disturbances to the HGA must not create rotational or translational oscillations of the HGA that are large enough to prevent data return. (←H1, H2, H4, H5, H6, H7), (→A&AC-ICA10, AC&DH-G1, AC&DH-C4, HA&T-G1, HA&T-G2, HA&T-G3), (↓A&AC-2.3, A&AC-2.4, SV-3, SV-5, SV-6, SV-10, SV-14)

Figure 27. Safety-Related Design Constraints associated with HGA articulation.

In applying STPA to the two design options on the extremes of the tradespace (i.e., options 1 and 5), it is apparent that they produce similar inadequate control actions and control flaws (see Figure 29 for the identified inadequate control actions for these options). However, the differences in the control flaws have important implications in the enforcement of the safety constraints generated from them. For example, preventing an interruption in power to a joint motor for some interval between HGA Boom deployment and the end of the mission (i.e., a safety constraint necessary for Design Option 5) can be a much more difficult problem than engaging a joint rotation lock in the event of such a power interruption (i.e., the corresponding safety constraint for Design Option 1). Additionally, the use of springs in the shoulder and elbow pitch degrees of freedom in Design Option 5 introduces a timing constraint on the lock disengagement of these two degrees of freedom (i.e., if one is released and the other is significantly late in its release, the HGA and/or boom might contact another part of the spacecraft). In other words, the design options affect which inadequate control actions will be of most relevance throughout the mission. Because Design Option 1, more than Design Option 5, leads to increased relevance of an inadequate control action that leads only to data loss (i.e., A&AC-ICA9 in Figure 29) and decreased relevance of inadequate control actions leading to data loss, mission failure, and orbital debris generation (i.e., A&AC-ICA10 and A&AC-ICA11 in Figure 29), we selected Design Option 1 for further elaboration in the OPE Intent Specification.

Design Option	Shoulder Roll	Shoulder Pitch	Elbow Roll	Elbow Pitch	Wrist Roll	Wrist Pitch	Rotation Locks	Rotation Lock Sensors
Option 1	Available as 2nd Backup to Wrist Roll (motorized)	Used for deployment and available as 2nd Backup to Wrist Pitch (motorized)	Available as 1st Backup to Wrist Roll (motorized)	Used for deployment and available as 1st Backup to Wrist Pitch (motorized)	Primary Roll DOF for HGA Pointing (motorized)	Primary Pitch DOF for HGA Pointing (motorized)	Can be selectively engaged/disengaged throughout mission	Six
Option 2	Available as 2nd Backup to Wrist Roll (motorized)	Used for deployment and available as 2nd Backup to Wrist Pitch (motorized)	Available as 1st Backup to Wrist Roll (motorized)	Used for deployment and available as 1st Backup to Wrist Pitch (motorized)	Primary Roll DOF for HGA Pointing (motorized)	Primary Pitch DOF for HGA Pointing (motorized)	Engaged for pre-deployment phases and permanently disengaged at deployment‡	Six
Option 3	Not Available	Used for deployment and available as 2nd Backup to Wrist Pitch (motorized)	Not Available	Used for deployment and available as 1st Backup to Wrist Pitch (motorized)	Primary Roll DOF for HGA Pointing (motorized)	Primary Pitch DOF for HGA Pointing (motorized)	Can be selectively engaged/disengaged throughout mission	Four
Option 4	Not Available	Used for deployment and available as 2nd Backup to Wrist Pitch (motorized)	Not Available	Used for deployment and available as 1st Backup to Wrist Pitch (motorized)	Primary Roll DOF for HGA Pointing (motorized)	Primary Pitch DOF for HGA Pointing (motorized)	Engaged for pre-deployment phases and permanently disengaged at deployment	Four
Option 5	Not Available	Used for deployment only (spring-loaded)	Not Available	Used for deployment only (spring-loaded)	Primary Roll DOF for HGA Pointing (motorized)	Primary Pitch DOF for HGA Pointing (motorized)	Engaged for pre-deployment phases and permanently disengaged at deployment	Four

Figure 28. Design options considered in this study for HGA boom joint rotation.

A&AC-ICA9. The HGA boom rotation joints do not rotate along the appropriate degrees of freedom at the required times. (→S/C-SC2), (↓A&AC-2.2.1.4, A&AC-2.2.1.8, SV-8, SV-9, SV-12, SV-13, SV-16, SV-17)

A&AC-ICA10. The HGA Boom Rotation Joints rotate too little, too slowly, or too much along the HGA Boom Rotation Joint degrees of freedom when the locks are disengaged. (→A&AC-SC3, A&AC-SC7), (↓A&AC-2.2.1.5, A&AC-2.2.1.8, SV-8, SV-9, SV-12, SV-13, SV-16, SV-17, SV-37, SV-38, SV-39, SV-40, SV-41, SV-42)

A&AC-ICA11. The locks on the degrees of freedom in the joints of the HGA boom disengage (i.e., allow rotation) at the wrong time. (→A&AC-SC1, A&AC-SC3), (↓A&AC-2.2.1.8, SV-8, SV-9, SV-12, SV-13, SV-16, SV-17, SV-37, SV-38, SV-39, SV-40, SV-41, SV-42)

Figure 29. Inadequate Control Actions associated with HGA boom design options 1 and 5.

6. Conclusions

The methodology demonstrated in the application described in this paper shows promise for addressing the systems engineering and system safety challenges presented by increasingly complex and ambitious space exploration missions. In synthesizing four state-of-the-art systems engineering frameworks, the methodology provides a more seamless approach for evaluating safety-related concerns in every step of the design process. In the Outer Planets Explorer mission application described in this paper, multiple design options for pointing of a spacecraft antenna were rigorously defined from high-level system goals and constraints and then evaluated for their potential to cause unacceptable losses or accidents during and after the mission. The design option ultimately selected reduced the potential for antenna/spacecraft structure collisions that could lead to data loss, mission failure, and orbital debris generation. Other design options that would increase the potential for these collisions were identified early and rejected. Furthermore, the traceability between many traditional systems engineering artifacts of the design process as well as those artifacts previously unique to STAMP and STPA, Intent Specification, and State Analysis was documented. Such documentation of traceability is inherent to the methodology and can be utilized to assist systems engineers during spacecraft verification and validation and in dealing with the many design changes that inevitably occur in the spacecraft development process.

Future research could lead to further development of the functional decomposition and safety-related trade study techniques discussed in this paper. Additionally, tools can be created to assist in the development of spacecraft using the safety-driven design methodology.

Acknowledgements

The authors would like to thank Brad Burt, Karla Clark, Bharat Chudasama, Jeffrey Estefan, Cecilia Guiar, Peter Kahn, Steven Larsen, Cin-Young Lee, Robert Lock, Kenneth Meyer, David Nichols, Robert Rasmussen, Henry Stone, Robert Vargo, and Stephen Wall at the Jet Propulsion Laboratory for their support of this study. Additionally the authors would like to thank the peer reviewers for their thoughtful suggestions for this paper.

This research was performed at the Massachusetts Institute of Technology (supported through JPL University Subcontract 1297013), and at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] Nancy G. Leveson, "The Role of Software in Spacecraft Accidents," AIAA Journal of Spacecraft and Rockets 41, 564–575, July 2004.

[2] Margaret Stringfellow Herring, A Safety-Driven Model-Based System Engineering Methodology, S.M. Thesis, Aeronautics and Astronautics, Massachusetts Institute of Technology, 2008.

[3] Nancy G. Leveson, "Intent Specifications: An Approach to Building Human-Centered Specifications," IEEE Transactions on Software Engineering 26, 15–35, January 2000.

[4] Nancy G. Leveson, "A New Accident Model for Engineering Safer Systems," Safety Science 42, 237–270, April 2004.

[5] Kathryn A. Weiss, Nicolas Dulac, Stephanie Chiesi, Mirna Daouk, David Zipkin, and Nancy G. Leveson, "Engineering Spacecraft Mission Software using a Model-Based and Safety-Driven Design Methodology," AIAA Journal of Aerospace Computing, Information, and Communication 3, 562–586, November 2006.

[6] Nancy G. Leveson, System Safety Engineering: Back to the Future, http://sunnyday.mit.edu/book2.pdf, Cambridge, MA, 2008.

[7] Nancy G. Leveson, Safeware: System Safety and Computers, Reading, MA: Addison-Wesley, 1995.

[8] Michel D. Ingham, Robert D. Rasmussen, Matthew B. Bennett, and Alex C. Moncada, "Generating Requirements for Complex Embedded Systems using State Analysis," Acta Astronautica 58, 648-661, June 2006.

[9] Karla Clark, "Europa Explorer—An Exceptional Mission Using Existing Technology," 2007 IEEE Aerospace Conference Proceedings, March 3–10, 2007.

[10] Nicolas Dulac and Nancy G. Leveson, "Incorporating Safety in Early System Architecture Trade Studies," International System Safety Conference Proceedings, August 22-26, 2005.

[11] Tyson R. Browning, "Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions," IEEE Transactions on Engineering Management 48, 292-306, August 2001.

[12] Charles Perrow, Normal Accidents: Living with High-Risk Technologies, 1999 edition, Princeton, NJ: Princeton University Press, 1999.

[13] Peter Checkland, Systems Thinking, Systems Practice, Chichester, UK: John Wiley and Sons Ltd., 1981.

[14] National Aeronautics and Space Administration, NASA Software Safety Standard, NASA-STD-8719.13B, July 2004.

BIOGRAPHIES

Brandon D. Owens *is a doctoral candidate in Engineering Systems at the Massachusetts Institute of Technology and a research assistant in MIT's Complex Systems Research Laboratory. He previously was a research assistant at the W. W. Hansen Experimental Physics Laboratory at Stanford University where his responsibilities included mission planning and radiation anomaly investigation for the Gravity Probe B satellite mission. Prior to that, he served as a co-op student for United Space Alliance in Houston, Texas in the departments of Cargo Operations and Flight Control and Environmental Systems. He has a B.S. in Aeronautical and Astronautical Engineering from Purdue University and an M.S. in Aeronautics and Astronautics from Stanford University.*

Margaret Stringfellow Herring *is a doctoral candidate in the Aeronautics and Astronautics Engineering Department at the Massachusetts Institute of Technology and research assistant in MIT's Complex Systems Research Laboratory. She previously worked at MIT's Lincoln Laboratory where she performed research in collision avoidance for UAVs. Prior to that, she was a flight software engineer at JPL where she developed State Analysis frameworks for the automatic control of the next-generation Deep Space Network array. She has a B.S. in Aeronautics and Astronautics and a B.S. in Electrical Engineering from MIT*

Nicolas Dulac *is a post-doctoral associate in the department of Aeronautics and Astronautics at MIT. His current research interests span system safety, system engineering, visualization of complex systems, hazard analysis in socio-technical systems, safety culture, and dynamic risk analysis. Lately, he has branched out from the aerospace world and started applying new safety and risk management techniques to areas such as pharmaceutical development, food safety, process operations, surgery units and corporate fraud. He holds a Ph.D. and M.S. degree in Aeronautics and Astronautics from MIT, and a B.S. degree in Mechanical Engineering from McGill University.*

Nancy G. Leveson *is a Professor of Aeronautics and Astronautics and a Professor of Engineering Systems at the Massachusetts Institute of Technology. She is also the director of the Complex Systems Research Laboratory at MIT. She is an elected member of the National Academy of Engineering (NAE). Prof. Leveson conducts research on the topics of system safety, software safety, software and system engineering, and human-computer interaction. In 1999, she received the ACM Allen Newell Award for outstanding computer science research and in 1995 the AIAA Information Systems Award for "developing the field of software safety and for promoting responsible software and system engineering practices where life and property are at stake." In 2005 she received the ACM Sigsoft Outstanding Research Award. She has published over 200 research papers and is author of a book, "Safeware: System Safety and Computers" published by Addison-Wesley. She consults extensively in many industries on the ways to prevent accidents. She has degrees in math, management, and computer science from the University of California, Los Angeles.*

Michel D. Ingham *is a senior member of the technical staff*

in the Flight Software Systems Engineering and Architecture Group at the Jet Propulsion Laboratory. His research interests include model-based methods for systems and software engineering, software architectures, and spacecraft autonomy. In particular, Dr. Ingham has been involved in the development and theoretical formalization of a state- and model-based systems engineering methodology ("State Analysis"). He earned his Sc.D and S.M. degrees from MIT in Aeronautics and Astronautics, and a B.Eng. in Honours Mechanical Engineering from McGill University in Montreal, Canada.

Kathryn A. Weiss *has a B.S. in Computer Engineering and*

Mathematics from Marquette University (2001). Her advanced degrees include an S.M. (2003) and a Ph.D. (2006) in Aeronautics and Astronautics from the Massachusetts Institute of Technology where her research focused on the intersection between spacecraft, software, systems, and safety engineering with an emphasis on architecture. While at MIT, Katie won the 2005 AIAA Foundation Orville and Wilbur Wright Graduate Award for contributions to the evolution of flight. Dr. Weiss works at the Jet Propulsion Laboratory as a flight software engineer.

APPENDICES

Appendix A: Outer Planet Explorer Intent Specification Notation

The purpose of this Appendix is to explain the acronyms and numbering used in labeling items in Outer Planet Explorer Intent Specification. While these specific acronyms and numbering conventions do not represent a universal standard for intent specification, they should be useful in understanding the OPE intent specification excerpts provided in this paper.

The acronyms are provided in Table A1, while the OPE Intent Specification item labeling conventions are defined in Table A2. Of special note is that the refinements of specification items are labeled through the addition of a numerical digit (e.g., item A&AC-2.1 is refined by A&AC-2.1.1, A&AC-2.1.2, etc.) or by the definition of a new subsystem/functional element (e.g., S/C-2.3 is refined by A&AC-2.1).

Table A1. Functional Element/Subsystem Acronyms

A&AC	Attitude and Articulation Control
AC&DH	A&AC Command and Data Handling
AC	Spacecraft Attitude Controller
AEP	A&AC Electrical Power
C&DH	Command and Data Handling
COM	Communications
DSN	Deep Space Network
EP	Electrical Power
GN	Ground Network
HA&T	HGA Attitude and Translation
HGA	High Gain Antenna
LV	Launch Vehicle
MOC	Mission Operations Center
S/C	Spacecraft
SCI	Science Data Collection
TC	Spacecraft Translation Controller

Table A2. OPE Intent Specification Item Labeling Notation

2.#	High-Level Design Decision Number
A#	Assumption Number
ACC#	Accident Number
DC#	Customer Design Constraint Number
EA#	Environmental Assumption Number
EC#	Environmental Constraint Number
G#	System Goal Number
H#	High-Level Hazard Number
HLR#	High-Level Requirements Number
PC#	Customer Programmatic Constraint Number
PR#	Project Programmatic Risk Number
SED#	State Effects Diagram Number
SC#	High-Level Safety Constraint Number
SV-#	State Variable Number
X#	Functional Element/Subsystem X Sub-Element Definition Number
X-2.#	Functional Element/Subsystem X Design Decision Number
X-A#	Functional Element/Subsystem X Assumption Number
X-C#	Functional Element/Subsystem Design Constraint Number
X-G#	Functional Element/Subsystem X Goal Number
X-R#	Functional Element/Subsystem X Requirement Number
X-SC#	Functional Element/Subsystem X Safety-Related Design Constraint Number

Appendix B: Derivation of Basic HGA Boom Configuration

In this Appendix, the derivation of the basic HGA boom configurations, design decision A&AC-2.2.1, is presented.

Adapting the scientific goals from [9], we have the following goal (shown in Figure B1), among others:

G1. Characterize the presence of a subsurface ocean on an icy moon of an outer planet. (↑ACC4, ACC5), (→HLR3, HLR4), (↓SV-81)

Figure B1. Derivation of the basic HGA boom configuration (part 1 of 11).

This goal establishes the need for the following high-level requirement (shown in Figure B2), among others:

HLR3. The mission shall image TBD% of the surface of the icy moon of the outer planet in spectra other than visual and infrared, to a resolution of TBD. (←G1, G2, G3, G4, G6), (→S/C-G1, S/C-G2, S/C-R1, S/C-R2), (↓2.1, SV-1, SV-101, SV-102)
Rationale: The bands of the spectrum other than infrared and visual provide insights into the chemical composition of the icy moon

Figure B2. Derivation of the basic HGA boom configuration (part 2 of 11).

Given this requirement and the design constraint and assumptions in Figure B3, we decide that we need to send a spacecraft to make the required observations. This design decision is captured in Figure B4.

DC1. The mission must be carried out with existing technologies and space exploration infrastructures as needed (i.e., technologies rated at Technology Readiness Level TBD as defined by NASA). (↓2.1)
Rationale: While technology development is expected to be an ongoing activity of NASA, it is assumed to be beyond the mandate of the mission.

A1. Technology for Earth-based observation of outer planets and their moons is inadequate to achieve the mission goals. (↓2.1)

A2. Technology for Low Earth Orbit based observation of outer planets and their moons is inadequate to achieve the mission goals. (↓2.1)

Figure B3. Derivation of the basic HGA boom configuration (part 3 of 11).

In order for the spacecraft to execute the directives of the human operators of the spacecraft, the spacecraft will need a command and data handling (C&DH) functional element to evaluate these directives, assign them to functional elements of the spacecraft, and otherwise manage the interactions between functional elements of the spacecraft. This need is captured by the design decision in Figure B5.

2.1. A new spacecraft will be used for data collection (↑HLR1, HLR2, HLR3, HLR4, HLR5, DC1, SC1, SC2, SC3, SC4, SC5, SC6, SC7, SC8, SC9, A1, A2, MOC-G1), (→2.2, 2.3, 2.4, 2.5, S/C-2.1, SV-1, SV-2)
Rationale: The data cannot be collected from Earth's surface or orbit. Additionally, no existing spacecraft design will be able to accomplish the mission goals.

Figure B4. Derivation of the basic HGA boom configuration (part 4 of 11).

S/C-2.1. The interfaces of the spacecraft-level functional elements are monitored and managed by a spacecraft command and data handling (C&DH) functional element that translates directives generated by the MOC (after they have been transmitted through the GN or DSN and demodulated) into directives for other functional elements after evaluating if the conditions for the directives are satisfied. (↑S/C-C4, S/C-R3, S/C-R5, S/C-SC1, S/C-SC2, S/C-SC3, S/C-SC4, S/C-SC5, S/C-SC6, S/C-SC7, C&DH-G1), (←2.1)
Rationale: A central concept in control and systems theory is that of hierarchy. By definition, control involves the imposition of constraints from one level of the control hierarchy to a lower level [13]. The C&DH functional element represents control on the spacecraft-level while the other functional elements mentioned in this design principle all control lower-level functions and need a higher-level controller to manage their interactions with each other.

Figure B5. Derivation of the basic HGA boom configuration (part 5 of 11).

In defining the (C&DH) functional element we establish the functional element goal shown in Figure B6.

C&DH-G1. To monitor the health and manage the interfaces of the following spacecraft-level functional elements: Electrical Power (EP), Attitude and Articulation Control (A&AC), Science Data Collection (SCI), and Communications Signal Processing (COM). (←S/C-SC1, S/C-SC2, S/C-SC3, S/C-SC4, S/C-SC5, S/C-SC6, S/C-SC7), (→C&DH-R1, C&DH-R2, C&DH-R3, C&DH-R4, C&DH-R5, C&DH-R6, C&DH-R7, C&DH-R8, C&DH-R9), (↓S/C-2.1, S/C-2.2, S/C-2.3, S/C-2.4, S/C-2.5)

Figure B6. Derivation of the basic HGA boom configuration (part 6 of 11).

The requirement in Figure B7 is among the many requirements necessary to fulfill this goal.

C&DH-R7. The C&DH shall provide spacecraft attitude directives to the A&AC in accordance with the needs of the COM and SCI functional elements. (←C&DH-G1), (↓C&DH-2.1.6, SV-1, SV-2)
Rationale: Spacecraft attitude affects both science data collection and spacecraft-to-Earth communications.

Figure B7. Derivation of the basic HGA boom configuration (part 7 of 11).

Requirement C&DH-R7 is fulfilled through design decision C&DH-2.1.6 and its refinements (refer to Figure B8).

C&DH-2.1.6. The C&DH provides attitude and translation directives to the A&AC functional element in accordance with SCI and COM subsystem needs and MOC Directives. (↑C&DH-R6, C&DH-R7, C&DH-C1, A&AC-C2), (→SV-1, SV-2)
…
C&DH-2.1.6.3. The C&DH receives a 'Science Data Collection' State Information input from the SCI functional element and uses this information to update directives for the A&AC and SCI health and status data. (↑C&DH-ICA10)

Figure B8. Derivation of the basic HGA boom configuration (part 8 of 11).

Design decision C&DH-2.1.6.3 creates the potential for the inadequate control action shown in Figure B9, through the listed control flaw.

C&DH-ICA10. The C&DH determines the wrong directives for the A&AC when considering the Science Data Collection State Information. (←S/C-SC1, S/C-SC2), (↓C&DH-2.1.6.3)

C&DH-CF10.1. During periods of simultaneous data communication with the DSN or GN and science data collection, the C&DH issues directives for the A&AC that degrades one of the two functions.

Figure B9. Derivation of the basic HGA boom configuration (part 9 of 11).

The inadequate control action in Figure B9 relates to both Hazard 1: Inability of Mission to collect data and Hazard 2: Inability of Mission to return collected data, which relate back to accidents ACC4 and ACC5, respectively (these accidents are defined in Figure 8). The safety constraint shown in Figure B10 is warranted to prevent these hazards.

Enforcing this constraint can involve several aspects of the spacecraft system design ranging from C&DH logic design to the physical design of the antenna and data collection instruments. Two of the design decisions made to enforce this constraint are provided in Figure B11. Finally, A&AC-2.2 refines directly into A&AC-2.2.1, shown in Figure 26.

C&DH-SC11. During times of simultaneous communication with the DSN or GN and science data collection, the C&DH must provide directives to the A&AC that do not degrade either function. (←H1, H2), (↓A&AC-2.1, A&AC-2.2)

Figure B10. Derivation of the basic HGA boom configuration (part 10 of 11).

A&AC-2.1. The A&AC functional element rotates the HGA relative to the main structure of the spacecraft. (↑A&AC-R7, A&AC-R8, A&AC-R9, A&AC-R10, C&DH-SC11, A&AC-ICA1, A&AC-ICA2), (→COM-2.1, A&AC-2.2, AC&DH-2.1.2, A&AC-2.4, SV-15, SV-18)
Rationale: Having the HGA articulate relative to the spacecraft decouples antenna pointing from science data collection. If the HGA was spacecraft-fixed, the spacecraft as a whole would have to rotate, possibly altering science data collection instrument pointing (which is essential to satisfactory data collection).

A&AC-2.2. The A&AC functional element translates the HGA relative to the main spacecraft structure. (←A&AC-2.1), (↑A&AC-R7, A&AC-R8, A&AC-R9, A&AC-R10, C&DH-SC11, A&AC-ICA3, A&AC-ICA4), (→SV-4, SV-7, SV-11, SV-15)
Rationale: Translating (or deploying) the HGA away from the main spacecraft structure allows for a wider range of rotation of the HGA (i.e., it distances the HGA from other parts of the spacecraft that it could collide with). It also distances the HGA from potential EMI noise sources in the main spacecraft structure.

Figure B11. Derivation of the basic HGA boom configuration (part 11 of 11).

It is worth noting that design decisions A&AC-2.1 and A&AC-2.2 relate to several state variables (e.g., HGA Frame Position and Orientation with respect to Spacecraft, HGA Boom Elbow Joint Frame Position and Orientation with respect to Spacecraft, HGA Wrist Joint Position and Orientation with respect to Spacecraft, etc.) that will be used in the HGA pointing control laws. Also, it is worth noting that the design decision to articulate the HGA also relates to volume constraints for the spacecraft while it is in the launch vehicle. This constraint-design relationship is derived from an additional thread in the intent specification that split off from the one described in this appendix after design decision 2.1. That thread diverged from the one in this appendix to capture launch vehicle selection and the corresponding volume constraints resulting from that selection. Ultimately, these threads partially re-converged at design decision A&AC-2.2.1, as it was the first design decision to describe volume and packaging characteristics of the HGA boom.