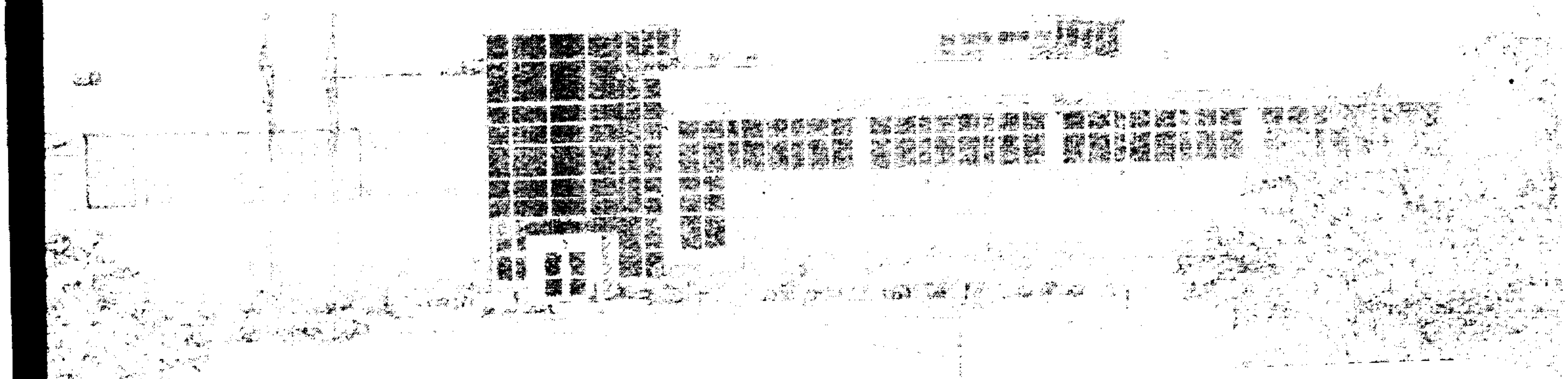
NASA-IVV-94-010
WVU-SRL-94-010
WVU-SCS-TR-95-10
CERC-TR-RN-94-013

NASA/WVU Software IV & V Facility
Software Research Laboratory
Technical Report Series

Development and Analysis of SCR Requirements Tables for System Scenarios

by John R. Callahan and Jeffery L. Morrison



National Aeronautics and Space Administration

West Virginia University

According to the terms of Cooperative Agreement #NCCW-0040, the following approval is granted for distribution of this technical report outside the NASA/WVU Software Research Laboratory

4-10-95
George J. Sabolish Date
Manager, Software Engineering

4-19-95
John R. Callahan Date
WVU Principal Investigator

Development and Analysis of SCR Requirements Tables from System Scenarios

John R. Callahan[1]
Jeffrey L. Morrison
NASA Independent Software Verification and Validation Facility
West Virginia University

Abstract

We describe the use of scenarios to develop and refine requirements tables for parts of the Earth Observing System Data and Information System (EOSDIS). The National Aeronautics and Space Administration (NASA) is developing EOSDIS as part of its Mission-To-Planet-Earth (MTPE) project to accept instrument/platform observation requests from end-user scientists, schedule and perform requested observations of the Earth from space, collect and process the observed data, and distribute data to scientists and archives. Current requirements for the system are managed with tools that allow developers to trace the relationships between requirements and other development artifacts, including other requirements. In addition, the user community (e.g., earth and atmospheric scientists), in conjunction with NASA, has generated scenarios describing the actions of EOSDIS subsystems in response to user requests and other system activities. As part of a research effort in verification and validation techniques, this paper describes our efforts to develop requirements tables from these scenarios for the EOSDIS Core System (ECS). The tables specify event-driven mode transitions based on techniques developed by the Naval Research Lab's (NRL) Software Cost Reduction (SCR) project. The SCR approach has proven effective in specifying requirements for large systems in an unambiguous, terse format that enhance identification of incomplete and inconsistent requirements. We describe development of SCR tables from user scenarios and identify the strengths and weaknesses of our approach in contrast to the requirements tracing approach. We also evaluate the capabilities of both approach to respond to the volatility of requirements in large, complex systems.

1 Introduction

The development efforts of large, complex computer systems are highly prone to cost overruns, schedule slippages, poor designs, and product errors because they involve hundreds of personnel that can misinterpret system requirements. Studies have shown that errors introduced as the result of requirements problems are the most common and the most expensive to fix at later stages of development. These problems especially occur when requirements are changing rapidly and the requirements are not up-to-date across development organizations.

In addition, such systems are developed over many years and must be maintained for 2-5 times as long as their development period. For such projects, early and continuous analysis of software requirements is critical in order to establish guidelines for system evolution and maintenance. These guidelines often stipulate that requirements be stated and maintained in an unambiguous manner to avoid misinterpretations by different development teams.

[1] This work is supported by NASA Cooperative Research Agreement NCCW-0040, NASA Grant NAG 5-2129 and the NASA Headquarters Office of Safety and Mission Assurance (OSMA).

In recent years, many projects have moved away from using traditional requirements documents on paper that simply list the functions that the system "shall" perform, to electronic databases of individual "shall" requirements that can be viewed from many perspectives. Each requirement can be linked to other requirements or system components and users can query the database for requirements with specific properties. This structure allows requirements to be traced throughout the design and implementation of the system at all times.

The ability to trace any artifact of development to a set of requirements is referred to as "housekeeping" because of the ability to "clean up" inconsistencies when changes are made. When a change is made to one requirement, *all* affected requirements and components can be identified. One major problem with housekeeping tools, however, is the meaning of the word "all" in the previous sentence. Nevertheless, modern software projects need the flexibility of housekeeping tools to maintain consistent requirements documents in the face of rapid requirements changes.

Requirements changes, however, cannot simply be introduced without analysis of the impact those changes will have on the existing development effort. Several studies have shown that early and continuous analysis of structured requirements, particularly by external, independent verification and validation (V&V) teams, can significantly reduce software development costs and errors in the long-term [1,2]. V&V teams need powerful tools and methods, including housekeeping techniques, to find problems quickly and effectively. The V&V effort depends heavily on being given adequate visibility into the development team's set of documents, code, and process.

This paper describes our work to improve the methods used by V&V efforts in identifying requirements problems. As an experiment, we applied a formal requirements analysis technique to a subsystem of NASA's Mission-To-Planet-Earth project. We constructed formal descriptions of two subsystems of the EOSDIS Core System (ECS) using the requirement tables method pioneered by the Software Cost Reduction (SCR) project at the Naval Research Laboratory (NRL). We used this method not as a means of describing the ECS requirements, but as a V&V analysis tool to discover anomalies in the existing requirements artifacts. We chose the SCR approach because of its successful use on other projects and the availability of related analysis tools [3,4,9].

Based on analysis of a subset of requirements derived from these scenarios, we found problems involving ambiguous requirements, off-nominal cases, and the independence of organizational components. First, several scenarios were found to contain ambiguous statements regarding processing of data requests. The ambiguities were often discovered during our attempts to characterize system behaviors via the SCR tables. Second, we found that very few scenarios dealt with off-nominal descriptions of flight and data operations. This was evident by the proliferation of positive conditions in the SCR tables and "true" transition events. Finally, the scenarios and derived requirements dealt with interactions between organizations of the ground system elements, but contained no explicit link between system-level scenarios and stated behaviors of individual subsystems in isolation.

First, we discovered several cases of ambiguous statements in the scenario descriptions. These ambiguities were discovered upon reading the scenarios and related requirements during our construction of the SCR tables. In some cases, it was unclear which events or sequence of events occurred for mode transitions in the system. This paper describes one case in which it is unclear

how science data requests on different instrument types are processed relative to other requests and spacecraft nominal operations.

Second, we found that a majority of the scenarios and derived requirements describe only *nominal* case behaviors of the system. The lack of off-nominal descriptions was the most serious set of problems discovered and the source of several ambiguities. Studies have shown that a majority of development and operational problems occur under off-nominal conditions. While the scenarios are the output of the Concepts and Operations phase of development where it is normal that nominal cases describe the desired behavior of the system, requirements derived simply from nominal cases are incomplete. To find incomplete scenarios, the SCR tables helped us to identify complementary behaviors to nominal case behaviors. We discovered the lack of off-nominal cases by noticing that we could not construct meaningful tables for failure mode classes and that many tables lack negative conditions or transitions on failure events.

Finally, it became clear during our construction of the SCR tables that the scenarios describe the external behavior of ECS subsystem components and their interactions during system operations. Yet, there is no link between these scenarios and their derived requirements with descriptions of the internal behavior of any subsystem. For example, scenario X describes how subsystems A and B interact, but does this agree with the individual descriptions of the behavior of subsystems A and B in general? Studies have shown that in large, complex system involving many organizations, each may have its own correct picture of its internal state relative to the system as a whole, but the system state, called the plant state, is different.

In our opinion, housekeeping tools alone are not sufficient to identify such problems and that V&V efforts need formal requirements modeling tools to perform more effectively. Our work is aimed at improving the analysis of V&V organizations and helping development groups create meaningful, unambiguous requirements that enable more effective analysis.

2 The EOSDIS Core System (ECS)

As part of its Mission-To-Plant-Earth project, the National Aeronautics and Space Administration is building the Earth Observing System (EOS) to monitor various aspects of the Earth's ecology. EOS is comprised of over 10 satellites in Earth orbit that each serve as platforms for several oceanographic and geological science instruments. The spacecraft operations, data archiving, and request handling subsystem of EOS system is the EOS Data and Information System (EOSDIS). EOSDIS will serve as the clearinghouse for data requests from end-users scientists around the world. It will merge such requests into daily flight operations and data collection schedules for the EOS spacecraft fleet.

The ECS is the central component of the EOSDIS, providing the coordinating functions for the EOS operational ground system. It constitutes the largest portion of the EOSDIS. The ECS will provide the planning and scheduling, command and control, data processing, data archiving, system management, communications management, networking, and data distribution functions required to support EOS operations and data access. The ECS itself consists of many individual subsystems organizations. The ECS internal organizations relevant to our analysis are the EOS Operations Center (ECS), the Instrument Control Centers (ICCs), and the Information Manage-

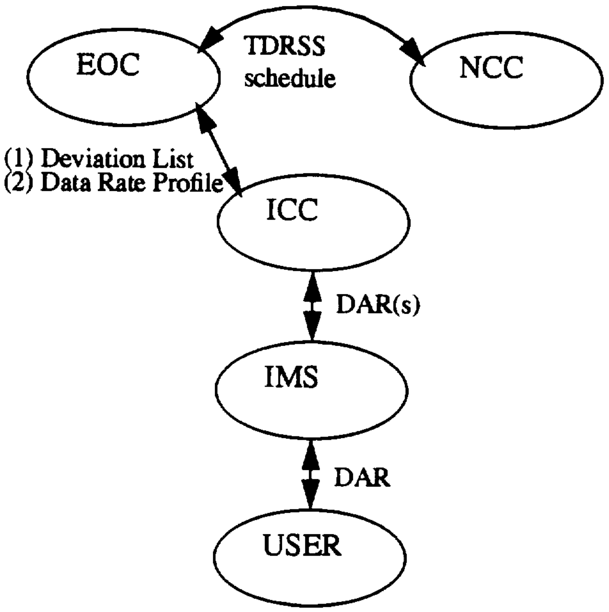


Figure 1: Conceptual View of Data Relationships between some ECS subsystem elements

ment System (IMS)[6]. In addition, we examine the role of an external, independent organization to EOSDIS itself, the Network Control Center (NCC), and its interaction with the ECS subsystems. Figure 1 depicts the general data relationships between these organizations.

2.1 The EOSDIS Operations Center (EOC)

The EOS Operations Center (EOC) is responsible for the mission critical operations of all the U.S. EOS spacecraft. The eight major services the EOC provides are: planning and scheduling, command management, commanding, telemetry monitoring, spacecraft analysis, data management, element management, and user interfaces.

2.2 The Instrument Control Centers (ICCs)

Each Instrument Control Centers (ICC) is responsible for planning, scheduling, commanding, and monitoring the operations of the U.S. instruments on-board one U.S. spacecraft. Functionally, there will be as many ICCs as there are instruments on spacecraft in operation. There will be two different types of ICCs. One type will control a complex instrument and the other will control a non-complex instrument. An instrument is defined as “complex” if it is capable of non-contiguous data acquisition with variable pointing; otherwise, it is defined as “non-complex.” Non-Complex instruments have activities that are normally routine and repetitive.

2.3 The Information Management System (IMS)

The Information Management System’s (IMS’s) primary role is to give the users efficient access to EOS products, providing them with all of the information and tools to search, locate, select and order those products required to perform their science investigations. These products may be

stored in the archives or may entail either higher level processing of an archived product or the placement of an acquisition and processing request.

2.4 The Network Control Center (NCC)

The Network Control Center (NCC) is an important player in the scenarios we analyzed but not a part of the EOSDIS itself. The NCC is responsible for the Space Network (SN), which includes the Tracking and Data Relay Satellites (TDRSS) used for communications to and from most NASA spacecraft. TDRSS is the primary communications link to the EOS spacecraft. Because TDRSS is shared among NASA projects, the NCC negotiates with the completing projects for use of TDRSS resources.

3 Development of Requirements Tables

Our goal was to construct requirements tables based on scenarios in the Concepts and Operations document and analyze the resulting tables for ambiguity, completeness, and consistency with other scenarios and requirements documents. Our hypothesis was that the SCR tables would help structure our small V&V analysis effort of these requirements documents.

The requirements for the EOSDIS are kept in several different documents and databases across different project organizations. The documents relevant to our studies include the ECS Concepts Document [7], the EOSDIS Architecture Document [6], and the ECS Requirements Document [11]. The ECS Concepts Document contains descriptions of scenarios for the ECS system as a whole and different subsystems. The EOSDIS Architecture Document describes the data relationships between subsystems within EOSDIS and ECS. The ECS Requirements Document is managed using a requirements housekeeping tool from which several views and the document itself can be generated automatically.

From the ECS Operational Concepts Document, we chose two scenarios dealing with planning and scheduling of science data requests and their subsequent effects on flight operations. The first scenario involves submission and resolution of Data Acquisition Requests (DARs). This scenario describes the nominal case process of a scientist requesting environmental data that is not already being acquired by normal EOS instrument observations. The other case we studied is called the Initial Scheduling scenario. The Initial Scheduling scenario describes the steps taken to determine the Space Network (SN) resources needed for a single week of EOS satellite requests. The reason we chose these two scenarios was simply that they are among the first ones in the document.

The use of scenarios seems to us to be a natural way for NASA and the ECS developers to describe the intended behavior of the software system before writing requirements. However, scenarios only describe NASA's needs in a very informal manner. Requirements must be derived from scenarios and elaborate behaviors in the face of unanticipated events.

To discover problems in the scenarios and related requirements, we needed to formally describe system behavior in an unambiguous form. We decided upon the use of requirements tables developed by the Software Cost Reduction (SCR) project at the Naval Research Laboratory (NRL) [5,8]. The SCR method represents a system as a finite state machine consisting of modes and mode transitions on conditions that are triggered by events. The SCR approach was selected

Table 1: Data Acquisition Request Mode Class

Current Mode	Special Data Needed	DAR Complete	Complex(t) Non-Complex(f) needed	Enough Info	Meets LT Plan	Meets Guide-lines	Project Scientist Approves	DAR corrected	Requester Notified	New Mode
Idle	@T	-	-	-	-	-	-	-	-	Submit
Submit	t	@T	-	-	-	-	-	-	-	Deter Instr
	t	@F	-	-	-	-	-	-	-	Rejected
Deter Instr	t	t	@T	-	-	-	-	-	-	Complex
	t	t	@F	-	-	-	-	-	-	Non-Com-plex
Non-Com-plex	t	t	f	?	?	?	?	?	?	?
Complex	t	t	t	@T	-	-	-	-	-	LT Plan
	t	t	t	@F	-	-	-	-	-	More Info
More Info	t	t	t	@T	-	-	-	-	-	LT Plan
LT Plan	t	t	t	t	@T	-	-	-	-	Guide-lines
	t	t	t	t	@F	-	-	-	-	Rejected
Guide-lines	t	t	t	t	t	@T	-	-	-	Accepted
	t	t	t	t	t	@F	-	-	-	Review
Review	t	t	t	t	t	f	@T	-	-	Accepted
	t	t	t	t	-	f	@F	-	-	Rejected
Rejected	t	t	-	-	-	-	-	@T	-	submit
Accepted	t	t	t	t	t	t	-	-	@T	Idle

because of its ability to formally model system behavior at an appropriate level of abstraction without reference to the detailed implementation of subsystem components. It is also able to handle descriptions of large, complex, concurrent systems like the ECS [10]. In addition, there are several automated analysis tools that exist for examining SCR tables [3,4].

3.1 The Data Acquisition Request Scenario

The Data Acquisition Request (DAR) scenario describes the process for a scientist to request environmental data that is not already being acquired by normal EOS instrument observations. This scenario starts with a user placing a request with the IMS. The user's request is interactively verified for lexical, syntactic, and (to limited extent) semantic correctness. After the user has properly filled out his request, the IMS forwards the DAR to appropriate ICC for analysis. At the ICC, the DAR will be examined for compliance with the long-term plan. If the DAR lacks clarity, the ICC may contact the user for more details. The accepts or rejects the DAR based on any guidelines and preset priorities. If the DAR does not meet the pre-established guidelines, it is passed on to the project scientist for a decision as whether to accept the DAR or not. Once the DAR has been accepted by the ICC the user is notified.

We developed a single table (Table 1) for the Data Acquisition Request scenario, which contained all the ECS components necessary to complete the process. To produce the table we looked at the scenario and determined what states (modes), events (transitions), and conditions brought about these states. The next step was to populate the table with these modes, transitions, and conditions. The system modes are listed in the far left hand column, the next (new) modes are listed in the far right hand column, and the transitions conditions are written in the top row of the table. To show how a new mode is arrived at from the current mode, we placed the appropriate transition and conditions in the row of the modes in question. A transition on an instantaneous event is represented by either "@T" or "@F". "@T" means when a particular events occurs and may trigger a transition to a new mode. "@F" means that event occurs that triggers a condition to change from true to false. Values of "t" for true and "f" for false are assigned to represent conditions that must be set for some measurable time before a mode transition can occur. A "-" is placed in a condition column if it is not required to characterize a mode transition.

3.2 The Initial Scheduling Scenario

The Initial Scheduling Scenario begins with a complex instrument's ICC generating an instrument data rate profile or a non-complex instrument's ICC generating an instrument resource deviation list, either of which are sent to the EOC. In the meantime, the EOC generates an instrument subsystem resource profile. For a non-complex instrument, a resource deviation list and a subsystem resource profile are combined by the EOC to find the instrument resource profile. The instrument resource profile is used by the EOC to develop a TDRSS schedule request. The EOC sends the schedule request to the NCC two weeks before the target week and then negotiates with the NCC to secure the best possible TDRSS resource allocation. Upon completion of the negotiations, an initial resource schedule is produced. For a complex instrument, a data rate profile and a subsystem resource profile are combined by the EOC to find the instrument resource profile. The EOC validates the resource profile to verify that they are consistent with the spacecraft operational constraints.

Table 2: EOC Initial Scheduling Mode Class

Current Mode	Initial Schedule Needed	SRP Completed	A Non-Complex ICC Deviation List	A Complex ICC Data Rate profile	IRP Completed	Target Week -2 weeks, request to NCC sent	Best possible use of Space Network	Complex ICC Satisfied with Allocation	Renegotiation with NCC Completed	New Mode
Idle	@T	-	-	-	-	-	-	-	-	GSRP
GSRP	t	@T	-	-	-	-	-	-	-	GIRP
GIRP	t	t	t	-	@T	-	-	-	-	TDRSS Schedule Request
	t	t	-	t	@T	-	-	-	-	TDRSS Schedule Request
TDRSS Schedule Request	t	t	-	-	t	@T	-	-	-	Negotiate w/ NCC
Negotiate w/ NCC	t	t	t	t	t	-	@T	-	-	Notify ICC of Allocation
Notify ICC of Allocation	t	t	t	t	t	-	t	@T	-	Idle
	t	t	t	t	t	-	t	@F	-	Renegotiate w/ NCC
Renegotiate w/ NCC	t	t	t	t	t	-	t	t	@T	Idle

IRDL - Instrument Resource Deviation List
GIRDL - Generate IRDL

IDRP - Instrument Data Rate Profile
GIDRP - Generate IDRP

Table 3: ICC Initial Scheduling Mode Class

Current Mode	Initial Schedule Needed	Complex(t) Non-Complex(f)	GIRDL Completed	GIDRP Completed	Received Needed Allocation	Modifica-tions Completed	New Mode
Idle	@T	f	-	-	-	-	GIRDL
	@T	t	-	-	-	-	GIDRP
GIRDL	t	f	@T	-	-	-	Idle
GIDRP	t	t	-	@T	-	-	Waiting for Response
Waiting for Response	t	t	-	t	@T	-	Idle
	t	t	-	t	@F	-	Renego-tiate
Renego-tiate	t	t	-	t	@T	-	Idle
	t	t	-	t	@F	-	Modify Needs
Modify Needs	t	t	-	t	f	@T	Idle

IRDL - Instrument Resource Deviation List
GIRDL - Generate IRDL

IDRP - Instrument Data Rate Profile
GIDRP - Generate IDRP

Table 4: Integrated Initial Scheduling Mode Class

Current Mode	Initial Schedule Needed	Complex(t) Non-Complex(f)	(EOC) SRP Completed	(ICC) IRDL Completed	(ICC) IDRP Completed	(EOC) IRP Completed	Target Week -2 weeks, request to NCC sent	Best possible use of Space Network	Complex ICC Received Need Allocation	Modifica-tions Completed	New Mode
Idle	@T	-	-	-	-	-	-	-	-	-	Generate Info
Generate Info	t	-	@T	t	-	-	-	-	-	-	(EOC) GIRP
	t		@T	-	t	-	-	-	-	-	
	t	f	t	@T	-	-	-	-	-	-	
	t	t	t	-	@T	-	-	-	-	-	
(EOC) GIRP	t	-	t	-	t	@T	-	-	-	-	(EOC) TDRSS Schedule Request
(EOC) TDRSS Schedule Request	t	-	t	-	-	t	@T	-	-	-	(EOC) Negotiate w/NCC
(EOC) Negotiate w/NCC	t	t	t	-	-	t	-	@T	-	-	(EOC) Notify ICC of Allocation
(EOC) Notify ICC of Allocation	t	t	t	-	t	t	-	t	@T	-	Idle
	t	t	t	-	t	t	-	t	@F	-	(EOC) Renegotiate w/NCC
(EOC) Renegotiate w/NCC	t	t	t	-	t	t	-	t	@T	-	Idle
	t	t	t	-	t	t	-	t	@F	-	(ICC) Modify Needs
(ICC) Modify Needs	t	t	t	-	t	t	-	t	f	@T	Idle

After validation, the instrument resource profile is used by the EOC to develop a TDRSS schedule request. The EOC sends the schedule request to the NCC two weeks before the target week and then negotiates with the NCC to secure the best possible TDRSS resource allocation. Upon completion of the negotiations, the EOC notifies the ICC of it's resource allocation. If the ICC received it's needed allocation, an initial resource schedule is produced. However, if the ICC did not received it's needed allocation, the ICC can ask the EOC to re-negotiate with the NCC for an additional resource allocation. If the ICC did not receive the additional resource necessary for the target week, the ICC must adjustments to their resource request to meet the previously negotiated resource allocation. The ICC generates an initial resource schedule.

For the Initial Scheduling scenario, we developed one table for each ECS component (Tables 2 and 3). Then we integrated the component tables into one table representing the entire scenario (Table 4). Developing the mode classes in this manner allowed us the view the components individually and as a whole. Finally, the process of developing an initial resource schedule involves multiple ICCs. However, in the tables, we model the process with just one ICC which could be non-complex or complex and the EOC because we felt the scenario did not adequately describe how all the profiles got integrated to form a TDRSS schedule request.

4 Discussion

Both scenarios made no mention of what state the processes were in at the beginning of execution. We introduced the Idle mode into the scenario to model the organization waiting for external inputs. The Idle mode did not affect the scenarios and it did help define the processes during their inactive periods. However, it is currently unclear how this external Idle mode agrees or conflicts with the internal organizational behaviors.

4.1 Ambiguity

The Initial Scheduling scenario describes scheduling one instrument at a time. However, this process occurs concurrently for all instruments. It is unclear how the different ICCs, the EOC, and the NCC negotiate instrument schedules. We hypothesize that one source for this ambiguity comes from a familiarity of the requirements authors with the application domain (i.e., ground satellite support systems). Many behaviors are assumed to occur in nominal cases that appear ambiguous to external reviewers. We believe that this is unacceptable given that large projects have increased personnel turnaround over the long-term. The loss of expertise increases the likelihood that wrong assumptions are made by inappropriate personnel [1].

For the Data Acquisition Request mode class, the table was relatively easy to develop because the scenario had been written with definite actions and decisions that had to be performed during the process. However, the following passage from the ECS Operations Concept Document was unclear as how to handle non-complex instruments:

> Once the DAR has been submitted to the IMS, the IMS will forward it to the associated ICC for analysis. DARs that request coordinated observations between a complex instrument and a non-complex instrument and a non-complex instrument will be forwarded to the appropriate ICCs and the complex instrument ICC will have responsibility for coordinating the observation. At the ICC, the DAR will be

> examined for compliance with the long-term plans, and expanded if only minimal information is provided.

In the last sentence of the passage, we had difficulty determining if the authors were describing a complex ICC or both types of ICCs. If they were only describing a complex ICC, we had no idea how a request for a non-complex instrument completed. In Table 1, we entered question marks for the non-complex mode. It was possible that the authors just wanted to describe in a single statement the coordination between complex and non-complex instrument for intricate data requests. If this were true, the process of requesting data would be exactly the same for both complex and non-complex instruments. Again, the potential for an ambiguous interpretation leaves room for errors at later stages of development.

The scenarios for the Initial Schedule process lacked a definite flow of events. The authors tried describing the process of scheduling a complex and non-complex instrument simultaneously, which caused some confusion. The description of the non-complex request processing lead us to believe it never had a problem getting its needed allocation. Whereas the complex instrument had resource usage checks during the scheduling process.

Another problem with the Initial Scheduling scenario was that the authors repeated the process of generating the non-complex instrument's resource deviation list, the complex instrument's data rate profile, and the instrument's subsystem resource profile twice with some variations between both descriptions. The following passage from the Initial Scheduling Scenario is the first description of the process of generating the non-complex instrument's resource deviation list, the complex instrument's data rate profile, and the instrument's subsystem resource profile.

> ...A complex instrument ICC generates an instrument resource profile and sends it to the EOC. For non-complex instruments the ICCs and PIs/TLs generate instrument resource deviation lists and send them to the EOC. In the meantime, the EOC identifies the SN resources required for subsystem operations (e.g., TONS operations, orbit adjustment operations) in a spacecraft subsystem resource profile. Based on the baseline activity profiles and instrument resource deviation lists for the non-complex instruments, the EOC generates their instrument resource profiles. By combining all the instrument resource profiles with the spacecraft subsystem resource profile, the EOC estimate's the recorder's data volume profile for SN resource needs and develops a TDRSS schedule request....

The next passage reiterates a description of the process for developing the non-complex while providing no additional information.

> ...For each non-complex instrument that has an instrument resource deviation list, the EOC builds an instrument resource profile by combining the resource deviation list with the corresponding resource profile associated with the baseline activity profile...

The complex instruments role is described again with the authors giving some better insight to the process.

> ...Complex instruments, in contrast, have resource needs that cannot be predicted much in advance of the initial scheduling phase. The ICC for such an instrument generates an approximate, yet adequate, instrument resource profile that incorporates accepted DARs, anticipated observations and instrument support activities... Once created, the instrument resource profile, containing a data rate profile along with other resource profiles, is sent to the EOC...

Finally, the authors repeat the steps necessary to produce the spacecraft subsystem resource profile while again provide very little in the way of new information.

> ...The spacecraft subsystems also have resource needs that must undergo initial scheduling. From the long-term spacecraft operations plan, the EOC identifies the activities that the subsystems must perform during the target week. Based on these activities, the EOC generates a spacecraft subsystem resource profile. The EOC uses the data rate information to identify the spacecraft subsystems' SN resource needs...

The scenario would have been easier to follow if the redundancy had been removed and if the scheduling process for complex and non-complex instruments had been better explained. Even though in some cases, the intent of requirements seem clear there exists the potential for misunderstanding. The construction of the tables elucidated such ambiguities.

4.2 Lack of Off-Nominal Descriptions

The scenarios make little mention of off-nominal cases. Likewise, any requirements derived from these scenarios describe only nominal behaviors of the delivery software system. This can be easily seen from the proliferation trues and very few false conditions in tables we developed. In addition, it was difficult to construct meaningful failure mode classes from the scenarios. We believe that development of the SCR tables by a V&V organization can help quickly identify safety and fault tolerance problems.

We are currently trying to apply some automated tools for analyzing SCR tables for completeness with respect to detecting the lack of off-nominal conditions. We believe that they may be able to detect the lack of off-nominal conditions when trying to prove invariant safety properties of the system. The ability to prove various assertions about safety conditions will allow us to audit the SCR tables and introduce complementary conditions and failure mode classes that more completely describe system behaviors.

Current analysis via housekeeping tools can easily miss the lack of off-nominal requirements specifications. We are continuing to explore analysis of completeness of our specifications and recommend enhancements to the development organizations.

4.3 External vs. Internal Behaviors

Trying to develop separate mode classes for each component in the ECS and then merging the mode classes to find interface inconsistences between components was not very useful in finding interface inconsistencies. This was because the scenarios described how the components would work together, thus describing the interfaces between them. However, we believe the process of creating the individual mode classes and integrating them can be helpful. We believe the process provides insight into a components relationship to the overall scenario. Also, user/developer can examine the individual component mode classes to verify if their perceived views of the components agree with what they have described in the scenario.

In the combined Initial Scheduling Mode Class, we had a problem with concurrent operations. To model this we had to show all possible orderings for completion of the concurrent events. This can be a problem for a large number of concurrent operations in a scenario. A somewhat related problem that can arise using these table is when you have a large scenario with a large number of modes and transitions. This would cause the table to be very large and physically impossible to fit on one or two pages.

4.4 Flexibility

The requirements for a large system like EOSDIS are constantly under review and change. The housekeeping tools used by the current development team allow for auditing of requirements in the face of such changes, but it is difficult to trace the subtle, propagated effects of changes to other subsystems. These indirect effects are especially overlooked when off-nominal conditions are ignored. For example, if we remove a required step in the DAR process, we can trace the removal of requirement to the DAR process and related processes, but if an off-nominal behavior (especially of another process) depends on this step, then the change might not be noticed.

When we changed conditions in the SCR tables, it was straightforward to audit all affected mode transitions. Each row that had a checkbox in it had to be analyzed with respect to the new condition. Furthermore, because of the formal specification of table conditions, it was easy to track changes needed to multiple conditions.

As part of a V&V metrics program, we are instituting a research study to look at requirements dynamics under the EOSDIS development. By analyzing the effects of requirements changes and the depth of propagation of the effects of such changes into the system requirements, design, and code during development we hope to gain a better understanding of defining "flexibility" in more formal terms. Some metrics we are examining include the number of links followed in the housekeeping graph, the number of changed mode transitions, and the number of affected modules. We believe that it may be possible to measure the severity of requirement changes in terms of stress on the software design. This will be particularly useful in the maintenance phase in trying to prioritize or reject requirements changes that result from discrepancy reports and engineering change requests during system operations.

5 Conclusions and Future Directions

Constructing the tables was a valuable exercise because they provided a common, unambiguous

media for fostering an understanding between developers, domain experts, and V&V team members. Even when we constructed naive, abstract views of system behaviors, the requirements experts were able to quickly correct and increase the detail of our tables.

We believe that developing SCR tables from scenarios is an excellent way to formalize software system requirements, but they can also be used by V&V organizations to structure their reviews of more traditional requirements artifacts. By using tables to model subsystem behaviors, we can discover ambiguities, off-nominal cases, and reconcile organization views.

We are only at the beginning of our partial analysis of the requirements for ECS and the subsequent SCR tables. In the future, we will apply the same techniques to analyze flight operation behaviors in the ECS. In response to our lack of understanding the process of merging experiment schedules, we are also trying to clarify the techniques used for resolving conflicts. We believe that the SCR tables will continue to provide a common media for interaction between developers, researchers, and V&V practitioners.

References

[1] National Research Council, An Assessment of Space Shuttle Flight Software Development Processes, National Academy Press, Washington, D.C., USA, 1993.

[2] NASA JPL, The Cost-Effectiveness of Independent Verification and Validation, NASA Jet Propulsion Laboratory, Technical Report, 1985.

[3] Atlee, J., Automated Analysis of Software Requirements, Ph.D. thesis, University of Maryland - College Park, 1993.

[4] Campos, S., E. Clarke, W. Marrero, M. Minea, and H. Hiraishi, Computing Quantitative Characteristics of Finite-State Real-Time Systems, CMU Technical Report CMU-CS-94-147, May 1994.

[5] Heninger, C., Specifying Software Requirements for Complex Systems: New Techniques and Their Application, IEEE Transactions on Software Engineering, Volume 6, Number 1, January 1986.

[6] SES, Inc., EOS Ground System Architecture Description Document, Systems Engineering and Security, Inc., Greenbelt, Md., April 1993.

[7] HAIS, Inc., ECS Operations Concept Document for the ECS Project, Hughes Applied Information Systems, Inc., Landover, Md., August 1993.

[8] Faulk, S., State determination of hard-embedded systems, Ph.D. thesis, University of North Carolina - Chapel Hill, 1989.

[9] Heitmeyer, C. and B. Labaw, Consistency Checks for SCR-Style Requirement Specifications, Technical Report NRL/FR/5540--93-9586, Naval Research Laboratory, Washington, D.C., USA, December 1993.

[10] Alspaugh, T., S. Faulk, K. Heninger Britton, R. A. Parker, D. Parnas, J. Shore, Software Requirements for the A-7E Aircraft, Technical Report NRL/FR/5530--92-9194, Naval Research Laboratory, Washington, D.C., August 1992.

[11] NASA Goddard Space Flight Center, Functional and Performance Requirements Specification for the Earth Observing System Data and Information System (EOSDIS) Core System (ECS), NASA Goddard Space Flight Center, Greenbelt, Md., June 1994.