REFERENCE ARCHITECTURE FOR SPACE DATA SYSTEMS

Peter Shames
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA

Takahiro Yamada
Institute of Space and Astronautical Science
Sagamihara, Japan

ABSTRACT

This paper introduces the Reference Architecture for Space Data Systems (RASDS) that is being developed by CCSDS. RASDS uses five Views to describe architectures of space data systems. These Views are derived from the viewpoints of the Reference Model of Open Distributed Processing (RM-ODP), but they are slightly modified from the RM-ODP viewpoints so that they can better represent the concerns of space data systems.

KEYWORDS

System Architecture, Space Data Systems and CCSDS.

INTRODUCTION

Interoperability of space data systems is of great concern to Space Agencies because sharing or reusing interoperable resources among multiple projects and multiple Agencies can reduce the cost of developing and operating space data systems. However, an on-going problem is that each space data system often has a different architecture and therefore the elements of one system cannot be easily used by other systems. Moreover, the method of describing the architecture is usually different from system to system and it is sometimes difficult to even describe the problems associated with interoperability among systems. Standard interfaces and protocols, and standard architectures, are ways of providing interoperability and reuse, and reducing costs.

To cope with this situation, the Consultative Committee for Space Data Systems (CCSDS) [1] has developed various architectures to describe space data systems. CCSDS is an international, consensus based, space system standards organization which has as members NASA, ESA, ISAS, and all the other major space agencies. Recognizing that there are already different architectures of space data systems, the approach taken by the CCSDS System Architecture Working group (SAWG) was to generate a reference architecture that can be used as a framework to generate various architectures in a coherent way. This reference architecture is known as the Reference Architecture for Space Data Systems (RASDS). From this reference architecture, architectures of different space data systems can be generated in a standard way so that the commonality and differences among the systems can be easily understood. This leads to the understanding of how a system can be used by other systems and which interfaces should be used to connect the systems. This further leads to the identification of standard interfaces and protocols that can be used by multiple systems and standard elements that can be shared by multiple systems.

Once they are developed and approved these interfaces and protocols are broadly adopted by the space agencies and commercial products are produced that are compliant with these standards. The end result is a high level of interoperability and cross support, lowered costs for individual missions and agencies, and reduced risk due to the availability of well tested and proven components.

OVERVIEW

Space data systems are complex entities, which may be viewed from various aspects. In order to generate the architecture of a space data system in a manageable way, RASDS uses multiple Views to present the architecture of a space data system, each view focusing on one aspect of the system. The Views used by RASDS are derived from the viewpoints defined in the Reference Model of Open Distributed Processing (RM-ODP) [2] but they are slightly modified from the RM-ODP viewpoints so that they can better represent the concerns of space data systems. The views used in RASDS range from the organizational to the physical component and from abstract representation to concrete implementations, they include: Enterprise, Connectivity, Functional, Information, and Communications.

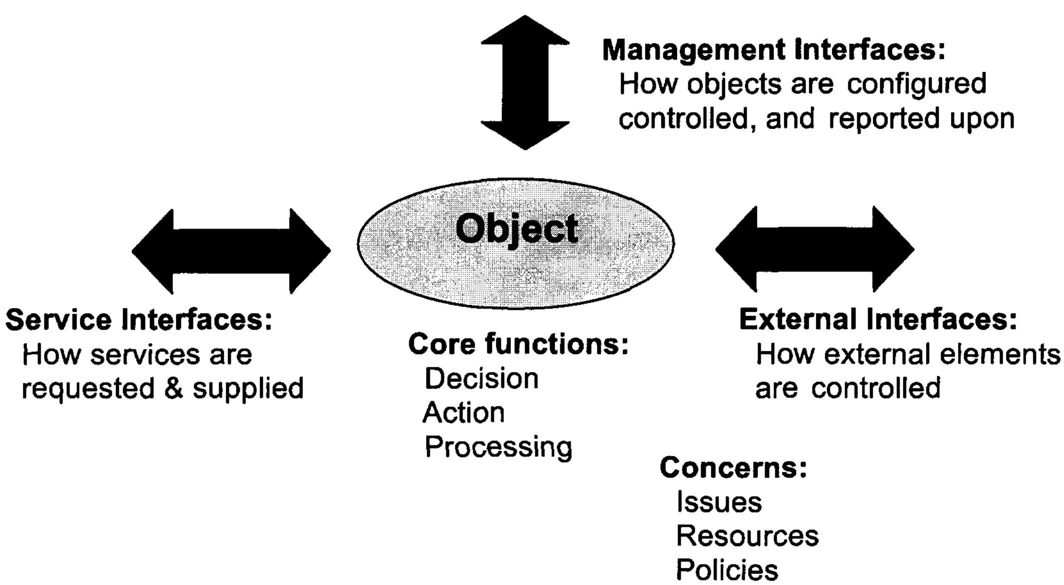

Figure 1. Representation of Objects

Each View is an abstraction that uses a selected set of architectural concepts and structuring rules, in order to focus on particular aspects within a space data system. Each of the Views describes the space data system in question as a set of Objects, the interactions among them, and the concerns that must be addressed for that viewpoint. An Object is an abstract model of an entity in the system.

As shown in Figure 1, each Object is described with its core functions and interfaces with other Objects. Also, a set of concerns is associated with each Object.

RASDS uses the five Views that are explained in the subsequent sections to describe the architecture of space data systems. The user may decide not to use all of these five Views to describe a particular system if the system can be characterized with less than five Views. The user may also choose to combine Views using the basic concepts defined in RASDS if it is impossible to capture all the important aspects of the system with a single pre-defined View. Examples of this are shown in the text.

ENTERPRISE VIEW

The motivation for the Enterprise View is that we often have complex organizational relationships involving spacecraft, instruments, ground systems, scientists, staff, and contractors that are distributed among multiple organizations (space agencies, science institutes, companies, etc). The Enterprise View is used to address these aspects of space data systems and the relevant concerns that arise, i.e. polices, contracts, agreements, organizational interfaces and, from a security perspective, trust relationships.

The Enterprise View describes the organizations involved in a space data system and the relationships and interactions among them. The Enterprise View is depicted as a set of Enterprise Objects and interactions among them, where each Object is an abstract model of an organization or facility involved in a space data system. An Enterprise Object represents an independent Enterprise (such as a space agency, a government institute, a university, or a private company) or an element belonging to an Enterprise (such as a tracking network, a control center, a science center, or a research group). An Enterprise Object may be composed of other Enterprise Objects. A group of Enterprise Objects that plays some role in a space data system (such as a community, a committee, or a joint project) can also be an Enterprise Object.

Table 1 shows typical Enterprise Objects used in space data systems.

Table 1. Typical Enterprise Objects

Enterprise Objects	Description
Space Agency	An Enterprise Object that is responsible for building and managing space missions.
Ground Tracking Network	A multi-mission Enterprise Object that may be comprised of one or more tracking stations, used for communicating with spacecraft and performing radiometric measurements against spacecraft.
Spacecraft control center	An Enterprise Object that is responsible for controlling a spacecraft.

Instrument control center	An Enterprise Object that is responsible for controlling (a) instrument(s).
Science Institute	An Enterprise Object that requests activities of a spacecraft and analyzes data obtained from that spacecraft.

Figure 2 shows an example of an Enterprise View for Mission A, in which Agency P builds and operates a spacecraft, Agency Q provides tracking support and Science Institute R performs scientific data analysis.

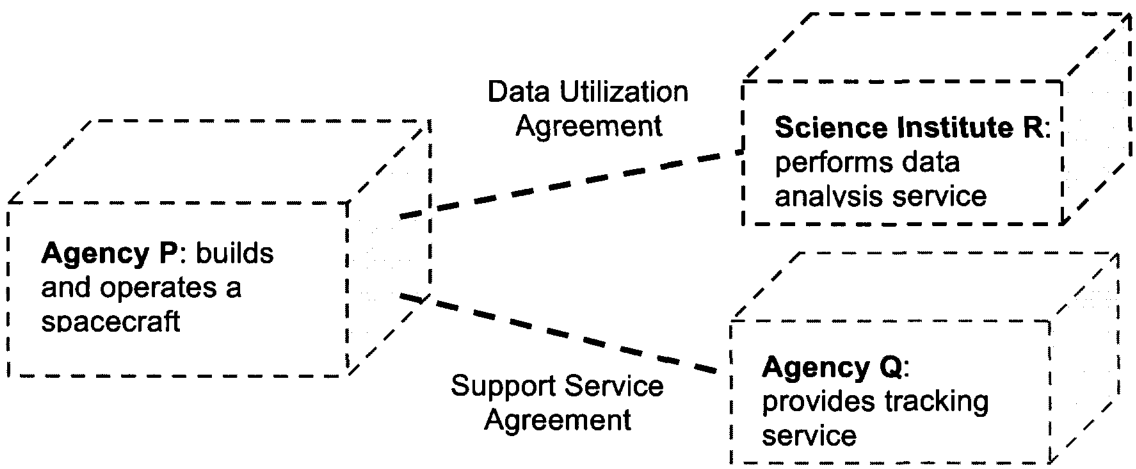

Figure 2. Example of Enterprise View (Mission A)

CONNECTIVITY VIEW

The Connectivity View describes the physical elements, how they are connected, and the physical environment of a space data system. The Connectivity View is depicted as a set of Nodes and Links. A Node is an abstract model of a physical entity or component used in a space data system, which is connected to other Nodes by a Link of some sort. A Node represents a system (such as a spacecraft, a tracking system or a control system) or an individual physical element of a system (such as an instrument, a computer, or a piece of equipment). A Node may be composed of other Nodes. A Link is a physical connection between or among Nodes. A Link represents an RF link, a wired link, or a network of some kind (such as the Internet, a LAN, or a bus). Both Nodes and Links have associated behavioral properties, which include performance, location, and possibly motion. The entire set of Nodes and Links is embedded in a physical environment, which has its own properties and behaviors.

The motivation for the Connectivity View is that we have system elements that are in motion through space and consequently connectivity issues associated with pointing, scheduling, long round trip light times, and low signal-to-noise ratios, all of which require special protocols and functionality to deal with. The Connectivity View is used to address all of these physical and performance aspects of space data systems. This is a concrete view of system elements, used in conjunction with more abstract views, such as the Functional View, to show allocation of functions, and with more concrete views, such as the Communications View, to show the protocols that are required to deal with the link and environmental characteristics.

Tables 2 and 3 show typical Nodes and Links, respectively, used in space data systems.

Table 2. Typical Nodes

Nodes	Description
Spacecraft	A Node in space used to achieve mission goals.
Relay satellite	A spacecraft that relays data between spacecraft and a tracking station or between different sets of spacecraft.
Instrument	A sub-Node in a spacecraft used to achieve mission goals.
Tracking station	A Node used for communicating with spacecraft and performing radiometric measurements against spacecraft.
Spacecraft control center	A Node used to control a spacecraft.
Science center	A Node that requests activities of a spacecraft and analyzes data obtained from that spacecraft.

Table 3. Typical Links

Nodes	Description
Space link	A Link between a Node in space and a Node on the ground, or between two Nodes in space.
Ground link	A Link between Nodes on the ground.
Onboard bus	A Link among multiple Nodes on a spacecraft.

Figure 3 shows Nodes and Links used for Mission A , as shown in Figure 2.

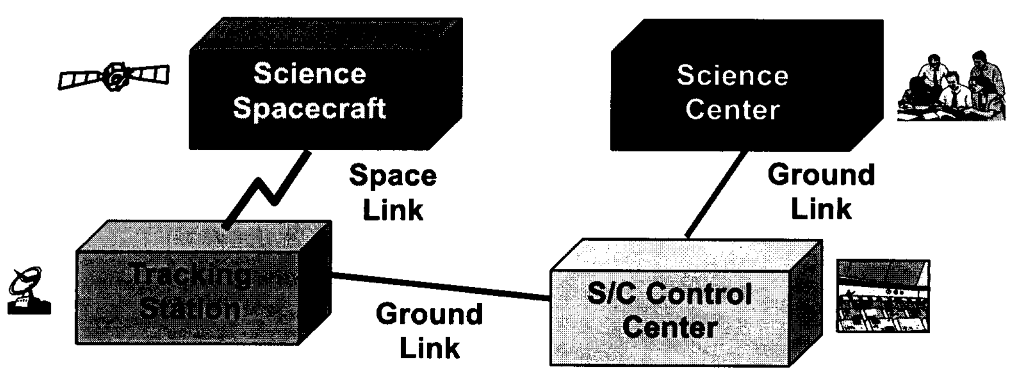

Figure 3. Example of Connectivity View (Mission A)

An Enterprise Object owns each Node. Figure 4 shows which Enterprise Object from Figure 2 owns which Node(s) from Figure 3.

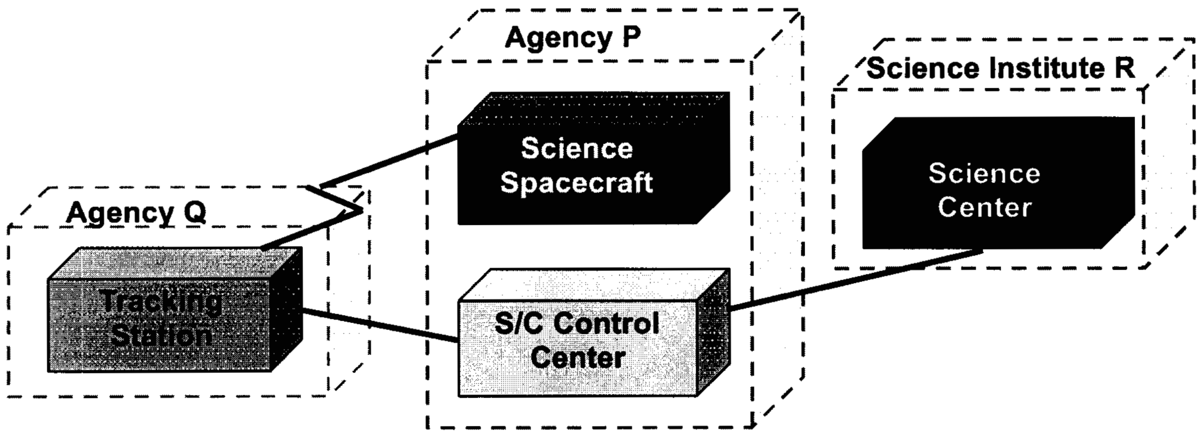

Figure 4. Example of Enterprise and Connectivity Views (Mission A)

FUNCTIONAL VIEW

The motivation for the Functional View is to separate functional elements and their logical interactions from the engineering concerns of where functions are housed, how they are connected, which protocols are used, or which language is used to implement them. The Functional View is an abstract view used to address these aspects of space data systems.

The Functional View describes the functional structure of a space data system and how functions interact with each other. The Functional View is depicted as a set of Functional Objects and the logical links among them. A Functional Object is an abstract model of a functional entity that performs actions and generates or processes data in a space data system. Each Functional Object has a set of associated behaviors and a set of defined interfaces. An Object that only moves data is called a Communications Object and is treated in the Communications View. A Functional Object may be realized as either software or hardware. A Functional Object may be composed of other Functional Objects. A Functional Object may use a service provided by other Functional Objects, provide a service to other Functional Objects, or perform actions jointly with other Functional Objects. These kinds of interactions are described in the Functional View.

Table 4. Typical Functional Objects

Functional Objects	Description
Data Acquisition	A Functional Object to collect data using an instrument.
Directive execution	A Functional Object to execute a set of directives (goals or a time-ordered set of commands).
Directive generation	A Functional Object to generate a set of directives (goals or a time-ordered set of commands) based on a mission plan.
Mission planning	A Functional Object to generate a mission plan (time-ordered set of goals or sequence of activities).

Spacecraft analysis	A function to analyze the status of a spacecraft using data received from a spacecraft.
Science analysis	A Functional Object to analyze the status of instruments and to extract scientific values from collected data.
Data management	A Functional Object to manage data exchanged among other spacecraft and ground-based functions.
Tracking	A Functional Object to track a spacecraft in order to communicate with the spacecraft and to performing radiometric measurements.

Figure 5 shows some of the Functional Objects used for Mission A together with the logical interactions between them (shown with dotted lines).

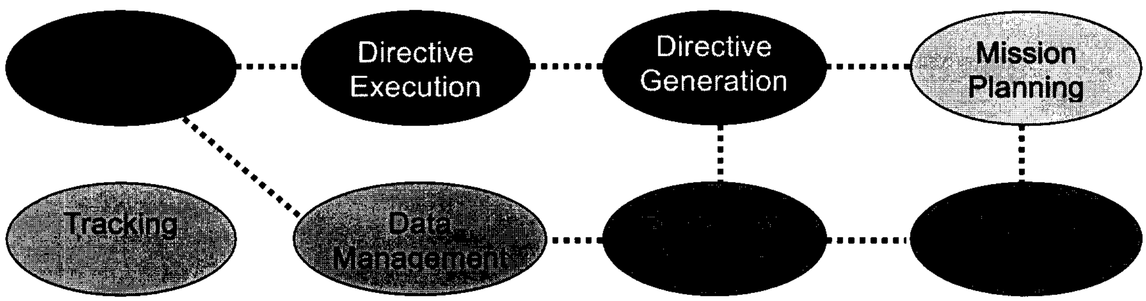

Figure 5. Example of Functional View (Mission A)

Functional Objects actually reside in physical entities (i.e., Nodes) of the system. Overlaying the Functional View on the Connectivity View of the same system will show the distribution of Functional Objects among Nodes. Such an example is shown in Figure 6, in which the Functional objects from Figure 5 are overlaid on the Connectivity View from Figure 3. The allocation of Functional Objects to Nodes is a part of the system design trade space.

INFORMATION VIEW

The motivation for the Information View is to clarify relationships among data objects that are passed among the functional elements, and to define their structures, relationships, and policies. Data Objects are managed (that is, stored, located, accessed, and distributed) by information infrastructure elements. The Information View is used to address these aspects of space data systems.

The Information View describes the space data systems from the perspective of the Information Objects that are exchanged among the Functional Objects. It includes descriptions of Information Objects (their structure and syntax), information about the meaning and use of these Objects (contents and semantics), the relationships among Objects, rules for their use and transformation, and policies on access. It also provides descriptions of the Distributed Information Infrastructure

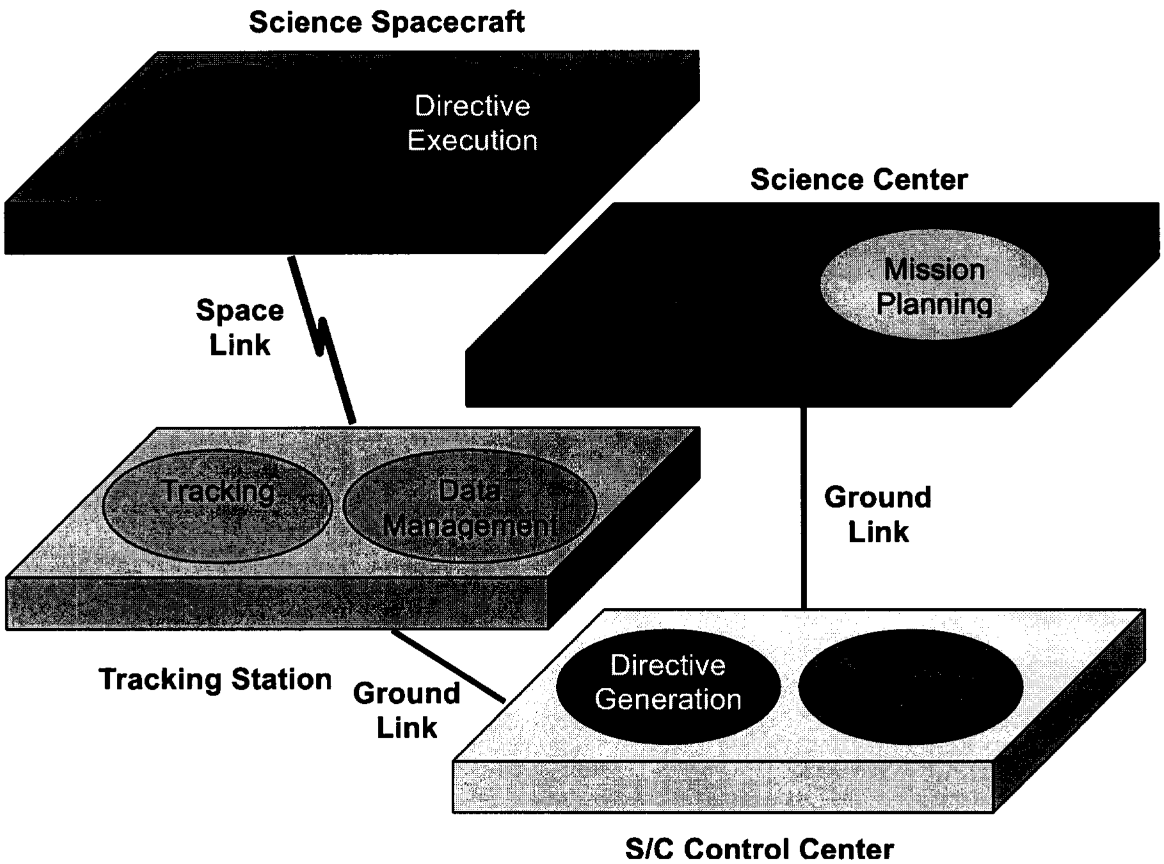

Figure 6. Example of Functional and Connectivity Views (Mission A)

(DII) that supports the location, access, delivery, and management of these Information Objects and descriptions of the Information Management Functional Objects that support the operations of DII. Finally, this View shows the relationship between the Information Objects and the Functional Objects that manipulate and exchange them.

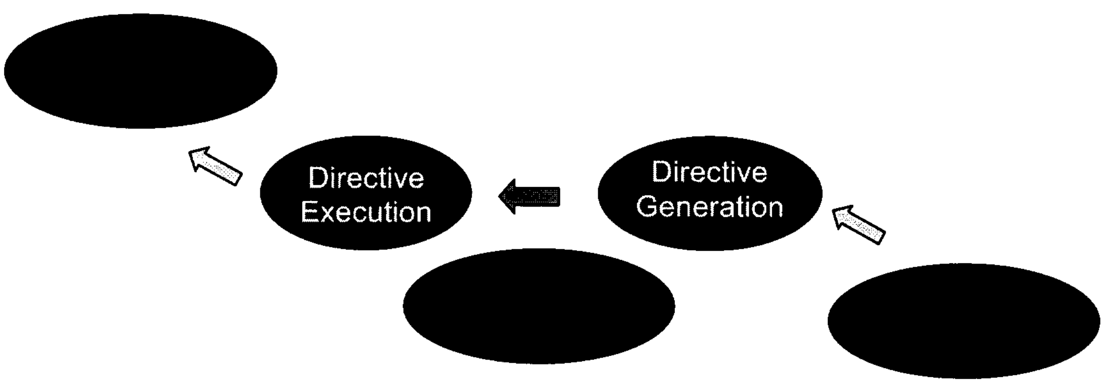

Figure 7. Example of Information and Functional Views (Mission A)

Figure 7 shows the relationship between some typical Functional Objects and the Information Objects that they exchange. This example shows a mission planning flow for Mission A, where the green objects are Functional Objects and the blue objects are Information Objects.

COMMUNICATIONS VIEW

The motivation for the Communications View is to define the layered sets of communications protocols that support communications among the functional elements. These protocols, and the Communications Objects that implement them, are needed to meet the requirements imposed by the connectivity and operational challenges. The Communications View describes the engineering solutions to these space data systems challenges and is a key area of technical focus within CCSDS. The Connectivity View describes the operating environment and the physical connections among Nodes and links.

The Communications View describes the mechanisms for information transfer among physical entities (i.e., Nodes) in a space data system. The Communications View is depicted as a set of Communications Objects and interactions among them. A Communications Object is an abstract model of a communications protocol that may be realized as either software or hardware. Communications Objects support information transfer between or among Functional Objects over Links (i.e., physical connections between or among Nodes). A stack of Communications Objects is usually used to support information transfer from a Functional Object to another Functional Object for a sequence of functional interactions. In the communications stack, the topmost Communications Object directly supports the Functional Object, and the lowest Communications Object handles the Link.

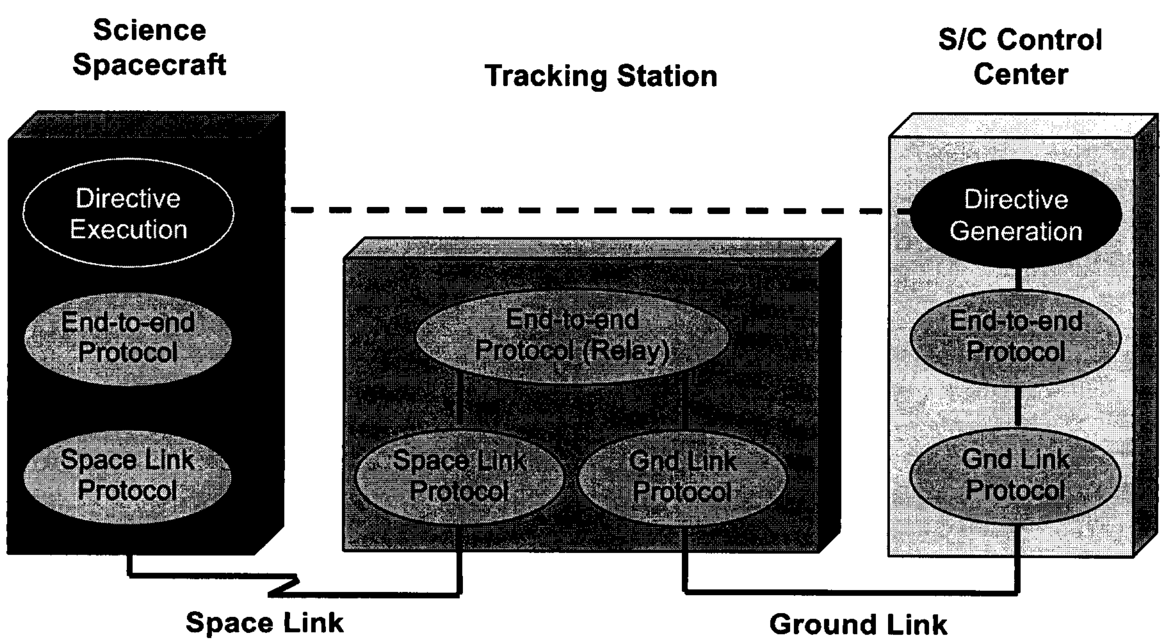

Figure 8. Example of Communication, Functional and Connectivity Views (Mission A)

The selection of Communications Objects to support information transfer between Functional Objects over a Link heavily depends on the characteristics of the Functional Objects, the Nodes, the physical Link and the space environment. Therefore, it is useful to show the Functional Objects, the Nodes and the Link together with the Communications Objects in the Communications View.

Such an example is shown in Figure 8, in which the Communications View (Communications Objects) are overlaid with a simplified Functional View (Functional Objects) and the Connectivity View (Nodes and Links).

CONCLUSION

This paper has briefly presented the Reference Architecture for Space Data Systems (RASDS) that is being developed by the CCSDS System Architecture Working Group (SAWG). The SAWG generated some sample architectures (spacecraft onboard architectures, space link architectures, cross-support architectures) using this RASDS approach, and RASDS was proven to be a powerful tool for describing and relating different space data system architectures.

In order to enable sharing and exchange of information on architectures of space data systems among different organizations or teams, we plan to develop formal methods for describing these architectures (for example, UML profiles and/or XML schemas). With these methods, each View of a space data system will formally described with the Objects contained in the View and the interactions among the Objects. The properties of Objects, their behaviors, and their interactions will also be formally described. In order to facilitate generation and manipulation of architectures, we plan to develop software tools, based on existing commercial or academic tools, for generation and manipulation of architectures.

Many aspects of space data systems that are considered in the RASDS have not been addressed in this brief paper, but are covered in the full report of the SAWG. These include security, system management, engineering details, lifecycle issues, IV&V, and other aspects of designing and building real systems. This Reference Architecture offers a consistent way of dealing with a variety of critical system viewpoints, starting with high level abstractions and working toward more concrete realizations and implementations.

ACKNOWLEDGMENTS

The authors wish to thank the members of the CCSDS System Architecture Working Group (Eduardo Bergamini, Fred Brosi, Adrian Hooke, Lou Reich and Don Sawyer) for their valuable contribution to the development of the RASDS presented in this paper.

REFERENCES

[1] http://www.ccsds.org

[2] "Information Technology - Open Distributed Processing - Reference model: Overview," International Standard, ISO/IEC 10746-1, December 1998