The ASSET Client Interface: Balancing High Level Specification with Low Level Control

Christopher A. Kitts *
Space Systems Development Laboratory
453 Durand Building
Stanford University, Stanford, CA 94305-4035
(650) 725-6794
kitts@leland.stanford.edu

Abstract - This paper describes an advanced client interface for spacecraft operations. This interface permits system clients to specify operations at a level of abstraction relevant to their experience, need, and desire. The interface encourages high-level specification of desired products in which only the relevant product attributes are described. This mode is simple, convenient, and promotes system level flexibility through the elimination of unnecessary constraints. More sophisticated clients may augment this process with lower level directions that mandate particular command and telemetry operations or constrain specific operational variables.

The design of this interface is guided by a Product Specification Model which relates the attributes of general space-based products to the underlying command and telemetry operations that generate those products. By exposing all of the Model's properties to the interface's interview process, a varying level of abstraction is supported during product specification.

This interface is being incorporated into a real-world, experimental mission operations system that consists of several amateur and university-built microsatellites, a global network of remote groundstations, an Internet and amateur radio-based communications infrastructure, and a central mission control center. Initial experimentation suggests that this style of specification capability will contribute to the effectiveness of space systems by 1) enhancing ease of use while conserving the level of control required by some clients, 2) providing a framework for automating the generation of product production procedures, 3) eliminating costly overconstraints commonly imposed through lower level specification, and 4) identifying opportunities to generate a single product that satisfies multiple customers.

This paper describes the operational issues of high and low level specification and presents the conceptual framework for the developed Product Specification Model. The design of the resulting advanced user interface and its implementation within SSDL's mission operations system is also described.

TABLE OF CONTENTS

1. INTRODUCTION

Declining federal budgets and an array of commercial initiatives are providing a significant impetus to improve the competitiveness of space systems. This involves lowering the cost, reducing the cycle time, and increasing the quality and features of a system's products and services. These innovations are particularly important for the broad class of space systems that produce discrete, custom, and/or on-demand products. Such systems include a large segment of NASA science missions, military and commercial space-based imaging systems, as well as certain broadcasting services. NASA's "Science from a Laptop" initiative is one example of a technology program targeted at improving the competitiveness of such services.

As part of its research program, Stanford University's Space Systems Development Laboratory (SSDL) is also developing innovations in this domain. These advancements are being validated through the use of the Automated Space System Experimental Testbed (ASSET). ASSET is a real-world mission operations system which consists of several amateur and university-built microsatellites, a global network of remote groundstations, an Internet and amateur radio-based communications infrastructure, and a central mission control center [1].

In improving space system competitiveness, a particular technological focus has been to improve the process by which clients request products or services. In many

* Doctoral Candidate, Stanford University.

contemporary space systems, this is typically an inefficient process. First, the mechanism by which the request is represented, such as a proposal or form, is often difficult and tedious to use. Second, submission can be slow due to the mode of delivery and the need for iterative, synchronized consultation with a human mission planner. Third, clients are often exposed to complex processing details, such as equipment configurations or station visibilities, that are inappropriate to their level of knowledge or interest. Fourth, the process often forces or induces an overconstrained specification that is actually a small subset of the possible operational implementations that could effectively satisfy the client.

Early SSDL work that aimed at solving these inefficiencies involved the development of a Web-based client interface. This interface collected all necessary specification parameters as part of a context-sensitive interview process [2]. This system directly addressed problems of speed, synchronization, and convenience by automating the request process through a widely accessible, common, and graphical client-server scheme.

An extension to this laid the groundwork for high-level product specification by modifying the interview process to collect product attributes [3]. During this phase, the incorporation of JavaScript significantly improved the performance of the automated interface; JavaScript allowed the migration of request processing to the client's terminal thereby reducing interview delays due to server and communications load.

Current work in this area focuses on 1) generalizing the high-level view of a discrete space product and 2) formalizing the model that relates this view to a set of consistent low-level plans for generating the product. With such a foundation, the resulting client interface is being implemented to exploit this product model by allowing users to specify operations through a combination of high and low-level constraints. The resulting system combines the features of convenient high-level direction with the full authority of low-level control. The interface is being integrated into the ASSET system in order to 1) take advantage of the flexibility afforded by a complete specification of all possible operational options and 2) identify common operational implementations that can satisfy more than one client. As an added feature to further improve the performance of the Web-based interview process, Dynamic HyperText Markup Language (DHTML) is being combined with JavaScript to significantly speed processing and reduce download times.

In describing the design and implementation of the ASSET client interface, the ASSET system if first reviewed. Next, the operational issues of high and low level specification are presented, and the developed Product Specification Model is described. A descriptive tour of selected pages of the implemented client interface is then presented. Finally, future work planned for the interface is noted and general conclusions are drawn.

2. THE ASSET SYSTEM

One of SSDL's primary research activities is the development and validation of new operational innovations that contribute to the system level competitiveness of space systems. In order to conduct this work in a relevant experimental environment, ASSET system is being developed. The ASSET system is a simple yet comprehensive real-world space operations network. This system will be used to operate a variety of academic and amateur microsatellites; in doing so, it will also serve as a low inertia, flexible, real-world validation testbed for new operational methods and technologies.

Figure 1 shows a high level view of the ASSET mission architecture [1]. The basic components include the user interface, a mission control center, groundstations, communications links, and the target spacecraft. During the current developmental phase, a highly centralized operations strategy is being pursued with nearly all mission management executed in the mission control center. These tasks include product specification, resource allocation throughout the ground and space segment, anomaly management, contact planning, data formatting and distribution, and executive control.

Spacecraft

Four university microsatellites are currently being integrated into the ASSET system. SAPPHIRE, SSDL's first satellite, will characterize the space-based operation of experimental infrared sensors, photograph the Earth, and broadcast voice messages [4]. SAPPHIRE is currently undergoing final testing; secondary launch options are being explored. Operations with Weber State University's WeberSat spacecraft, launched in 1990, will include Earth photography and telemetry analysis [5]. WeberSat's on-board software is currently being modified to accommodate these services; during the past year, however, WeberSat has been experiencing intermittent CPU resets which have hampered development. SSDL's second satellite, OPAL, will test a variety of inexpensive commercial off-the-shelf sensors and will validate a launch mechanism for deploying hockey-puck sized science craft [6]. The Barnacle microsatellite, a joint mission between SSDL and Santa Clara University, is being designed to characterize experimental fluxgate magnetometers and a low-cost spacecraft processing system [7]. While these spacecraft have simple missions, it is worth noting that their mission products are reasonable operational analogs to the remote imaging, direct broadcasting, and sensor recording products offered by many industrial, civil, and military space systems.

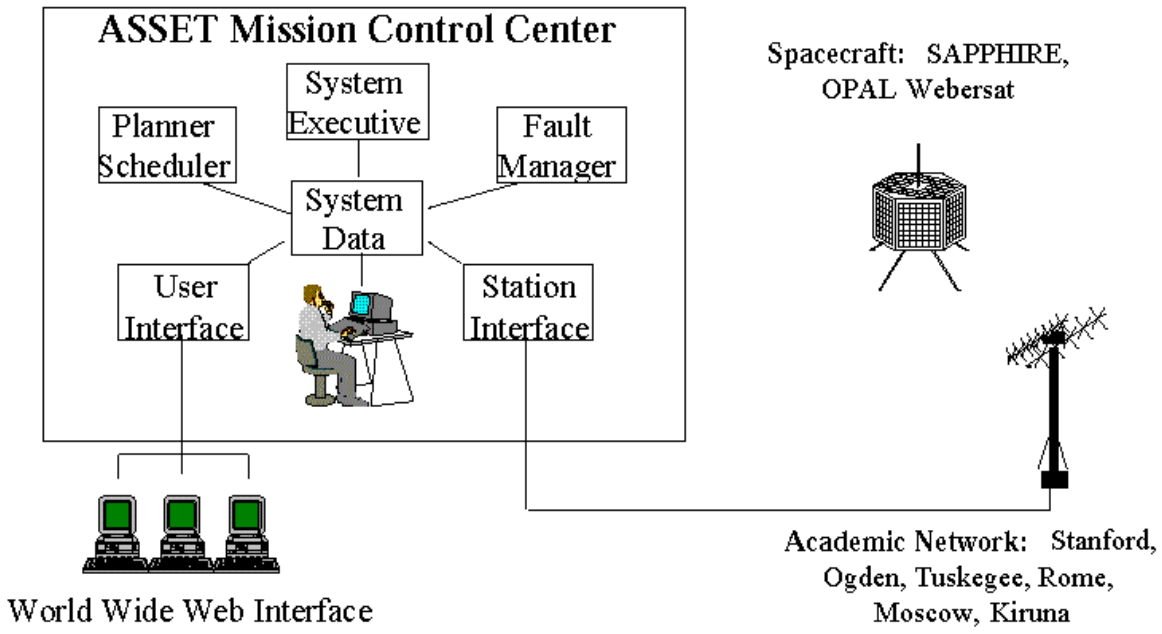

Figure 1 - The ASSET Space System Architecture

Groundstations

The ASSET groundstations employ HAM radio frequencies and equipment commonly used for amateur satellite communications. Typical stations have steerable antennae and use packet radio data formats. To date, facilities at Stanford and at Weber State University have been used for experimentation; the Stanford station runs software capable of supporting both tele-operation and automated, bent-pipe, programmable control. Nearly a dozen other stations throughout the world (Sweden, Italy, Russia, Japan, Saudi Arabia, and across the U.S.) have been identified for future integration. The resulting network will pose planning, scheduling, and execution challenges similar to those currently experienced within the Air Force Satellite Control Network (AFSCN), the NASA Deep Space Network (DSN), and a variety of other large scale space operations systems.

Mission Control Center

The ASSET mission control center resides at Stanford University and consists of several workstations and operators/developers. In the current centralized architecture, the agents in this control center are responsible for mission planning, resource scheduling, executive control, health management, and interfacing with external users and internal engineers. SSDL's research is aimed at understanding these tasks well enough to support automated end-to-end mission products processing and system health management.

System Processing

In the developing mission architecture, clients submit requests for products through a World Wide Web interface; these requests are stored in a central database. Various software modules filter these and system-originated health monitoring requests in order to select products for processing, schedule the spacecraft and groundstation resources required, and plan the low level contact plans necessary. Contact plans are executed via the groundstations using the spacecraft-specific command and telemetry formats. Mission products are returned to the control center for delivery to customers and for storage in a searchable archive. Telemetry is analyzed in order to detect anomalies. Anomalous conditions trigger operator notification, rescheduling of resources to support contingency activities, and a variety of diagnosis and reconfiguration agents in order to assist spacecraft engineers.

System Products and Services

Three high level product/service offerings are currently being developed for external clients within the ASSET framework. Earth photography can be provided by both SAPPHIRE and WeberSat. Synthesized voice broadcasting is generated by SAPPHIRE. Data collection services are provided by all four spacecraft. Data collection is being used specifically for sensor characterization on SAPPHIRE, OPAL, and Barnacle; it is being used for health management operations for SAPPHIRE, OPAL, and WeberSat.

3. HIGH LEVEL DIRECTION VS. LOW LEVEL CONTROL

Before developing the relationships between high and low-level product specification, it is worth considering associated operational issues. These include the benefits obtained through high level direction, the barriers to obtaining true high level specifications in conventional space systems, and why low level specification is still necessary [8].

Benefits of High Level Direction

Simplicity is perhaps the most obvious benefit of being able to direct the actions of a system at a high level. Because an informed user generally knows what is to be achieved, merely describing this end result, typically a product or service, requires no other knowledge concerning the particular structure or behavior of the system. For example, a photograph may be ordered merely by specifying the subject of the photo, the deadline for delivery, and any relevant photographic parameters such as resolution, light level, etc. No knowledge concerning the capabilities or orbits of specific spacecraft, the location of communications stations, the functional status of the system, the system's command and telemetry procedures, or the nature of other product requests are required in order for the client to describe the product of interest. All operational concerns of this nature are transparent to the customer.

An additional benefit of high level direction is that it often results in a less constraining specification for system actions. At the lowest level of control, a request for system action is equivalent to specifying a single possible implementation; this is a fully constrained request in which no operational variable is left as a degree of freedom. Often, however, there are many possible ways in which a system may generate a product with particular attributes. Requesting the product at this high level is equivalent to implicitly specifying a set of many possible implementations; each of these implementations can adequately produce the desired product.

For example, a low level request for a photo to be taken at a particular time results in a single implementation for obtaining the picture of a desired object such as North America. Alternatively, a high level request for the same picture, specified simply as being of North America, may be satisfied by taking the photo any time when North America is in view. More precisely, the operational variable of time is limited to a single value for the low level request. It is constrained only to a time range or a series of time ranges for the high level request; these time ranges are derived through knowledge of the spacecraft's orbit, attitude, field of view, and other system parameters.

Capturing a broader set of possible implementations for a particular request increases the overall system flexibility for satisfying that request. The resulting degrees of freedom can be exploited to accommodate other product requests and/or to optimize product generation with respect to resource utilization.

For these reasons, the ASSET client interface is being designed to accept high level specifications. Furthermore, the planning, scheduling, and execution elements of the ASSET product management system are being integrated in order to exploit the resulting benefits.

Obstacles to Capturing True High Level Directives

Although high level direction capability provides a number of benefits, its integration into large scale space systems is often hampered.

One cause of this is that clients often overconstrain their true goals with particular suggestions on how to achieve that goal. For example, a client may ask for a photograph taken by a specific spacecraft when a variety of spacecraft may be suitable for obtaining the desired photo. This occurs due to attempts to be helpful; the client knows that such a constraint is satisfiable. It also occurs due to ignorance; the client may be unaware of alternative system capabilities that may be more desirable when the client's request is balanced with other system commitments.

Even when the client offers a true high level directive, the process that captures this specification may not be designed to represent requests at such a high level. This can be due to an inflexible method for representing and processing request data. It can also occur due to inefficient experiential processing of the request that effectively compiles it at a lower level. For example, a general request for a photo might be recorded by a mission planner as a request for a photo by a specific spacecraft simply because the planner knows that other spacecraft typically experience heavy operational loads. If this is done to simplify the capture process rather than as a controlled system planning heuristic, then potential operational implementations are needlessly ignored.

Another barrier is the aversion to enlarging the planning and scheduling search space; this certainly occurs if a full set of operational implementations is captured for each system request. But this search space can be easily decomposed based upon heuristics relevant to the planning process rather than those optimized to simplify the capture process. With this approach, the pruning becomes a rational element of the overall system rather than existing as an artifact of limited comprehension and poorly designed functional and/or organizational interfaces.

Finally, specifications are often inappropriately influenced by the request-time state of the space system. For instance, possible product implementations may be ignored if they require resources that are projected to be unavailable. But resource availability is often dynamic due to modifications of system tasking and to the evolving status of resource supplies. As a result, small changes in the system's state can

radically alter the available implementations for a particular product; the sacrificed implementations may then become missed opportunities.

To address these barriers, the ASSET interface encourages the specification of only high level product attributes unless lower level control is necessary. A fundamental system model is used to generate a set of low level implementations consistent with the high level specification. This transformation is done without regard for the request-time state of the system. In this manner, operational flexibility is conserved through the specification interface.

The Continuing Need for Low Level Control

Even with the benefits of high level direction, there are still a number of reasons why low level specification is still a desirable feature. First, using a low level format and language for a request may simply be preferred by the client. This can be especially true for principle investigators who have an intimate knowledge of a scientific payload. It can be useful, however, to make the client aware of any additional costs incurred by this preferred manner of product specification.

Another reason for justifying low level control capability is that the high level specification process may fail to permit control of relevant processing parameters. This certainly occurs with poorly designed interfaces. It also occurs when basic assumptions have been made to provide a simple interface to the majority of system clients. This situation may also develop when the user community develops interest in controlling an attribute of the product not originally assessed as relevant. Furthermore, the user community may discover entirely new applications consistent with the capabilities of the system but completely distinct, at a high level, from the original product offerings. In each of these cases, the high level needs for specification will lag the high level capture capability of the interface; this can be extreme with large scale, complex, and high inertia mission control organizations. Permitting low level control ensures that the system is still applicable to the needs of the clients.

Finally, multi-product considerations, which could be accommodated by permitting direction at a level even higher than the current product level, can require the need for low level control. For example, a campaign of photographs or the gathering of a collection of products that provide any sensory information on an object are legitimate goals for clients. In the absence of campaign level specification, low level control authority provides a means by which a series of product requests can be tailored to the needs of a client.

Overall, the ASSET system addresses these issues by incorporating low level control options into its client interface.

4. THE PRODUCT SPECIFICATION MODEL

In this analysis, a service is defined as a general capability to provide products to clients. A product is defined as a specific action, a tangible artifact, or a set of information that provides value to a client. In order to generate products, particular system configurations are required. The specification of a product is therefore used to constrain the system's configuration at product generation time.

An improper mapping from product to system constraints can result in either of two situations. An underconstrained system permits the generation of inadequate products. An overconstrained system eliminates product implementations that may be optimal from the system's perspective. The methodology by which the specification to system constraint mapping is made therefore has a significant effect on the overall system performance. By developing and applying fundamental models relating the relevant product and operational parameters, a systematic framework can be implemented. This should support improved system efficiency and automation.

In presenting the model currently being developed for implementation in the ASSET system, this section first provides a simplified view of typical system elements. The manner in which a product specification constrains the configuration of these elements is then described. The Product Specification Model used to capture a client's request for a product is reviewed. Finally, observations concerning this modeling approach are made.

The Basic System Schema

Figure 2 shows a simplified conceptual schema for a product processing system of interest. In modeling this schema, the Object-Role Modeling (ORM) technique has been adopted [9]. Previous work on this project used the Entity-Relationship technique for modeling the same domain [10]. The ASSET researchers have found the ORM technique to be better suited to conceptualizing the space processing system primarily due to the manner in which ORM represents objectified relationships.

In the schema of the ASSET domain, system entities capable of generating products are generally classified as tools. These tools have a variety of possible behaviors/capabilities as well as an array of configuration parameters. Some of these configuration parameters are directly controllable. An example of this type of system is the SAPPHIRE voice broadcasting subsystem which has the capability of synthesizing and transmitting voice messages. The subsystem's directly controllable parameters include the message's contents, the message's repeat parameters (number of repeats and interval between repeats), the time of broadcast, and the transmission power. Its constant configuration parameters include transmission frequency, modulation parameters, antenna gain, and orbital elements.

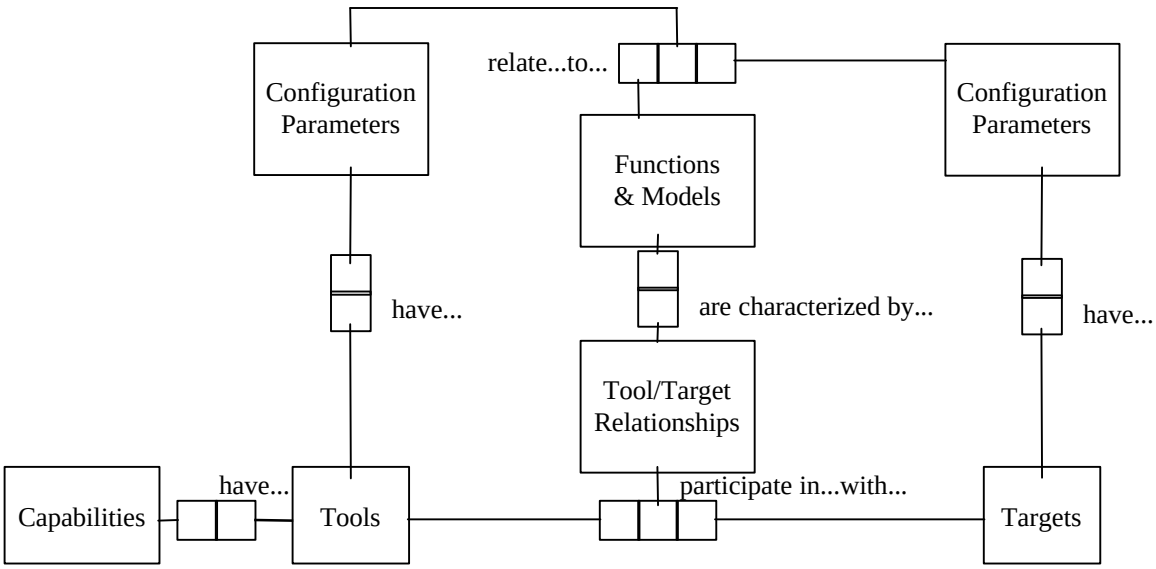

Figure 2 - A Simple System Schema

System entities that are the subject of the system's products or services are generally classified as targets. These targets also have a variety of configuration parameters most of which are not directly controllable in the short term. As an example, a person with a radio receiver would be a relevant target for SAPPHIRE's broadcast system. Configuration parameters for this target include location, receive frequency and performance (noise, sensitivity, etc.), antenna pointing and gain, and environmental conditions (such as rain, etc.).

Tools and their targets participate in roles as part of general relationships. These relationships are characterized by logical and/or mathematical functions or models. The variables in these models relate the tool and target configuration parameters. With respect to the SAPPHIRE voice broadcasting example, there is a line of sight existence/nonexistence relationship between the tool and target entities; this relationship is a function of the locations of the tool and the target. Other relevant relationships include the compatibility/incompatibility of communications link parameters and the existence/nonexistence of adequate link margin.

It should be noted that parameters and relationships relevant to other products also exist. These include items such as light level, orientation, distance, etc.

Specifying a Product

The act of specifying the generation of a product constrains the choice of tools and targets, their parameters, and their relationships. The specification can be made at a low level in which precise tool requirements are levied, or it can be made at a high level in which the attributes of the resulting product are defined.

For the voice broadcasting example, a low level specification would dictate values or ranges for the following variables: tool choice (the SAPPHIRE voice broadcasting subsystem), transmit power, broadcast time, message contents, and message repeat parameters. On the other hand, a high level product attribute specification would constrain values for the following variables: product type (a voice synthesized broadcast), intended recipient (i.e. target), deadline for receipt, message contents, and message repeat parameters.

In both cases, the constrained variables are elements of the system schema pictured in Figure 2. In the low level case, the processing parameters are specified, and a product results from these controlled actions. In the high level case, the product is specified, and a consistent set of processing parameters are derived in order to control the system.

The transformation from high level product attributes to low level processing parameters is performed by 1) directly constraining low level processing parameters when they directly correspond to high level product attributes, 2) constraining the values of the tool/target relationships, and 3) using the constrained tool/target relationships to calculate additional low level processing constraints from the remaining high level attributes.

For example, a typical high level specification includes a particular product type. This product type will require a particular type of processing capability which, in turn, limits the choice of tool. In addition, product type will constrain certain tool/target relationships. To continue the previous example, specifying a voice broadcast as the product type 1) limits tools to those capable of synthesizing and broadcasting voice, 2) requires that a line of sight exists between the tool and the target, 3) mandates communications compatibility between the tool and target, and 4) requires an adequate link margin for the broadcast.

For high level product specification, the product schema and client-supplied product attributes constitute the knowledge base from which consistent system configurations may be derived. These configurations are implicitly defined by the intersection of derived constraints upon the controllable tool configuration parameters. Explicit members of this set characterize the range of low level command parameters required for suitable product generation.

A Closer Look at Constrained Tool/Target Relationships

The process of deriving low level processing parameters from high level product attributes ranges from simple to complex. For example, the line of sight requirement for the voice broadcasting product decomposes into a simple constraint on possible processing times based upon the target's location and SAPPHIRE's orbital elements. This is largely due to the simplicity of the SAPPHIRE microsatellite which has no thrust capability.

On the other hand, a fully specified voice broadcasting product also requires adequate link quality. This results in a far more complex relationship as is shown in Figure 3. The physical constraints are primarily based upon standard communications link models [11]. The tool constraint refers to SAPPHIRE-specific design relationships [12]. Finally, the product constraint mandates adequate link quality; this essentially requires that the broadcast's signal to noise ratio is adequate for reception given the receiver sensitivity.

Reasoning proceeds along the following path. The product constraint mandates a certain minimum value for the broadcast's signal to noise ratio. A physical communications constraint transforms this into a minimum receiver to noise power ratio (since SAPPHIRE's broadcast modulation parameters are fixed). This effect propagates throughout the parameter space due to the enforced constraints. The overall result is that the original product constraint results in coupled limitations on two low level processing parameters: the transmission power and the broadcast time.

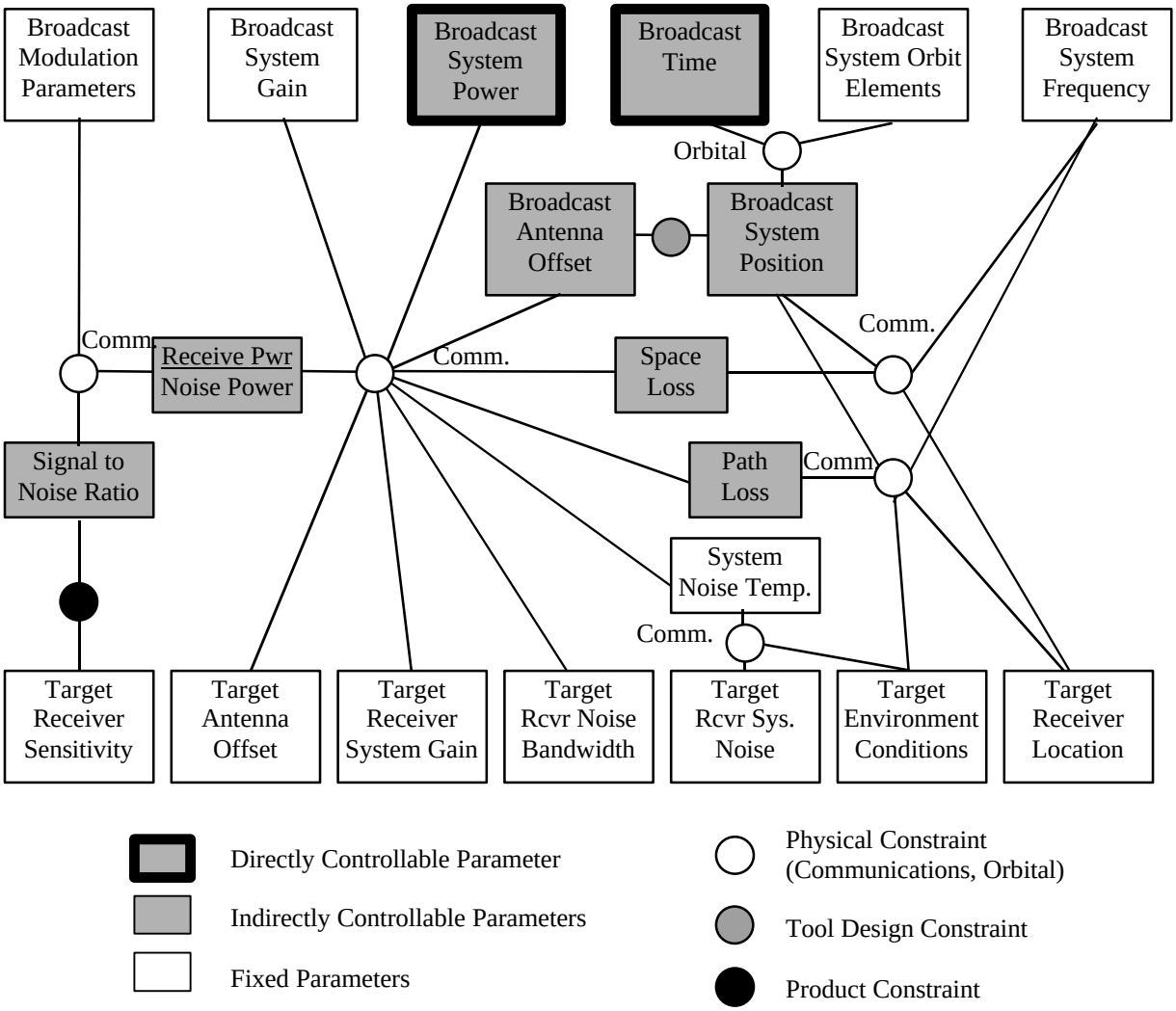

Figure 3 - Link Constraints for the Voice Broadcasting Product

The Product Specification Model

The ASSET interface adapts to the needs of the client by accepting product specifications at a high level, a low level, or through a mix of the two. This is accomplished by dynamically creating and populating a product request object during the interview process. The properties of this product request object include all relevant parameters from the system schema in which there is an operational degree of freedom. By exposing all of these properties to the request process, a varying level of abstraction is supported during product specification.

The Product Specification Model serves as the template for this process by defining the hierarchy of the product request object, the context-sensitive properties, and the constraints between properties. Figure 4 shows a simplified version of the model's object hierarchy. The first generation of properties includes information corresponding to request-specific data, product attributes, and product generation processing parameters. Subsequent generations in the hierarchy further characterize the requested specification; object properties may themselves be objects.

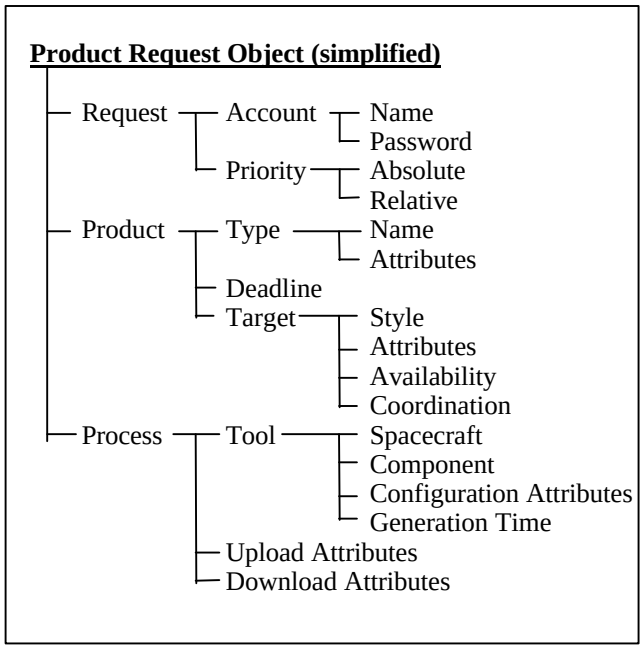

Figure 4 - A Simplified Product Specification Model

Some properties are defined for all possible specifications. Examples of this are the properties in the "Request" portion of the hierarchy; all requests will be assigned these specific properties, either by client specification or by derivation, since they are required for subsequent product processing. On the other hand, some properties, labeled "attributes" in Figure 4, are generic place holders; the object structure of these properties is defined during the interview process based upon previous client input. For instance, the "Product-Type-Attributes" property is an object with varying structure based upon the value of the "Product-Type-Name" property. If a client requests a voice broadcast product, then "voice broadcast" is assigned to "Product-Type-Name" and the internal structure of "Product-Type-Attributes" is defined by the properties "message content", "repeat number", and "repeat interval". A different set of "Product-Type-Attributes" properties are instantiated if a different product type, such as a photograph or data collection, is specified. The process of dynamically creating the structure of the product request object ensures that only relevant properties exist within the specification.

Many of the properties within the product request object are related due to the previously explained tool/target relationships. As a simple example, the specification of a product type will constrain the generation tool to those with the necessary capabilities. Similarly, specifying a specific target for a voice broadcast will place a constraint on processing time since a line of sight must exist between the tool and target.

Validation of the product request is accomplished through the use of both client and server processing. This process ensures a consistent specification with at least one existing implementation. When validated, the specification implicitly represents the set of all possible operational implementations for satisfying the client. Subsequent product processing, such as balancing the needs of competing requests, will add additional constraints to the original specification in order to eventually arrive at the specific set of procedures to be implemented.

Observations

By observing the functions and constraints relating high level product attributes and low level command and telemetry procedures, three particular characteristics are apparent.

The first is a one-to-many product-to-implementation quality. This confirms the general statements asserted earlier concerning the benefits of high level specification in giving a broad set of options. The resulting increase in system flexibility can be exploited by optimizing product generation and/or by increasing system throughput.

A more subtle characteristic is a potential many-to-one product-to-implementation quality. This highlights the fact that multiple product requests may be satisfied by a single operational implementation. For example, the same photograph of a region on the earth might have value for a resident of that region, a scientist gathering photographs of the entire earth, and a spacecraft engineer wishing to test and calibrate the camera. The ASSET planning system is being designed to identify and take advantage of opportunities of this nature. This is being accomplished by implementing a planning/scheduling preprocessor that identifies intersections between the implementation sets of different request specifications.

Finally, it should be pointed out that many services can be decomposed into a series of elemental products. While much of the previous discussion has focused on the system configuration at the time of product generation, post-generation operations are often required in order to retrieve products. For example, with SAPPHIRE, once a photograph has been generated (i.e. taken and stored into spacecraft memory), it must be retrieved from the spacecraft for subsequent delivery. This can be interpreted as a low-level data transfer product that is used to support delivery of client-level products. The ability of the product specification schema to adequately characterize product representations at a variety of levels such as this attests to the generality and value of the developed framework.

5. Implementation in the ASSET System

The principles and models described in this paper are being used to design and implement the client interface for the ASSET space system. This section provides a brief tour of the client interface.

The client interface consists of an automated, context-sensitive interview process that collects relevant specification parameters and submits these to the ASSET database. The interface is a series of Web pages written in HTML, DHTML and JavaScript. Upon request, the ASSET Web server relays the proper files to the client's Web browser. By supporting client-side processing and real-time page composition and layout, the use of DHTML and JavaScript permits speedy contextual processing during the interview, support for a variety of advanced interface elements, and simple state management of the interview process.

Upon loading of the client interface Web page, a product request object is instantiated by the client software. This object contains properties equivalent to the various high level product attributes and low level operational variables relevant to a general product as prescribed in the Product Specification Model. From the perspective of this object, the interview process will dictate the format of and constraints on these properties.

The first step in the client interview process, shown in Figure 5, gathers client's account information and the type of desired product. Account information is used for client authentication. Client name and the chosen product type are stored as properties in the request object; they are also used to tailor the interview process subject to the client's specification authority and chosen product type. As is seen in the figure, the first three product choices correspond to the broad product classes offered by the ASSET system. The fourth and fifth choices are special high use instances of the data collection product applied to sensor characterization of various experiments on the SAPPHIRE spacecraft. Finally, the last choice, not yet implemented, will allow direct entry of command and telemetry parameters which is essentially the lowest level of operational specification possible.

The second step of the interview gathers initial information regarding the product generation time. This information can be obtained in a variety of high or low level ways. The client chooses the desired specification method by selecting among an array of tabbed entry forms within the page. If the product target has been pre-defined within the system, then the simplest high level strategy is to choose the target from a selection list. Because the target location is a stored parameter, it is used to partially define the set of valid product generation times; in addition, choosing a pre-defined target saves time later in the interview since other target-specific attributes are already known. If the target is not pre-defined but a high level product specification is still desired, then the location of the target can be specified directly. The location for Earth surface targets may be designated by entering the latitude and longitude of the target or by clicking the appropriate region on a map. Additional location specification techniques for moving terrestrial targets, such as traveling people or vehicles, and for non-terrestrial targets, such as spacecraft, is soon to be implemented; objects of this type can be selected if they are pre-defined targets. If the client prefers to directly control the product generation time, the third entry option allows this by entering exact time constraints. Current constraint techniques include specifying the exact generation time, mandating before or prior to thresholds, and dictating a precise interest period or series of periods. A fourth technique is currently being added which will permit clients to require that product generation occur during a specific orbital event. The impetus for this is the SAPPHIRE IR experiment in which periods of interest are prescribed by observing the Earth during eclipse.

For this tour of the interface, a voice broadcast is assumed to have been chosen. Figure 6 shows step two in the interview process. The 'Pre-defined Object', 'Specific Location', and 'Period of Time' tabs can be seen across the top of the currently selected form. For this example, the client wishes to broadcast a voice message to a target audience in northern California which has not been pre-defined; accordingly, the 'Specific Location' tab has been selected. With the world map visible, the client selects the desired location and continues with the interview.

For this particular voice broadcast product, the third step of the interview is used to collect a combination of message content and target attributes. This is shown in Figure 7. Determining which of these attributes to gather occurs dynamically in response to previously entered product information. For instance, if the 'pre-defined target' specification method had been used in step two of the interview, then many of the target characteristics would already be known and would not need to be collected in the third step.

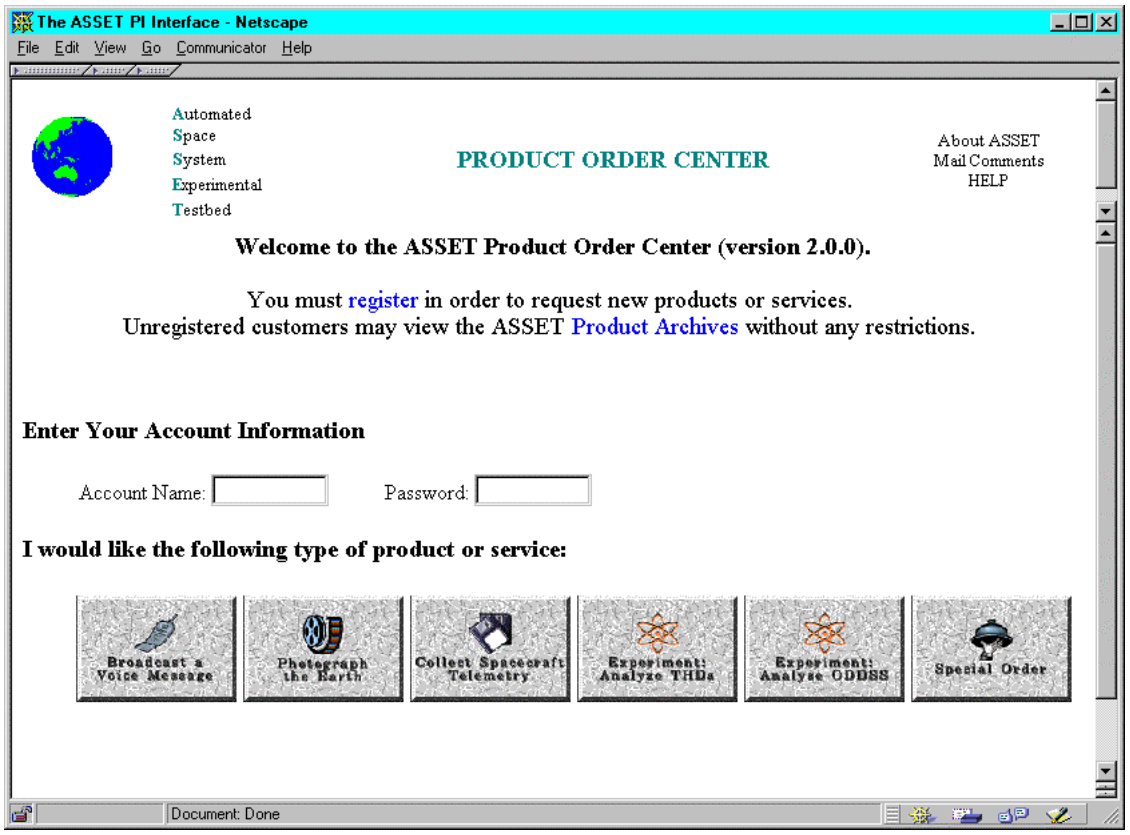

Figure 5 - Specifying Client & Product Type

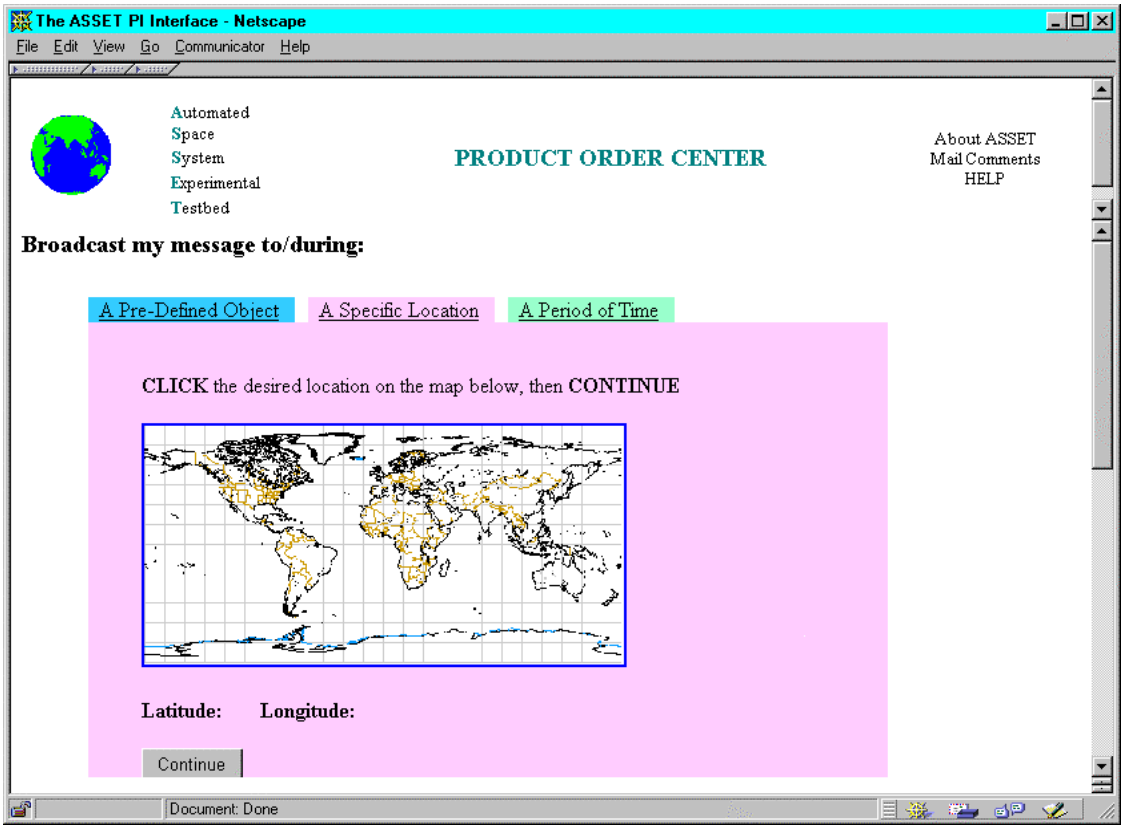

Figure 6 - Specifying the Product's Target

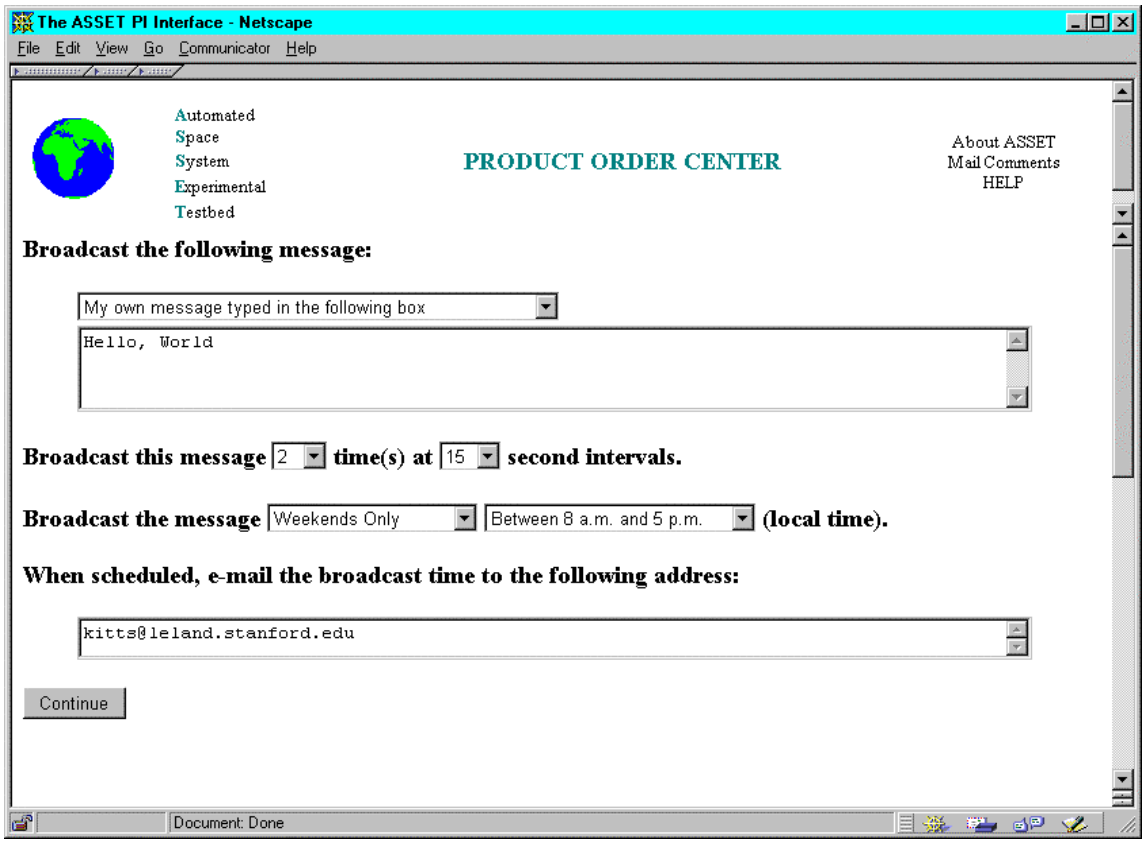

Figure 7 - Specifying Additional Product Attributes

Additional pages in the interview process permit additional control over many of the assumed product parameters, such as link margin, as well as other required request information, such as a cost bid for the requested product.

Upon completion of the interview, the request object is submitted to the request database within the ASSET server. Each of these is then transformed to an operational form consisting of an intersection of its constraints. A blackboard software control system is being implemented to manage additional product processing such as identifying opportunities for satisfying multiple clients with a single product, committing system resources to selected products, and integrating product command and telemetry plans into robust groundstation contact plans. Once generated, photograph and telemetry products are stored within the ASSET product database, and clients are notified of their availability for download.

6. Future Work

Near term work on this project will consist of maturing the product specification model for generality and applicability. One aspect of this will be to formalize the manner in which product-to-implementation transforms are defined so that reuse and extensibility is supported.

Development of the interface itself will concentrate on applying the principles noted in this work across the ASSET domain. In addition, more real-time processing will be migrated to the client workstation so that feedback concerning product existence and remaining degrees of freedom can be integrated within the interface. Additional upgrades include enhanced support for subject definition, client-side data validation, variable client-based specification authority, on-line HELP, real-time product costing, and a method for incrementally modifying a request prior to submission.

A more rigorous validation of the interface is also planned in the near future. Experiments and client evaluations will be conducted in order to gain both quantitative and qualitative metrics concerning the benefits of variable level specification. Particular measures will focus on interface ease of use and freedom in specifying products, improvements in the time and cost involved in processing requests, and the ability to increase the value and/or throughput of products throughout the system. The controlled, prototype nature of the ASSET system is expected to provide a unique opportunity to comparatively evaluate these system level competitive metrics.

In the long run, an even higher level of direction will be integrated into the ASSET client interface. This will support entire operational campaigns specified through simple statements concerning an overall goal. For example,

specifying general interest in a hurricane could trigger a series of photographs and sensor observations from a variety of spacecraft within the system. As always, the overwhelming theme will be to field an interface allowing convenient high level specification without sacrificing the capability to exert low level control.

In addition to evolving the client interface, research will continue in order to develop a variety of other ASSET components in order to achieve more advanced and cost-efficient capabilities. This work includes investigations in planning, scheduling, anomaly management, engineering interfaces, robust execution, software architectures, and systems integration [13]. To ensure the applicability of this work, SSDL collaborates with industry and government organizations; current projects include work with NASA's New Millennium Program and Ames Research Center. Developmental progress is specifically being targeted to permit the expansion of the space system architecture so that it can accommodate more users, missions, experiments, spacecraft, and ground stations. This will certainly require the continuation of many successful partnerships with other universities. Apart from providing a more complex research testbed, this expansion will help to build a significant population of system elements in the sense that the resulting operational methodologies and models will be truly general.

7. CONCLUSIONS

The incorporation of advanced user interfaces is a strategy for increasing the competitiveness of space systems. A particularly useful quality of such interfaces is to permit varying degrees of abstraction in the process of specifying operations to be performed. High-level specification allow clients to conveniently describe the products they desire while conserving operational flexibility. Low-level specification allows clients to precisely control additional processing parameters as required by the client; due to familiarity, sophisticated clients may prefer to use this level of specification even when it affords them no increased level of control. The combination of these capabilities creates a flexible specification process that enables abstract direction without surrendering precise control authority. Preliminary results in using the ASSET client interface attest to these benefits.

In addition to improving the quality and control of services, this work supports improvements in other competitive dimensions. Use of the Product Specification Model lays the groundwork for model-based automation that can ultimately be implemented in order to reduce operational cost and to speed cycle time. Also, throughput can be increased by exploiting a complete set of operational options and by identifying opportunities for mutually satisfying multiple clients with single products. Together, these improvements contribute to making the benefits of space systems directly accessible to investigators, customers, and the public.

Overall, the ASSET system is proving to be a valuable prototype architecture for developing and validating innovations that will contribute to increasing the performance and competitiveness of future space systems. The benefit is clear: as a comprehensive, low inertia, flexible, real world validation testbed, the ASSET system will provide an unparalleled opportunity for experimentation with high risk operational technologies. Furthermore, the academic validation process will assist in supplanting anecdotal analysis commonly performed within the space community with standard evaluation practices aimed at assessing overall system competitiveness.

8. ACKNOWLEDGMENTS

The author wishes to thank current and former research students who have contributed to the development of the ASSET operations network and its client interface. Special thanks is given to current team member Mike Swartwout for his detailed review and feedback regarding this particular work. Also, the insightful comments of this paper's technical reviewers have assisted in making this a stronger contribution. In addition, appreciation is extended to Professor Robert Twiggs, the SSDL Director, for his support and guidance in the administration of this research program. The NASA Jet Propulsion Laboratory and Ames Research Center are acknowledged for their support and assistance with this research project. Finally, appreciation is extended to all of the institutions and organizations that have participated in, contributed to, and provided feedback concerning various aspects of the ASSET system. This work has been performed in partial satisfaction of graduate studies at Stanford University.

REFERENCES

[1] Christopher A. Kitts, "A Global Spacecraft Control Network for Spacecraft Autonomy Research." *Proceedings of SpaceOps '96: The Fourth International Symposium on Space Mission Operations and Ground Data Systems*, September 16-20, 1996.

[2] Christopher A. Kitts and Clemens Tillier, "A World Wide Web Interface for Automated Spacecraft Operation", *ITC/USA '96: Proceedings of 32nd Annual International Telemetering Conference*, October 28-31, 1996.

[3] Christopher A. Kitts, "Specifying Spacecraft Operations at the Product/Service Level", *Proceedings of the 47th International Astronautical Federation Congress*, October 6-10, 1997.

[4] Robert J. Twiggs and Christopher A. Kitts, "SAPPHIRE, A University Student Built Satellite for Space Experimentation", *The AMSAT Journal*, November/December 1995.

[5] Charles Bonsall (ed.), *WeberSat Users Handbook*, Center for Aerospace Technology, Weber State University, January 1991.

[6] Brian Engberg, Jeff Ota, and Jason Suchman, "The OPAL Satellite Project: Continuing the Next Generation Small Satellite Development", *Proceedings of the 9th Annual AIAA/USU Conference on Small Satellites*, September 19-22, 1995.

[7] Sean Boyle, et. al., "The Barnacle Microsatellite", in preparation.

[8] Christopher A. Kitts, *Theory and Experiments in Model-Based Space System Operations*, Draft Research Report, Space Systems Development Laboratory, Stanford University, 1997.

[9] G. M. Nijssen and T. A. Halpin, *Conceptual Schema and Relational Database Design*, New York: Prentice Hall, 1989.

[10] P. Chen, "The Entity-Relationship Model - Toward a Unified View of Data", *ACM Transactions on Database Systems 1:1, 9-36*, March 1976.

[11] Wiley J. Larson and James R. Wertz (eds.), *Space Mission Analysis and Design*, Dordrecht: Kluwer Academic Publishers, 1992.

[12] SAPPHIRE Design Team, *SAPPHIRE Engineering Documentation*, Space Systems Development Laboratory, Stanford University, December 1997.

[13] Christopher A. Kitts and Michael A. Swartwout, "Experimental Initiatives in Space Systems Operations", *Proceedings of the 1997 Annual Satellite Command, Control, and Network Management Conference*, September 3-5, 1997.

Christopher Kitts *is a doctoral candidate in and the Graduate Student Director of Stanford University's Space Systems Development Laboratory. His area of specialty is in spacecraft command and control systems. He is also an engineer with Caelum Research Corporation at NASA's Ames Research Center where he develops spacecraft design and autonomy strategies for NASA's New Millennium Program. Mr. Kitts has served in the Air Force as a mission controller of and the Chief of Academics for the Defense Satellite Communications System III spacecraft constellation. He has held a research position at the Air Force Phillips Laboratory and has taught numerous graduate courses in space system design. Mr. Kitts received a BSE from Princeton University, an MPA from the University of Colorado, and an MS from Stanford University.*