Escola Tècnica Superior d'Enginyeria de Telecomunicació de Barcelona
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Architectural Model for Evaluating Space Communication Networks

Master Thesis

Degree: Telecommunications Engineering Degree

Author: Marc Sánchez Net

Thesis supervisor: Edward F. Crawley, Bruce Cameron, Daniel Selva

Department: MIT Department of Aeronautics and Astronautics

Year: December 2012

Table of Contents

Table of Figures

Table of Tables

Abstract

The space exploration endeavor started in 1957 with the launch and operation of the first manmade satellite, the URSS Sputnik 1. Since then, multiple space programs have been developed, pushing the limits of technology and science but foremost unveiling the mysteries of the universe. In all these cases, the need for flexible and reliable communication systems has been primordial, allowing the return of collected science data and, when necessary, ensuring the well-being and safety of astronauts. To that end, multiple space communication networks have been globally deployed, be it through geographically distributed ground assets or through space relay satellites.

Until now most of these systems have relied upon mature technology standards that have been adapted to the specific needs of particular missions and customers. Nevertheless, current trends in the space programs suggest that a shift of paradigm is needed: an Internet-like space network would increase the capacity and reliability of an interplanetary network while dramatically reducing its overall costs. In this context, the System Architecting Paradigm can be a good starting point. Through its formal decomposition of the system, it can help determine the architecturally distinguishing decisions and identify potential areas of commonality and cost reduction.

This thesis presents a general framework to evaluate space communication relay systems for the near Earth domain. It indicates the sources of complexity in the modeling process, and discusses the validity and appropriateness of past approaches to the problem. In particular, it proposes a discussion of current models vis-à-vis the System Architecting Paradigm and how they fit into tradespace exploration studies.

Next, the thesis introduces a computational performance model for the analysis and fast simulation of space relay satellite systems. The tool takes advantage of a specifically built-in rule-based expert system for storing the constitutive elements of the architecture and perform logical interactions between them. Analogously, it uses numerical models to assess the network topology over a given timeframe, perform physical layer computations and calculate plausible schedules for the overall system. In particular, it presents a newly developed heuristic scheduler that guarantees prioritization of specific missions and services while ensuring manageable computational times.

Acknowledgments

First of all I would like to thank my friend and guide Daniel Selva for giving me the opportunity to work with him in his MIT experience. His dedication to my (now ours) work has been completely altruistic and at the same time invaluable. His insights on the system architecting and rule-based expert systems have been the basis upon which this thesis has laid its foundations.

I am also sincerely thankful to Bruce Cameron, Prof. Ed Crawley and Prof. Eytan Modiano. They have welcomed me in the MIT System Architecting Lab and have had full confidence on my work since the first day. It has been an incredible experience to share both personal and professional experiences with them.

My friends and colleagues at MIT and Boston have also been fundamental for the success of this effort. Ramon Martinez, Dolores Blanco and Antonio Hernandez deserve a special mention. Thank you for sharing this experience with me and helping me to make the most of it. Alessandro Golkar, Peter Davison, Alexander Rudat, Morgan Dwyer, Emily Calandrelli and Jonathan Battat have also enlightened these last months. It has been great working with all of them.

Last but not least, I would like to thank my mother for her incredible support. She has encouraged and helped me since the very beginning.

1. Introduction

1.1. Motivation

In summer 2006 the NASA Administrator created the Space Communication and Navigation (SCaN) program in order to centralize the management, operations and research on the present and future of NASA's space communication infrastructure. It was initially conceived as part of the Constellation Program that would return mankind to the Moon by year 2020. Despite this program no longer being under development, the SCaN taskforce has continued its activities to enhance NASA's space communication capabilities while trying to reduce its costs.

Reference [1] presents the baseline architecture and roadmap for the evolution of NASA's space communication network until 2025. The underlying assumption is that the current capabilities need to be upgraded and integrated into a single network that provides seamless communications between a spacecraft and the Earth both in the near-Earth and deep space domain. Enhancements like a lunar and a Mars relay constellation are considered key enablers to foster the robotic and human exploration activities on these planets. Similarly, [2], [3] provide a somewhat futuristic view of how a global solar system network would work. They introduce the idea of placing multiple relay satellites at the Sun-Earth and Jupiter-Earth Lagrange points in order to dramatically increase the return data rates from distant locations.

Although the vision of a global network similar to the Internet across the solar system is certainly appealing, the path to progressively evolve NASA's space communication network must be realistic and leverage at least two factors: what is the expected demand for the network over the next years; what is feasible with the available technical and economical resources. Reference [3] indicates that the demand for a space communication network will grow both in capacity and locations. Expected data rates for the Earth vicinity in the 2020 period are expected to be at least ten times greater than that of the current systems (10 Gbps instead of 1 Gbps). New requirements will also be imposed by missions to the Moon and Mars with data rates of up to 1 Gbps and 150 Mbps respectively.

Accommodating these increasing data rates will entail investing in new affordable technologies. In this context, space optical communications offer a promising technical solution that not only increases the achievable data rates but also reduces the mass and power of the communication payloads [4]. Similarly, reducing the lifecycle cost of the network can be done through new contracting modalities both on the manufacturing and operating phases. As an example, using hosted payloads as backup nodes on the network can increase the capacity and reliability of the system while minimizing the cost of operating the spacecraft.

Architecting the future of NASA's space communication network is a complex process. Numerous architectures can be envisioned and there is no proxy to understand how well they satisfy stakeholder needs. This thesis presents a framework and computational tool to support such process and promote tradeoff and sensitivity analyses. It also gives insight into what decisions are important and what degree of fidelity is required to successfully capture an architecture performance and cost.

1.2. Background

This section provides the background upon which the tool was developed. It focuses on three main areas of interest: the systems architecting paradigm; the rule-based systems and its applications; and the legacy and state-of-the-art space network architectures. It provides the basis to sustain the assumptions and limitations of the model, as well as to understand its scoping decisions.

1.2.1. The Systems Architecting Paradigm

Although the Systems Architecting Paradigm is a relatively new (late 80's) discipline with origins in civil engineering, its application to other fields has quickly proven successful. As an example, both the aerospace and communication industries have embraced its methodologies and have applied them in the earliest phases of the design of complex systems [5].

The architecture of a system can be viewed as its highest level design [6]. Once fixed, all subsequent decisions will be partially determined or constraint by its characteristics and guidelines. Since the system to design is complex in nature, three main effects may arise. First, the ambiguity and uncertainty associated with the system and its environment may render traditional engineering methodologies too specific to tackle the problem. Second, constraints and directives on the system may come from non-technical factors that cannot be well modeled using analytical and numerical models. Third, the span of plausible alternatives for the system may be very large (from thousands to millions of architectures), thus invalidating detailed and time intensive design methodologies.

In this scenario, using a computer-based tool can provide enough flexibility and computational power to effectively explore a large set of architectures. However, the question of how to evaluate one of them remains and must be addressed for each particular problem. In order to do so, three pieces of information are needed:

- How do we define a good architecture?
- What are the architecturally distinguishing decisions?
- How are the architecturally distinguishing decisions coupled?

The first question can be answered through the systems architecture methodology developed by Crawley [7]. In it, a system is composed by both a form and a function. The form represents what the system is, its physical or informational representation. It is the sum of its elements and their structure. On the other hand, the function is what the system does to an external subject. The mapping between the elements of form and function constitutes the concept.

Therefore, the architecture becomes *the embodiment of a concept: the allocation of physical/informational function to elements of form, and the definition of interfaces among them and with the surrounding context* [7].

In this context, understanding the definition of a good architecture implies determining the sources of associated benefit and cost. Crawley argues that the benefit is delivered to an external party through the function of the system while the cost arises from its form. Therefore, the value of the architecture can be measured as the benefit perceived by this external subject given its cost.

For instance, let's apply the system architecting methodology to a motorcycle. The main function of system is to transport its operator, the external party, from an origin to a destination. Thus, the benefit of the system arises from its ability to successfully perform that function. On the other hand, the form of the system is the sum of its physical constitutive parts and how they are structured: two wheels, an engine and a seat mounted around a steel frame. The cost of the system is equal to the sum of buying this parts and performing the manufacturing process to assemble them. Finally, the value of the system towards its operator is defined as the benefit given the cost, that is, *how well* the motorcycle can transport him from point A to point B given the economic cost of buying and maintaining it. Qualitatively and/or quantitatively defining *how well* depends primarily on the owners subjective expectations with respect to the system. If he intends to use the motorcycle to avoid the traffic delays of a city he will require a motorcycle that is small, maneuverable, with a limited range and as cheap as possible. However, if his aim is to take on long journeys then he will probably look for a large comfortable motorcycle and will probably be willing to pay a higher price.

The second question, *how to identify the architecturally distinguishing decisions*, has two main purposes: first, it reduces the perceived complexity of the system by understanding which parameters have a significant effect on its performance and cost; second, it scopes the model so as to ensure that only important decisions are being captured and explored. In order to do so, the system architect begins with a set of candidate decisions and then engages in an iterative process to analyze them until only the architecturally distinguishing are left. Both the initial set of candidate decisions and the iterative process rely primarily on the system architect's experience and prior knowledge of the system. If this is insufficient, then the system architect must take advantage of what Maier and Rechtin define as heuristics [5], that is, abstractions of lessons learned that condense and codify the practical experience from passed complex problems.

Examples of these heuristics can be found in Selva's System Architecting Problems [6], a collection of five typically encountered problems while architecting a system. Although his formulation is purely mathematical and therefore solution neutral, the realization that a complex system can be analyzed as a particular case of one of them can lead to an optimized formulation of what decisions should be included and left out.

Finally, the last question to address is whether the architecturally distinguishing decisions are coupled or not. This is important because it adds a new layer of complexity to the system. If two decisions are coupled they will need to be explored together in order to successfully capture their correlation. This may lead to two undesired outcomes: more complexity in explaining the results of the analysis, because it is not always easy to keep track of what decision caused a particular effect on the system; and the computational time required to analyze a set of architectures may increase dramatically rendering the model computationally inefficient.

The System Architecting Paradigm primary contributions to the modeling process of a complex system are as threefold. First, it helps to sequentially decompose the system's parts and distinguish between their function and form. Second, it defines a process for setting the system's boundaries and the associated scoping decisions of the model. But foremost, it indicates the need for developing value-centered models as opposed to performance or cost based models, i.e., tools that have enough fidelity to effectively capture the value delivered by the architecture to its stakeholders instead of focusing on computing the optimal parameters of the system's design.

1.2.2. Rule-Based Expert Systems

The intent of this section is to provide a first approach to rule-based expert systems (RBES). It will first present the RBES from a formal point of view, briefly describing its main parts and how they interact. It will then move on to justify the usefulness of RBES in the development of computational tools to aid the system architecting process. Finally, it will lay out a general methodology to efficiently develop value centered models from a RBES perspective.

1.2.2.1. Definitions and Structure of a Rule-Based Expert System

A rule-based expert system is a program designed to "mimic the reasoning of human experts in solving a knowledge intensive problem" [8]. Solving such problems typically includes (1) gathering the expert knowledge, (2) storing it in a comprehensible and manageable database and (3) applying it to the specifics of the problem. Rule-based systems facilitate steps (1) and (2) by encoding the information as sets of independent rules that, when combined, indicate what to conclude. Similarly, RBES implement step (3) by means of an efficient algorithm that autonomously applies the rules to find a plausible solution.

A rule-based expert system is generically composed of three main elements:

- A set of facts. A fact is a piece of data that contains information about the system. Initial facts encode the knowledge that the user has on the current state of the system. This facts can then be modified, lead to new facts or destroyed (retracted in the RBES jargon) to capture the evolution of the system.
- A set of rules. A rule is the logical form to encode the knowledge that the user has on how the system should evolve. It is composed of a set of conditions (the left-hand side

LHS) and a set of actions (the right-hand side RHS) that must happen if the former are met.

- An inference engine. It is responsible for efficiently performing the pattern-matching between the facts and LHS of the rules. State-of-the-art rule-based expert systems use the *Rete Algorithm* as the heart of its inference engine. It was first introduced in 1982 by Charles Forgy [9] and organizes the set of rules as a net of conditions, each link representing a LHS of a rule. Facts percolate through this network automatically performing the pattern-matching process, therefore identifying what rules are fully satisfied and can be fired.

The typical structure of a fact is as follows:

(fact (slot1 value1) (slot2 value2) ... (slotN valueN) (multislot1 list1) ... (multislotM listM))

Each slot is equivalent to an attribute in an object-oriented class, it stores one particular piece of information about the system. In turn, a multislot is comparable to a one-dimensional array that contains an ordered list of data. Both slots and multislots can store any type of information regardless of their nature (integer, float, boolean, string).

On the other hand, a rule has generally the following structure:

if *(?f1* ***<-*** *condition 1) (?f2 <- condition 2) ... (?fN <- condition N)* ***then***

(action1) (action2) ... (action M)

In the LHS of the rule a set of N conditions are checked against the database of facts. The operator <- indicates that any fact matching that condition will be stored in the variable *?fi* for later use. The rule fires (i.e. executes the RHS) only if all the conditions in the LHS are met unless the operator *or* is explicitly used.

1.2.2.2. Using Rule-Based Expert Systems on the Systems Architecting Paradigm

Traditional programming languages (C, C++, Java) are imperative, that is, the developer has to specify the exact sequence of events or commands that must be executed in order to produce a meaningful result. This approach is very well suited for engineering problems where numerical models have been developed and tested. Numerous examples exist in almost every engineering field, from tools to design ships (FORAN) to programs that model thermodynamic (Solkane) or biochemical (Gepasi) systems.

The usage of these imperative languages in the Systems Architecting Paradigm is possible and has been proven to be effective assisting what Maier and Rechtin identified as *the prescriptive method* and the *rational method* for systems architecting [5]. The first conduct the architecting process by means of handbooks, regulations, civil codes that collect a set of procedures of how to successfully design the system. These procedures can be translated to programming code and, given the correct inputs, will prescribe the appropriate architecture. Similarly, the *rational* *method* is method based, that is, it builds upon analytical models from the engineering sciences. As such, these models can be easily programmed and will take advantage of the computational power provided by state-of-the-art computers.

However, the applicability of imperative languages to heuristic based system architecting methodologies is limited. Not only do they lack flexibility to capture general fuzzy knowledge, but they are also stringent in the ordering and evaluation of this knowledge during the system architecting process. In other words, since an imperative programming language is a set of instructions and procedures that are sequentially executed, a system architecting tool based on it will always have a fixed prescription of the architecting process structure.

To avoid that, one should consider programming tools that do not focus on the sequence of events take place while architecting a system but rather focus on properly capturing the knowledge that the system architect has. At this point, the appropriateness of logical programming languages and particularly rule-based expert systems is unquestionable. Some of the advantages they offer compared to traditional imperative programming languages include:

- Easiness to encode expert knowledge and reuse it on similar problems. System architects usually rely on past experience and lessons learned as a way to delimit the scope and the uncertainty of the system. Encoding this knowledge as separate independent entities (the rules) serves as a progressively growing database.
- Adaptability to non numerical and qualitative metrics. The definition of the architecture is usually driven by both technical and non technical factors. While the first can be easily quantified, the second tend to be better expressed by means of qualitative adjectives and fuzzy logic.
- Consistency of the obtained results. Rule-based expert systems are deterministic, i.e., for a given set of initial facts and rules the output will always be the same. This allows using rule-based expert systems inside optimizing algorithms.
- Understandability of the results. The information about the system design or state is encoded as facts on the working memory. Retrieving this information is easy and allows building explanation facilities.

Nevertheless, rule-based expert systems also have certain disadvantages that must be noted before using them. Among them the most important are:

- Lack of understandability in the results. The developer has limited control over the order in which rules are activated and fired. As a result, in a growing body of knowledge it is progressively difficult to understand the effect of one rule on the system. This hinders the debugging process of the program as well as the traceability and validation of how a result was obtained.
- Problems in computational performance. Rule-based expert systems can be very inefficient for some numerical computations. If part of the system architecting process has intensive processing requirements, RBES will typically underperform compared to traditional programming languages.

1.2.2.3. Value Assessment of System Architecting Using Rules (VASSAR)

In the previous section the idea of architecting a system by means of rule-based system was introduced. This section presents VASSAR, a general framework to assess the value of an architecture through sets of expert knowledge. A detailed description of the framework and its constitutive part can be found in [6]. This section will only discuss the generalities of the process due to their relevance to this thesis. It will also provide a paradigmatic example so as to facilitate its understandability.

The main idea behind VASSAR is that the value delivered by an architecture to its stakeholders can be computed by means of a four step process:

- Model the architecture at the system, subsystem and component level.
- Compute the architecture capabilities or performance.
- Compare the architecture capabilities with the requirements imposed by the stakeholders to assess their satisfaction.
- Aggregate the stakeholder satisfactions to a single metric that indicates the value of the architecture.

Since the framework was specially designed to work on rule-based systems, the following description assumes that the reader is familiar with the concept of both facts and rules.

As previously stated, the first step to model an architecture with VASSAR is to define its characteristics at the system, subsystem and component level. One would start by asserting a set of facts that describe the architecture at the system level. The information contained in these facts is an input to the model and depends exclusively on how the user defines the architecture. Next, the subsystem facts will be introduced and their attributes will be populated by means of the *attribute inheritance rules*. They will specify where the information of those attributes is stored, be it in the architecture fact itself or in other fact databases with generic knowledge about the system. Finally, the component facts will be inserted and populated in a similar manner.

As an example, let's consider the architecture of an electrical grid. Suppose it is defined by a group of power plants with given nominal capacity and location on a territory. Then, the system level of the architecture will be defined by a set of facts "power plant" and "electrical line" that indicate the physical structure and interconnections of the grid. At the same time, the subsystem and component level will include all constitutive elements of a power plant: The generation subsystem, with the electrical generators as components; the thermodynamic subsystem, with the boiler, the condenser and the turbine as components; and so on.

Once the architecture has been defined, the model will be ready to compute the system's performance or capabilities. This process is done through *capability rules* that encode how the system can perform given its design parameters. The general structure of these rules includes a LHS that looks for a set of component attributes and a RHS that asserts a new capability fact from their combination. Once the capabilities have been computed they can then produce new capabilities by means of the *emergence rules*. These contain information regarding advantageous combinations among capabilities and therefore allow capturing synergistic behaviors on the system.

Back to the example, the performance of the system will be measured by the number of households that can be connected to the system and the probability that one of these households is left without service. Therefore, a typical *capability rule* would say: if a power plant has a nominal capacity of 200 MW and a generic household requires 4.400 W then the system is capable of supplying energy to a maximum of 45.455 households. Similarly, an *emergence rule* would state: if a power plant is *capable* of having a 1% probability of failure and a household is connected to two power plants, then the probability of a user being without service is $1\% \cdot 1\% = 0.01\%$.

The third step on the VASSAR methodology is to compare the capabilities of the system with the requirements demanded by its stakeholders. The *requirement satisfaction rules* are used to this purpose. They encode, for each stakeholder, its subjective perception of how well the capabilities of the system meet its expectations. In other words, they capture the subjective needs of an stakeholder by means of a set of objective requirements. As a result, they can be used to measure the benefit that a stakeholder can withdraw from the system given its performance.

In the electrical grid example the *requirement satisfaction rules* will be related to the quality of service that a household expects from its electrical provider. For instance, a rule would say: if a household is left without service for 5 days each year then its satisfaction is low. As previously stated, the rule contains two pieces of information: a stakeholder *needs* to have power more than 360 days to have a high degree of satisfaction (subjective value assessment); the *requirement* of the stakeholder can be expressed through the number of shortage days (objective metric). Note also that the requirement is expressed in terms of shortage days as opposed to percentage of unattended time. In this case, the *capability rules* would be responsible for transforming the percentage of unattended time to shortage days while the *requirement satisfaction rules* would strictly account for the comparison between capabilities and requirements.

Finally, the last step of the VASSAR methodology is aggregating the satisfaction of all the requirements among all the stakeholders. In order to do so, information regarding the relative importance of a requirement with respect to a particular stakeholder must be available. Then, the *value aggregation rules* will compute the satisfaction of a stakeholder as the weighted sum of its requirement's satisfaction. In fact, this procedure can be extended by allowing the user to specify objectives and subobjectives between the stakeholder and requirement level. Granting this feature increases the understandability of the results by grouping requirements according to logical categories.

A typical structure for the value aggregation step in the electrical grid example would include specifying two stakeholders: households and special users. The first would comprise regular houses while the second would contain users that demand better quality of service. One can also envision dividing the special users into two objectives: premium clients, i.e. those that want better quality of service because they pay higher rates; and extraordinary clients, which require better service due to the activity they carry out (e.g. hospitals, data centers or banks). Note also that the relative importance between stakeholders and between objectives is an input to VASSAR.

All in all, using VASSAR as a reference framework for designing system architecting models and tools implies that:

- The model is value-centered and provides, for each architecture, information regarding the satisfaction of each stakeholder.
- The list of stakeholders, objectives, subobjectives and requirements is available from the beginning. The relative importance for all of them is also an input to the model.
- The value of the system is determined by comparing the capabilities of the system to requirements of the stakeholders.
- Computing the capabilities of the system can be done by means of a set of rules that approximate the model's performance.

1.2.3. Satellite Communication Network Architectures

This section will provide an introduction to satellite communication networks from a physical and technological perspective. It will first describe the types of constellations that are typically considered along with a qualitative assessment of their advantages and weaknesses. Second, it will discuss the differences between multiple payload architectures at each node of the network. Third, it will introduce the differences in the access mechanisms that can be used to share a common link between multiple spacecraft. And finally, it will present a first description of the ground segment and its interactions with both the spacecraft and the network users.

1.2.3.1. Satellite Constellations and Orbits

A satellite communication network can be defined as "the arrangement, or configuration, of satellites and ground stations in the a space system, and the network of communication links that transfers information between them" [10]. It is generally decomposed into two main parts: the space segment and the ground segment.

The space segment is composed of a set of communication satellites flying in a constellation pattern. Next, a categorization of the space segment will be presented according to its constellation characteristics. Most of the assertions herein presented can be found in [10].

Constellation type	Advantages	Disadvantages

Geostationary Orbit	• Very few satellites can provide almost full coverage (3 may suffice).
	• No need to track the ground station antenna.
	• The antennas require small FOV[1], typically less than 15º.
	• There is little need for handovers between satellites.
	• Long access durations, typically around 20 minutes.
	• Reduced network of ground stations even if no intersatellite links are provided.
		• High cost on the launching phase.
		• Each satellite typically carries multiple payloads. As a result each spacecraft has:
		  • Increased mass, volume, complexity and cost.
		  • Increased programmatical risk due to launch/operation failure.
		• No coverage of Polar Regions.
		• Limited available slots above areas of interest (continents).
		• Long propagation delay, typically 250ms. This may limit the services supported.

Low Earth Orbit	• Low cost on the launching phase.
	• Satellites are typically designed to offer a specific service. As a result each spacecraft has:
	  • Few communication payloads
	  • Low to moderate mass, volume, complexity and cost
	  • Low to moderate programmatical risk due to launch/operation failure.
	• Coverage on polar areas through inclined orbits.
	• Low propagation delay.
	• Less demanding power subsystem due to shorter link distances.
		• Complex antenna tracking and acquisition systems.
		• Large number of satellites to provide continuous coverage.
		• Extensive network of ground stations if no intersatellite links are available.
		• Complex network control, especially if intersatellite links are provided.
		• Short access durations, typically no longer than 12 minutes.

Medium Earth Orbit	• Reduced number of satellites compared to the LEO case. • Medium cost per satellite. • Increased footprint area	• Increased cost on the launching phase. • Handovers between satellites are required to

[1] The Field Of View (FOV) is defined as the conic angle between the spacecraft nadir and maximum direction towards which the antenna can be steered.

	compared to LEO satellites. • Medium access times. • Relatively modest requirements on the power subsystem of the spacecraft. • Can provide coverage of the Polar Regions with inclined orbits.	provide continuous contact. • Complex antenna tracking and acquisition systems.
Molniya Orbit	• Provides coverage of the Polar Regions. • Low cost on the launching phase. • Long access times depending on the eccentricity of the orbits.	• Require more than one satellite to cover a hemisphere. • Complex antenna tracking and acquisition systems.

Table 1. Orbit comparison for space communication networks

The decision of what is the most appropriate orbit and constellation design for a satellite communication network is not a standalone problem. At least three other architectural parameters must be leveraged: first, what kind of payload and technology will be available on the spacecraft; second, what type of end terminal will be connecting to the network; and third, what kind of services will be offered.

1.2.3.2. Satellite payload architectures

The type of communication payload that each relay satellite carries has major implications in the design and cost of both the spacecraft and the ground stations. Three main architectures can be envisioned:

- Bent-pipe payloads: Relay satellites operate as a mirror that reflects the signal from the source to the destination. The general structure of the payload consists of a RF front-end with an antenna, a low noise amplifier (LNA) and a mixer to down convert the signal to an intermediate frequency (IF). Next, the signal is filtered, frequency-translated, power-amplified and routed to the antenna that will retransmit the signal to the destination. Since the signal is never demodulated onboard the spacecraft the routing information of the network protocol cannot be recovered and used to automatically decide the next hop. Being that the case, the decision of when and where to point the antennas and spacecraft in order to achieve successful relaying capabilities is either pre-established and fixed or is scheduled in advance and transmitted to the satellite beforehand.
- Onboard processing (OBP, also known as regenerative) circuit-switched payloads: The satellite operates similarly to the bent-pipe architecture but is now able to perform some degree of data processing. Advantages of this decision choice include the

possibility of demodulating, error correcting and remodulating the signal at each hop, a strategy that has been proven to increase the link performance between 3 and 5 dB [11]. If the onboard processor can handle routing algorithms, this architecture might also eliminate the need for scheduling customers in advance and therefore promotes autonomy in the network.

- OBP packet-switched payloads: The satellite payload emulates the behavior of an Internet node. As such, it is able to extract the transport and network information of each packet and find the next hop to the destination. References [12], [13] prove that packet-switched architectures offer an increased capacity of the system thanks to the possibility of statistically multiplexing the incoming bit streams. Similarly, [14] demonstrates that the flexibility of a packet-switched network in front of planned and unplanned changes in the network is better than that of circuit-switched and bent-pipe architectures. Nevertheless, the implementation of these kind of architectures is still in its early stages, with most commercial providers considering the technology too complex, expensive and risky to fully develop its potential [15].

Selecting the type of payload in the relay satellites has a direct impact on the network's topology. Bent-pipe architectures are well suited for hub-spoke topologies, that is, there is a high speed trunk line between the ground stations and the relay satellite and signals are disseminated to and received from all the users. Examples of these architectures include NASA's Tracking and Data Relay Satellite System (TDRSS) or the Globalstar constellation.

On the other hand, onboard processing architectures allow meshed networks where satellites communicate with ground stations and between them. Including inter-satellite links increases the overall capacity of the system by providing multiple paths between a source and a destination. Ideally, if a link is congested the routing algorithm can determine an alternative. It also permits reducing the number of ground stations since there is no need for maintaining physical line of sight from a ground station to all the spacecraft.

1.2.3.3. Connecting to the Network of Relay Satellites

The main purpose of a communication network is to allow information transfer between a set of transmitters and receivers. In order to correctly perform this function, the network has to provide an easy entry mechanism that ensures that any potential user will be able to properly connect to it. Since the network under consideration in this thesis is intended to relay information from near-Earth orbiting satellites to the Earth surface (and vice versa), the potential users of the network are spacecraft and their operators.

In order to define what type of access is being provided to the customer, each node on the network must specify at least two parameters: what part of the spectrum is being used; and what type of multiple access is necessary. The first decision is usually driven by the amount of data rate that is required in the connection: the higher the data rate, the higher the frequency needed to support the link. To justify this assertion two pieces of information are needed:

- The amount of bandwidth required to obtain a certain data rate is directly related to it. In other words, if a customer requires higher data rates then it will always necessitate more bandwidth to achieve a successful transmission. Understanding the details of this can be found in [10], [11].
- The frequency and amount of bandwidth that a certain user can exploit to transmit its signal is highly regulated by international organizations [16]. These organizations have partitioned the available spectrum in multiple correlative slots in order to avoid interferences among systems. This partition always allocates more bandwidth at higher frequencies.

Therefore, if a given user requires high data rates it will necessitate greater available bandwidth which in turn will force him to use a higher frequency band. In fact, [17–19] are just some examples that justify using optical (10^{14} Hz) as a key enabler to high data rates in the space communication industry.

The second parameter that a node has to specify in order to understand the network-customer interface is the multiple access mechanism being used. This mechanism will ensure that when multiple users transmit at the same time through a shared channel they do not interfere with each other. Since the frequency and amount of bandwidth that the node has available is fixed, a multiple access scheme will try to coordinate users so that they use this bandwidth in an orderly fashion. Four types of schemes are typically used (alone or combined):

- Frequency division multiple access (FDMA): the bandwidth is divided in channels and subchannels that are then assigned to different users. The maximum achievable data rate depends on how this channelization is done and how much bandwidth is assigned to each user.
- Time division multiple access (TDMA): All users can use the full bandwidth to transmit but at different time intervals. The maximum achievable data rate depends on how much time a user can transmit in a certain period of time. More users on the system imply less transmission time per user and therefore lower average data rates.
- Code division multiple access (CDMA): All users can use the full bandwidth during all the time. Their transmissions are separated through a set of unique sequences $\boldsymbol{c_i}$ that have two fundamental properties:

$$\begin{cases} \boldsymbol{c_i} \cdot \boldsymbol{c_i} = 1 \\ \boldsymbol{c_i} \cdot \boldsymbol{c_j} = 0 \ \forall i \neq j \end{cases}$$

  In order to share the channel, each transmitter encodes its messages through a fixed assigned sequence $\boldsymbol{c_i}$. Then, the receiver can separate the transmission of different users by multiplying the received signal by the same sequence $\boldsymbol{c_i}$. However, since all users are transmitting at the same time on the same bandwidth they are in fact interfering with each other. Thus, the maximum achievable data rate depends on what level of interference is acceptable on the receiver.
- Random Access Mechanisms: All users can use the full bandwidth during all the time regardless of the other users. If a collision (two users transmitting at once) occurs both transmissions are lost and users will attempt to retransmit their messages after a random amount of time.

The appropriateness and usefulness of these multiple access schemes is thoroughly discussed in [11]. Analytical results indicate that FDMA systems have better throughput when the users send high volumes of traffic in a small number of accesses (for instance, voice and video services). In turn, TDMA and CDMA are more efficient if the information comes in the form of small volumes and a high number of connections (e.g. internet services). The choice between TDMA and CDMA is usually made depending on technical complexity. It is known that TDMA can always outperform CDMA in a nominal scenario. However, in order to function properly TDMA requires a high degree of synchronization between users. This is not always easy to achieve, especially in the satellite domain where the relative motion of bodies and long distances can cause unpredictable time delays. Thus, there are currently multiple relay constellations that have adopted CDMA as their multiple access scheme [20], [21].

Finally, random access mechanisms are used in the satellite communication industry due to three main reasons: first, they are the simplest and cheapest to implement since there is practically no coordination among users. Second, they can have better performance than FDMA, TDMA and CDMA if the load of the network is very low. And third, they can be easily used in combination with the other multiple access schemes to obtain a demand assigned multiple access (DAMA), that is, a mechanism to assign resources (reserve bandwidth, a time slot or a sequence) to the user only when it needs them.

1.2.3.4. The ground and user segment

Let us consider a network of relay satellites that channelize information from orbiting satellites to their owners on the ground and vice versa. In order to do so, at least two strategies can be considered: connect the relay satellites to a set of dedicated facilities that act as hubs and then retransmit the information to the final destination; or send the information from the relay satellites directly to the user. Table 2 qualitatively summarizes the architectural differences between both options and how they affect the space and ground segment.

	Dedicated Facilities	Direct to the user
Space Segment	• Single access payload that communicates exclusively to the facility. • Data from multiple sources can be multiplexed and sent to the ground in a high speed trunk. • Possibility to use optical payloads on the downlink together with site diversity for higher reliability.	• Multiple access payload to provide connection to multiple users in its footprint area. • Possibility of using a multiple beam antenna with frequency reuse to increase the capacity of the system. • Optical communications are mature enough to be used in this context

Ground Segment	• Each facility contains a small number of big size antennas (5 to 30m).
	• Possibility to connect to deep space users thanks to the high receiving gain.
	• High data rate links can be established due to:
	  • High gain antenna
	  • High transmit power
	  • Tracking antennas
	• High cost associated with the facility operation.
	• The network operations center (NOC) can be integrated into an existing facility.
	• Political/Diplomatic burden due to facilities in foreign territories.
	• Need for a ground infrastructure to relay the information from the facility to the final user.
		• High number of small terminals[2] (<1 to 5m) owned by the user.
		• Limited range on the communication links.
		• Limited data rate links can be established due to:
		  • Limited gain antenna
		  • Limited transmitting power
		  • Limited tracking ability
		• No cost associated to the operation of the user related terminal.
		• The network operations center (NOC) requires a dedicated facility.
		• No need for a ground infrastructure to relay information from facilities to the users.

Table 2. Ground segment comparison for space communication networks

Choosing between a "dedicated facilities" approach or a "direct to user" approach depends primarily on the system requirements. For networks that have to relay large amounts of information at high data rates the first option has usually been adopted. Examples include NASA's Tracking Data Relay Satellite System with its facilities in White Sands (New Mexico, US) and Guam [22] or ESA's European Data Relay System with locations in Weilheim and Ottobrun (Germany), Redu (Belgium) and Harwell (UK) [23].

On the other hand, the "direct to user approach" has been traditionally more successful in the commercial fixed and mobile satellite service industries. The first uses VSAT terminals (typically parabolic antennas) to allow two-way communications at moderate speeds (tens of Mbps). The second uses portable low gain antennas that only allow low communication speeds (tens of kbps).

1.3. Thesis overview

Chapter 2 introduces past frameworks and tools developed for the design of space communication networks and space relay systems. It starts by making a comprehensive summary of past models for the network topology, schedule, traffic and performance. Then, it

[2] Usually referred to as Very Small Aperture Terminal (VSAT)

presents a discussion of these models vis-à-vis the System Architecting Paradigm in order to identify the research goals and state this thesis' objectives. Chapter 3 is entirely devoted to the description of the tool, focusing the attention to the performance model. It starts by describing the generic modeling elements, it then presents its overall high level structure and it finally describes each module in detail.

2. Literature Review

2.1. Introduction

The engineering sciences originally relied on analytical models to explain the behavior of complex systems. Network performance was no exception and by 1917 Agner Krarup Erlang had already deduced his famous formulas Erlang-B and Erlang-C to size the capacity of the telephone network for a given blocking and queuing probability [24]. Nevertheless, his mathematical formulas and those that would be derived later only studied the performance of individual parts of the network under a precise set of assumptions. Therefore, end-to-end performance was difficult to assess even when the network size was static and small in size.

With the advent of the information technology era and the development of personal affordable computers, a new paradigm to study communication networks became real: simulation. One could program a set of nodes, define their interconnections and specify the stack of protocols to use when transmitting the information across the network. Then, a discrete event simulation engine would sequentially compute the state of the network at multiple time steps and store the status of the nodes and links. Finally, this information could be gathered and statistically analyzed in order to help network engineers understand the actual load of the network.

This section presents a summary of past and state-of-the art network simulators in the context of space communications. It first describes how to define the network topology, schedule and traffic flows in networks where nodes are orbiting around the Earth and therefore can only send and receive data when there is clear line of sight. Then, it presents a discussion of these models vis-à-vis the System Architecting Paradigm and identifies the research goals in this area.

2.2. Modeling the Network Dynamics

The first step when designing a network is the specification of its constitutive nodes and their interconnections. This process is usually known in the network engineering community as the network topology definition and can be easy or cumbersome depending on the degree of mobility between nodes. Four paradigmatic cases have been identified:

Backbone Nodes	Terminal Nodes	Topology Complexity	Examples
Fixed	Fixed	Low	Plain Old Telephone Service, Ethernet networks
Fixed	Mobile	Medium	Cellular networks such as T-mobile, Verizon, AT&T GEO Satellite networks for terrestrial users: Inmarsat

Mobile	Fixed	Medium	LEO/MEO Satellite networks for terrestrial users: Skybridge, Teledesic, Globalstar, Iridium
Mobile	Mobile	High	Satellite networks for spatial users: TDRSS[3], EDRS[4], DRTS[5]

Table 3. Topology complexity depending on network node mobility

In this table it is assumed that a node being "fix" or "mobile" is always done with respect to how the other nodes on the network behave. For instance, in a cellular network the base stations are usually fixed antennas placed in the roofs of city buildings while terminal nodes are users walking or driving on the streets. In this case, little confusion on mobile vs. fixed terminals is left. However, in a satellite network for terrestrial users both the satellite and the user could be moving: a LEO satellite orbiting around the Earth at 27000 km/h and a user in an airplane at 2500 km/h[6]. Since the first is moving at least one order of magnitude faster than the second, as a first approximation it can be assumed that the user behaves as a static node.

This thesis focuses on the architecting process of a satellite relay network for spacecraft operating in the near-Earth domain. Being that the case, it is clear the both the backbone elements of the network (the relay satellites) and the users (the customer satellites) will orbit around the Earth constantly changing the topology of the network. In order to account for that effect, a possible solution is to simulate the movement of all nodes over a representative period of time and capture the network topology at multiple time intervals.

Three main strategies can be envisioned so as to model the orbital mechanics of the network:

(1) Use a network simulator and program a set of libraries that model the behavior of the orbiting spacecraft. The resulting equations are treated by the discrete event engine of the network simulator in order to compute the position, velocity and acceleration of the nodes.
(2) Create a network simulator that has the orbital dynamics embedded. This option implies programming an orbital and network simulator from its foundations and therefore has greater flexibility on both its conception and implementation.
(3) Use an already built software that is optimized for orbital dynamics. The network architect can focus on properly modeling the behavior of the network and rely on the validity of the orbital mechanics simulation. Nevertheless, an interface between the orbital and network software has to be implemented.

References [25], [26] are examples of the first approach. [25] starts with a motivation for applying simulation tools in the design phases of satellite communication networks. It then describes a set of modules incorporated to the freely available simulator *ns* (now *ns-2*) [27] to model both the orbital mechanics of LEO satellite constellations and the physical layer of a communication system. In turn, [26] extends the work done in [25] by presenting the analytical

---

[3] NASA's relay satellite system: Tracking and Data Relay Satellite System
[4] ESA's relay satellite system: European Data Relay System
[5] JAXA's relay satellite system: Data Relay Test Satellite
[6] The speed of the airplane in a geo-centric frame reference is equal to the airplane speed with respect to the Earth surface (≈900 km/h approximately) plus the rotation speed of the Earth surface (≈1668 km/h).

formulae required to model multiple constellations flying at different altitudes. These are introduced in the *ns* simulator in order to perform a case study with a network containing LEO, MEO and GEO spacecraft.

The second strategy, *create a network simulator that has the orbital dynamics embedded*, is applied in [28], [29]. The first work describes a simulation environment in which there is a dedicated module devoted to the kinematics of the Earth and satellite rotations. Results presented demonstrate the ability of the tool to determine the contact windows between pairs of nodes of the network. On the other hand, [29] introduces a simulator built upon the strengths of object-oriented programs. Different nodes of the network can be defined by subclassing generic interfaces provided by the tool. Each node has a motion model associated, that is, a class that contains the parameters needed to describe its motion in an Earth-based reference framework. Specific classes for orbital dynamics are already provided by the software developers.

Finally, the third strategy focuses on using already available software that is optimized for modeling orbital mechanics and the physical layer of a communication network. It has been applied in multiple occasions [30–35], proving that it is currently the most popular choice. In all these cases, the software chosen by the authors is Satellite ToolKit (STK) [36], a comercial off-the-shelf analysis and simulation software for spatial systems. The network is modeled as a set of objects (facilities, satellites, vehicles) and STK autonomously computes their movement in a common reference frame. Moreover, sensors, transmitters and receivers can be appended to these objects in order to simulate the physical layer of the communication system. For instance, parameters such as the transmit power, modulation or antenna temperature can be inputted to compute the expected BER[7] of a radiofrequency link.

Choosing between strategies (1), (2) or (3) might not be always straightforward. Next, a comparative table is presented in order to clarify the advantages and disadvantages for each of them.

Strategy	Advantages	Disadvantages
Program libraries for a network simulator	• Easy and quick to implement. • The programmer can chose the desired level of fidelity while modeling the orbital dynamics. • Approximate analytical formulae can replace time expensive simulations.	• A discrete event engine is not optimal for orbital mechanics computations. • The programmer needs knowledge in orbital mechanics and the physical layer of a link. • High accurate models might be difficult to find.
Program a network and orbital simulator	• It is the most flexible approach. The model can be focused	• It is very time consuming since the program needs to

[7] Bit Error Rate

	specifically to the areas of most interest. • The tool can be designed so as to meet exactly the desired degree of fidelity.	be built from scratch. • It is hard to validate the results without a similar tool. • The programmer has to know details on both orbital mechanics and all the layers of a communication system.
Use a specifically designed software	• The user can understand the network topology visually. • The results obtained are easy to understand and need little validation. • Most software incorporates an interface library to facilitate the process of creating new models. • The user does not need to know the specifics of orbital mechanics. • Models for communication are already built in and have a high degree of fidelity.	• The user is limited to functionalities built in the software. • The user has to create an interface between the software and his program. • The user has to go through a learning process for the specifics of the software. • Most software of this type is not freely available.

Table 4. Strategies for simulating the network topology in space communication networks

2.3. Modeling the Network Scheduling Process

Having addressed the physical topology of the network, the next step is to understand what contacts between ground stations, satellites and users can be supported given the number of point-to-point links that each node can support and the visibility windows between them. This process is known as *the scheduling problem* [37].

The generic formulation of a scheduling problem includes (1) a data structure that defines what resources are available on the network at each moment in time and (2) a list of jobs that need to use part of those resources in order to be served. In the context of space relay data networks, (1) is usually a set of matrices that represent snapshots of the network topology at one time interval. Obtaining these matrices is generally done through the methods discussed in section 2.2. On the other hand, (2) typically consists of a list of users (i.e. customer satellites) that want to use the relay network to send data to the Earth.

Information regarding the number of users, jobs and particular requirements can be found in either *Mission Models* or design documents for particular missions. The first specify a set of paradigmatic missions that are to be expected over a certain time frame. [38] is an example of how a mission model would be structured. It categorizes missions according to their requirements in terms of data rates and times of contacts. It also gives a rough estimate of the number of users that each category would have and then computes the number of service hours and number of channels to support them. On the other hand, [39], [40] are two examples of design documents for particular missions. [39] is the preliminary design of the Wide-Field Infrared Survey Explorer (WISE) that was launched in 2009. It indicates that the observatory should use TDRSS to communicate with the Earth, with 4 nominal contacts per day that account for a maximum daily access time of 60 minutes. Similarly, [40] has information regarding the operations of the Earth observation satellite Terra. In particular, it states that the "nominal mode of operations is to acquire two-12 minute TDRSS contacts per orbits" using a Ku-band 150 Mbps link.

Once the input information for the scheduling problem has been defined, an appropriate strategy to tackle it must be selected. Reference [14] identifies three main approaches:

(1) Heuristic algorithms: They use a set of rules to create a possible schedule that has no guarantees to be optimal.
(2) Static algorithms: At each time interval users and jobs are scheduled through an optimizer that takes into account only the present state of the network.
(3) Dynamic algorithms: Users and jobs are scheduled through an optimizer that takes into account present and future states of the network.

Reference [42] presents a scheduling algorithm for the European Data Relay System based on approach (1). It uses state-of-the-art knowledge-based schedulers that follow two strategies: first, analyze the problem through a set of rules that foresee possible difficulties in the schedule generation process and help building a partial solution; second, evaluate this partial solution and implement methods that use it to compose new improved schedules. Alternatively, [37] introduces the results of a scheduling algorithm based on approach (2). It expresses the problem as constrained linear program with binary variables and then implements Munkres [43] and Gomory-Balinsky [44] algorithms to solve them. Finally, [41] exemplifies approach (3) by presenting a genetic algorithm that schedules users by trying to minimize the relative difference between the duration of a job and the contact windows among satellites. Note that this strategy takes into consideration the whole history of contacts between satellites and customers, as a complete list of these contact windows is assumed to be an input to the problem.

2.4. Modeling the Network Traffic

Modeling the network traffic is very dependent on the type of satellite communication network under consideration. In this case, choosing the best approach to tackle the problem depends primarily on the type of connectivity that will be established between a relay satellite and the users. Two main configurations can be envisioned [11]:

- Point-to-point (unicast): The relay satellite and a user have a dedicated link to transmit the data. Traffic is modeled through *traffic source models*, i.e. an approximation of each individual traffic source that randomly generates demand for the network resources [45].
- Point-to-multipoint (multicast): The relay satellite and a set of users share a common channel to transmit the data. Traffic is modeled through *network traffic models* that "characterize the spatial and temporal distribution of the traffic intensity" [45].

Satellite communication networks for terrestrial users typically require point-to-multipoint connectivity and therefore use *network traffic models*. As an example, both [45], [46] present a methodology for estimating the offered traffic to a constellation of relay satellites from a set of Earth based mobile subscribers. Their work divides the Earth surface in regular [46] or variable [45] geographic areas to determine the number of expected users for each of them. Then, they assume a given random distribution for the amount of required services (e.g. two outgoing and two incoming calls per day, Poisson distributed) and a given resource allocation (156 fixed seconds per call) to derive the load for the network.

This approach is very well suited for scenarios where there is a large number of users trying to access a node on the network. Nevertheless, relay satellite networks for spatial users tend to use point-to-point connections with less than ten users per spacecraft. Therefore, using *traffic source models* is generally more appropriate and can lead to more accurate results. These models typically define statistical properties of the traffic being sent for each user such as, for instance, the amount of data rate required to successfully achieve a transmission.

Reference [47] represents a comprehensive study on the types of traffic generated in space missions. It gives specifically designed models for voice, video, telemetry and command services based on past NASA missions. Their results indicate that the transmission data rates required to support a successful communication can be well modeled by means of gamma distributions. One might note that these results are consistent with the traffic modeling approach followed in [12] for broadband services over space IP networks. In fact, combining the works of [12] and [47] would allow to numerically assess the gain of using packet-switched architectures vs. circuit-switched architectures in space relay satellite systems.

2.5. Modeling the Network Performance

Once the network topology and incoming traffic have been properly modeled, the last step to simulate a space communication network requires understanding how data flows through the nodes so as to assess whether it can support the incoming traffic. Literature review indicates that the first question can be answered using two main approaches:

- Define the stack of protocols that each node on the network will implement. The simulator will process bit streams and packets at each node of the network in order to determine how to route them and what fraction of the data being sent is real useful information (the throughput).
- Define a generic data flow as an amount of data rate that is required to achieve successful communication between nodes of the network. The simulation can only infer whether the links capacities are high enough to support these incoming data flows given the evolution of the network topology. Understanding what fraction of the data being sent is real useful information is usually done by means of proxies or rules of thumb (e.g. the header of an IP packet plus an Ethernet frame is 40% of the packet length).

[30–33], [35] present tools that use the first aforementioned approach. In particular, they use the commercial network simulator QualNet [48] so as to define the protocols used in every layer of the OSI reference model [49]. For instance, [30] simulates an "Orion to ISS" mission scenario that uses TDRSS to relay the information from the manned capsule to either NASA's White Sands or Guam complexes. The stack of protocols at the Orion spacecraft and ground stations include a custom protocol for the link layer, Internet Protocol for the network layer and User Datagram Protocol (UDP) for the transport layer. Similarly, [25], [26] introduce tools that use the freely available network simulator *ns* (instead of QualNet) to perform the same task. In turn, [50] studies the effect of multi-topology networks with respect to the common routing algorithms through the commercial network simulator OPNET [51].

On the other hand, [28], [46], [52–56] compute the network performance from a certain degree of abstraction. The basic premise in their computations is that the network is a set of nodes interconnected by links with a given capacity. Traffic generated by users is routed from the source to the destination through intermediate nodes subtracting at each hop a certain amount of the link's capacity. In other words, evaluating the performance of the network is basically done by means of a routing algorithm that models how data is directed. No information regarding the specifics of the packets being sent is used and there is no modeling of the protocols that interact with them.

Reference [55] categorizes the routing algorithms used to evaluate networks in two main groups: shortest-path based algorithms (Dijkstra, Kruskal, k-shortest paths) and multi-commodity flow algorithms. The first approach is usually more common in scenarios where desired QoS[8] is to minimize the latency or number of hops experienced in a transmission. However, since all the information is typically routed through the best path there is no guarantee that a set of links will not become congested, thus reducing the actual throughput of the network. Both [55] and [56] suggest that this problem can be tackled using a k-shortest path routing algorithm, that is, the best *k* paths between a source and a destination are computed and then the best non-congested route is selected.

[8] Quality of Service

The main advantages of shortest-path based are as twofold. First, they are computationally more efficient than generic linear programming problems. Second, the user has the ability to decide what metrics have to be used in order to assess the cost of sending information over a hop. As an example, [50] defines the cost of a link as the ratio between the link that has maximum bandwidth and the bandwidth of that particular link.

Alternatively, routing algorithms based on the multi-commodity flow problem are specifically designed to maximize the throughput of the network. Their main goal is to balance the network load over all links regardless of what is the number of hops and latency required for a particular service. Reference [55] argues that this limitation hinders the applicability of multi-commodity flow algorithms in designing routing strategies for LEO constellation networks that must support multiple types of services. Nevertheless, [54] proves that this can be partially overcome by establishing priorities among them. In fact, its set-up as a generic linear programming minimization problem subject to a set of constraints has enough flexibility to capture both which services should be prioritized and what is the expected queuing delay at each node of the network.

All in all, modeling the performance of the network by means of a protocol based approach vs. an algorithm based approach depends on the requirements of the tool being developed. The following table tries to capture the pros and cons identified through the literature review:

Strategy		Advantages	Disadvantages
Protocol based approach		• High fidelity of the simulation and the results. • The simulation can be used to benchmark a particular protocol over the end-to-end network. • The simulation can be used to validate new custom designed protocols. • Changes in the network topology are automatically accounted on the discrete event engine.	• Very time expensive simulations. • Space protocols[9] are usually not available on the network simulator. They have to be manually programmed.
	Shortest path	• Computationally efficient. • Can be used to minimize the latency/number of hops for delay sensitive services. • The user has the ability to decide the cost metric on	• Does not capture network balancing to allow throughput maximization. • Proxies to maximize throughput lose part of the computational efficiency.

[9] Protocols published by the Consultative Committee for Space Data Systems

Algorithm based approach		the link according to the desired evaluation.	• Changes in the network are captured through snapshots that define static topologies in particular time intervals.

	Multi-commodity flow	• Network load is balanced on all links in order to obtain maximum throughput.
		• Formulation through a minimization problem subject to restrictions allows:
		  ◦ Capturing the priorities between different services.
		  ◦ Computing the queuing size at the nodes.
			• Computationally demanding, especially if the problem variables are either binary or integers.
			• Difficult to encode because the number of restrictions is typically very large.
			• Changes in the network are captured through snapshots in a set of time intervals. This can dramatically increase the encoding and computational complexity of the problem.

Table 5. Strategies to simulate the performance of satellite communication networks

2.6. End-to-end Network Simulators and the System Architecting Paradigm

Section 1.2.1 introduced the concept of System Architecting. It stated that architecting a complex system is difficult because one has to leverage competing requirements from multiple stakeholders in order to find a solution that maximizes the value while minimizing the cost. It also emphasized the need for developing tools with enough fidelity to capture how value is delivered to the stakeholders instead of focusing on the specifications and design parameters of the system.

In order to understand the appropriateness of current end-to-end network simulators with respect to the System Architecting Paradigm, five main questions have to be answered:

(1) Does the tool define all the decision variables and constraints of an architecture?
(2) Does the tool effectively enumerate the full span of possible architectures?
(3) Does the tool effectively simulate the performance of an architecture?
(4) Does the tool effectively capture the value delivery flow to the stakeholders?
(5) Does the tool provide information to help the users understand the obtained results?

Literature review suggests that (1) is almost never addressed. Most tools do not focus on the architecting process of space communication systems but instead emphasize the differences in particular design parameters. In other words, the tool assumes a predefined architecture (e.g. a constellation of 24 satellites flying under a Walker pattern) and then assesses the performance of the system based on a set of design variables (for instance, IP-based vs. ATM-switched routing algorithms). Similarly, (2) is never properly tackled since enumerating the full span of architecture requires answering (1) to understand what possibilities are feasible and relevant.

On the other hand, most tools focus their attention on question (3). Multiple approaches have been identified depending on the degree of fidelity and computational complexity of the simulation process. [30], [32] are two exemplary tools that use high definition simulations to understand the ability of a relay data system to meet the requirements of one user under a particular mission scenario. Their results are certainly very valuable in the mission planning phases of a spacecraft where there is a high degree of confidence on the specific design parameters of the mission. However, architecting a relay system is usually done with a lot of uncertainty in both how it is going to be configured and what exact missions requirements will be imposed. Therefore, using a high definition simulation can become burdensome because:

- A high level decision might change the whole mission scenario.
- Multiple configuration parameters might have to be modified if an architectural decision is changed.
- The time to build up and simulate one scenario is large.

On the other hand, specifically designed tools with a moderate level of fidelity are typically too stringent in the architectures they can model. Their computational time and number of design parameters might be more appropriate for a system architecting tool, but they usually lack flexibility to capture a full set of architectures. As an example, [46] describes a framework to analyze the performance of LEO and MEO polar constellations for mobile satellite communications. However, equations presented are not valid for inclined orbit constellations thus limiting the number of architectures that can be successfully captured.

Reference [26] presents the only tool that devotes some attention to (4). It defines a *system effectiveness synthesis* that methodologically computes the benefit of an architecture depending on a set of metrics and their weighting values. It also presents multiple alternatives for synthesizing a set of metrics into one single figure of merit, although the resulting values are so similar that they could probably be considered the same given the uncertainty inherent on the system architecture and simulation process. Moreover, the tool presented does not go beyond computing a single benefit measure for the architecture and therefore does not indicate how a set of stakeholders are satisfied.

Finally, question (5) is typically well treated in all the tools that have been developed. Be it through already existing graphing capabilities (mathematical software, graph generation software, etc.) or through specifically designed GUIs, the tools tend to present the results of the simulation process in a synthesized and structured way. Nevertheless, since most models are not intended to provide aid in a system architecting context they do not include explanation facilities that indicate how a particular metric or stakeholder was satisfied.

2.7. Identification of Research Goals and Thesis Statement

The review of the state-of-the-art simulation tools in the context of space communication networks and specifically in space relay systems reveal that:

- There is no clear understanding of what decisions are architecturally distinguishing and have a major impact in both the performance and the cost of the system.
- There is no methodology to properly capture the performance of the network other than discrete end-to-end simulation of the orbital dynamics and communication protocols.
- There is no tool that focuses in determining how value is delivered to the stakeholder instead of assessing the system performance.
- There is no tool that is specifically build to aid the system architecting process and promotes trade-off and sensitivity analyses in the context of space communication networks.

Therefore, the objectives of this thesis can be summarized in the following points:

- Understand the specificities of space communication networks and how to efficiently model them.
- Apply a rule-based expert system for architecting space communication networks.
- Develop a framework to assess the value of a space communication network based on its overall performance as a unified network.

3. A Model to Evaluate Space Communication Architectures

3.1. Introduction

Section 1.2.1 introduced the concept of developing computational tools to aid the system architecting process. It presented a general definition for *value* on a system (benefit at cost) and emphasized the need for tools that effectively capture the value delivery flow to the stakeholders. In this context, this chapter presents the computational tool developed for architecting space communication networks.

The overall structure of the model contains three main elements: an architecture enumerator, a search algorithm and an architecture evaluator. Given a discrete set of decisions and options for each of them, the architecture enumerator is responsible for creating valid architectures. If the number of decisions and options per decision is small, the architecture enumerator can create all the possible architecture and hand them directly to the evaluator. Nevertheless, in most cases this is not possible due to the large tradespace being explored. At this point, a heuristic search algorithm is introduced between the enumerator and the evaluator. The former will initially create a manageable population of architectures and hand them to the evaluator. Based on their assassement, the search algorithm will use this initial population to generate a new set of architectures that will, hopefully, be better than the first ones. This process will be repeated until an optimal solution or non-dominated set is found.

This chapter describes the work done on the evaluator part of the model. Based on the definition of a system's value, it is clear that an architecture evaluator must be able to compute both the performance and the cost side of the space network. The following sections present a detailed description of the performance model of the evaluator. In particular, they first introduce the modeling elements of the system based on the rule-based exert system Jess [57]. Next, they provide an overall view of the performance model along with the most representative information flows. Finally, they present a detailed description of the different parts of the model, clearly stating their interfaces, inputs and outputs.

3.2. Modeling the System

A generic space communication network can be modeled as a set of independent constellations that operate coordinately with a set of ground stations in order to provide services to a set of customers. In this system, three main types of actors can be envisioned:

(1) A *relay satellite*, that is, a satellite that can have multiple communication payloads to retransmit a signal from a source to a destination.
(2) A *ground station* or ground sink from where data is sent and/or received.
(3) A *user*, i.e. a satellite that is flying independently from the network but wants to use its resources in order to transmit data to and/or from a ground station.

Figure 1 provides a high level overview of the definition and constitutive elements for a generic space communication network. Note that the users are not part of the network architecture per se, but they directly interact with it and partly define the properties of their connecting nodes.

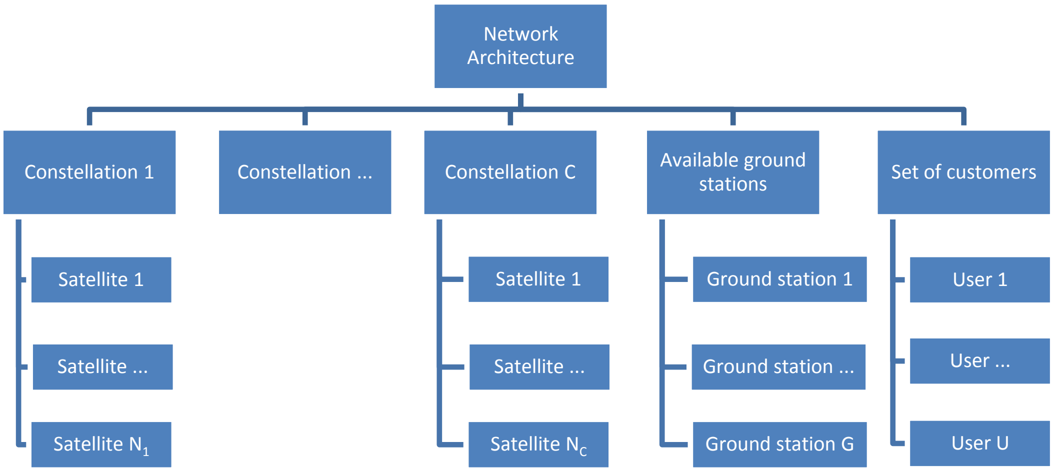

Figure 1. Space network modeling elements

Additionally, in order to properly model the communication links between satellites, ground stations and users it is also necessary to define the physical properties of the antennas, transmitters and receivers. This is done by appending antennas and communication payloads to all of them and then assessing the compatibility between the transmitting and receiving terminals.

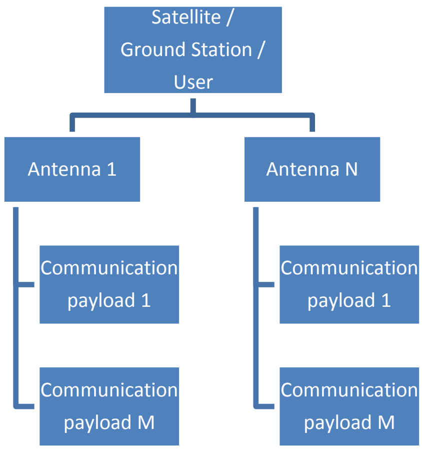

Figure 2. Communication elements on the space network

Since the tool being implemented stores the information in Jess facts that can then be queried and modified by the rule-based engine, a template for each of the aforementioned elements has been programmed. The external user of the tool will use these templates in order to create a database of facts that contain all the required input information for the system. From this initial body of facts, the model will then evaluate an architecture based on the VASSAR methodology presented in section 1.2.2.3.

The following sections (3.2.1 to 3.2.10) provide a detailed description for the main templates that have been defined, emphasizing the pieces of information that are essential to the development and behavior of the overall performance model. Note that the templates herein presented are limited to those that capture constitutive elements of the system such as constellations of satellites or ground stations. Other auxiliary templates have been defined (for instance, an RF-LINK template) but they do not contain information that has to be input to the model. Instead, they are used internally to perform the network modeling computations.

3.2.1. The Architecture Template

The architecture template contains all the information that defines space communication architecture. As such, three main types of data are encoded:

(1) Metrics: They store the performance and cost of the architecture. Their value is automatically filled once the performance and cost models have evaluated the network.
(2) Parameters: They define input parameters that remain unchanged among all the architectures. For instance, the current model assumes that the ground segment of the network is constant. Therefore, it is defined as a parameter in the architecture template.
(3) Decisions: They define system parameters that can be changed in order to produce new different architectures. Their values are automatically filled by the heuristic search engine. As an example, the architecture template defines the number of planes as a decision. Its value can be changed based on a discrete set of options.

3.2.2. The Constellation Template

The constellation template stores the basic information to define a constellation of relay satellites. Three main parameters are used to that end:

(1) Number of planes: The number of orbital planes that are evenly spaced in an Earth-based reference frame. This information is used to compute the satellite's orbit right ascension of the ascending node (raan).
(2) Number of satellites per plane: The number of satellites that are evenly spaced in each one of the orbital planes. This information is used to compute the satellite's orbit mean anomaly.
(3) Orbit: It indicates the orbit altitude, eccentricity, inclination and argument of perigee for any plane of the constellation. It also gives a unique identifier for the type of

satellite used in the constellation. This information is encoded by means of an orbit fact (see section 3.2.4).

This set of parameters allows the user to define custom constellations of satellites with a moderate level of fidelity. For instance, for circular orbits it contains the same information as a Walker constellation pattern where the relative spacing between satellites in adjacent planes is set to zero. If a more accurate model is needed, the tool also offers the possibility of inputting STK database identifiers for satellites that are already in orbit. In this case, the constellation design parameters are ignored and the program automatically selects and propagates the orbits based on real information.

3.2.3. The Satellite Template

Once the position of the orbital elements has been determined, the next step is to define the intrinsic characteristics of each satellite. The satellite template is used to contain the information regarding the relay satellite's subsystems, bus, antennas and communication payloads. Depending on the architecture, the model will automatically populate these fields in order to estimate both the available links to and from a spacecraft as well as its subsystem design and associated cost.

The satellite definition divides the antennas being carried on a spacecraft depending on their primary intended use. In particular, five main categories have been identified:

(1) User antennas: They are used to communicate the relay satellite with the users.
(2) ISL antennas: They are used to communicate the relay satellite with another relay satellite of the same constellation (InterSatellite Link).
(3) ICL antennas: They are used to communicate the relay satellite with another relay satellite from another constellation (InterConstellation Link)
(4) GS antennas: They are used to communicate the relay satellite with a ground station.
(5) Other antennas: They are used to model payloads that are carried by the satellite but are external to the network. They impact the spacecraft mass, volume and power requirements, thus influencing its cost but not its performance.

Using this categorization is useful since it easily indicates what links between two terminals are logically viable. For instance, one can program a set of rules that forces links between ISL payloads to be feasible while it invalidates a connection from a ground station to a satellite through an ISL antenna.

3.2.4. The Orbit Template

The orbit template is used to define a generic orbit that can be used by all the satellites of a constellation. The following parameters are required:

(1) Orbit identifier. It has to be identical to the identifier of the satellite using the orbit.
(2) Orbit altitude or semi-major axis.
(3) Orbit argument of perigee (set to 0 by default).
(4) Orbit eccentricity (set to 0 by default).
(5) Orbit inclination.
(6) Orbit right ascension of the ascending node (set to 0 by default).
(7) Orbit central body (set to Earth by default).
(8) Orbit type: LEO, MEO, GEO or HEO.

The orbit template also stores the STK identifiers to create constellations based on real orbital information. This parameter is, however, optional.

3.2.5. The Antenna Template

An antenna template defines the antenna physical characteristics. In particular, it determines the type of antenna being used (e.g., parabolic, dipole, phased-array, horn, helix), its efficiency, aperture size, overall dimensions, field of view (FOV) and associated payloads. Two main considerations must be emphasized: first, the antenna FOV represents the overall solid angle that can be covered by physically or electronically steering the beam(s) generated when combining the antenna and the communication payload characteristics. For instance, low gain antennas operating at low frequency bands (UHF, VHF or S-band) will typically have a wide FOV that is determined by the beamwidth of the antenna radiation pattern. Alternatively, high gain antennas for X, Ku or Ka-bands will use parabolic antennas with beamwidths in the order of mrad. In this case, the FOV will be determined by the possibility of physically steering the antenna and pointing it to the desired destination. Figure 3 provides an example of the second aforementioned case in which the FOV angle is set to 8 deg. Note that the tool assumes the footprint volume to be conical, although real systems such as TDRSS have also rectangular and elliptical FOV antennas.

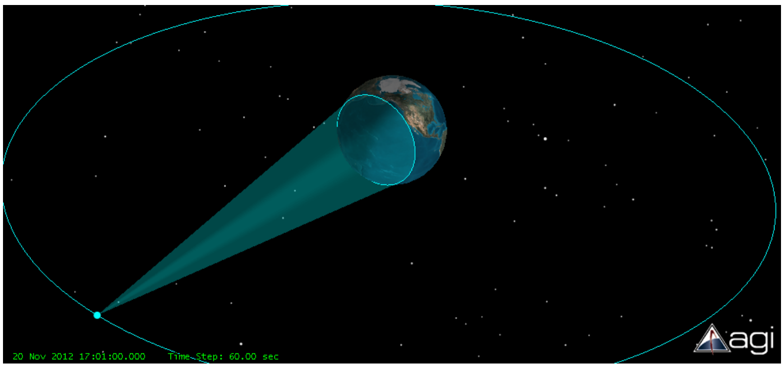

Figure 3. FOV of a geosynchronous satellite

The second important consideration in the antenna definition is the type of communication payloads that are associated with it. As a first approximation, one can assume that each communication payload has its own dedicated and optimized antenna to transmit. Nevertheless, the reality of current systems such as TDRSS indicates that an antenna can be shared among multiple communication payloads as long as they operate at different frequency bands. This fact has to be properly modeled in the tool because of its impact in the spacecraft design and cost.

3.2.6. The Payload Template

The communication payload temple is used to model three categories of communication payloads: single access RF payloads; multiple access RF payloads; and optical payloads. In order to do so three types of information are included:

(1) General information: It indicates type of payload (RF vs. optical), the frequency band, the modulation and the coding scheme.
(2) Link budget information: It indicates the antenna gain and G/T, the available bandwidth and transmitting frequency, as well as the transmitting power. For optical links, the quantum efficiency of the receiver is also required.
(3) Multiple Access information: It indicates the number of independent beams that can be formed. The particularities of the multiple access scheme being used are disregarded because it is assumed that they can be approximated by reducing the nominal data rate of the MA payload.

3.2.7. The Ground Station Template

The ground station template contains the information regarding the position (latitude and longitude) of the facility on the Earth surface as well as the antennas and communication payloads available at the location. One can also specify an STK database identifier so as to automatically define the position of the ground station based on current existing facilities.

Analogously to a satellite, a ground station also contains fields to store the number and type of antennas that are available at the facility. Nevertheless, since the model does not consider ground-to-ground links or ground-to-user links, there is no need to differentiate the payloads according to their main purpose. In this case, all the antennas specified will be used to compute communication links between a ground station and a satellite.

3.2.8. The User Template

The user template stores the generic data to model a customer of the network. A first set of parameters is related to the spacecraft orbital motion. As it is done for a satellite, one can input user defined values for the orbit altitude, eccentricity, argument of perigee, raan and mean anomaly or let STK propagate them through a unique database identifier.

The second type of parameters that must be defined indicates the services that the customer will require from the network along with their relative importance. This information is used to assess the satisfaction of the user once the network schedule has been computed. For instance, a human space flight mission such as the ISS might require voice, video and command services with a relative importance of 50%, 25% and 25%. The voice service might be prioritized because continuous communication with the astronauts might be critical for a mission.

Finally, the last piece of required information to fully define a user is the type of antennas and communication payload being carried. As for a ground station, a user only defines one type of antennas and communication payloads since it is assumed a user can only communicate to a relay satellite.

3.2.9. The Service Template

A network that wants to provide services to a set of customers has to take into account the particularities of traffic being processed and relayed through its nodes. In the context of space communications, five primary types of service have been identified:

(1) Telemetry: low/moderate data rate, low latency, high criticality one way housekeeping data sent from the payloads and subsystems of a satellite to a ground station.
(2) Command: low data rate, low latency, very high criticality one way data sent from a ground station to a satellite.
(3) Science data return: high/very high data rate, high latency, medium criticality one way data gathered by the instruments and payloads onboard the customer spacecraft. Although losing part of this data can compromise the value of a mission, it does not have an impact on the spacecraft survivability.
(4) Voice: low/medium data rate, real-time, high criticality two way communication channel for human missions. It can become extremely critical if emergency situations occur during a mission as it allows continuous communication with the endangered astronauts.
(5) Video: medium/high data rate, real-time, medium criticality two way communication channel between a human mission and its mission control center. It is becoming increasingly important in space operations, especially with the advent of robotic remotely operated missions.

For each of these cases, the service template stores a paradigmatic concept of operations. It is defined by the number of desired contacts per day, the data volume to return per contact and the minimum time between contacts. This information has to be extracted from the mission design documents and it is usually a function of the data generation rate for the instruments and the onboard storage capacity. For voice and video services, the concept of operations is also heavily influenced by the type of activities that are planned for the astronauts. As an example, the ISS has currently a dedicated single access service with the TDRSS in order to allow voice and video services continuously (24 hours a day, 365 days a year).

3.2.10. The Stakeholder and Objective Templates

The stakeholder and objective templates group missions into high level categories according to a set of priorities or weights. Their general structure contains the following multislots:

(1) user-id: list with the identifiers of the customers belonging to the objective.
(2) user-satisfaction: list with the satisfaction of the customers belonging to the objective.
(3) user-weight: list with the relative weights of the users belonging to the objective.
(4) objectives-id: list with the identifiers of the subobjectives belonging to the objective.
(5) objectives-satisfaction: list with the satisfaction of the subobjectives belonging to the objective.
(6) objectives-weight: list with the relative weight of the subobjectives belonging to the objective.

The only difference between a stakeholder and an objective template is that the former does not include the multislots (1), (2) and (3). In other words, a stakeholder is defined by a set of objectives while an objective is specified by both a set of subobjectives and users. This hierarchical structure is captured in Figure 4.

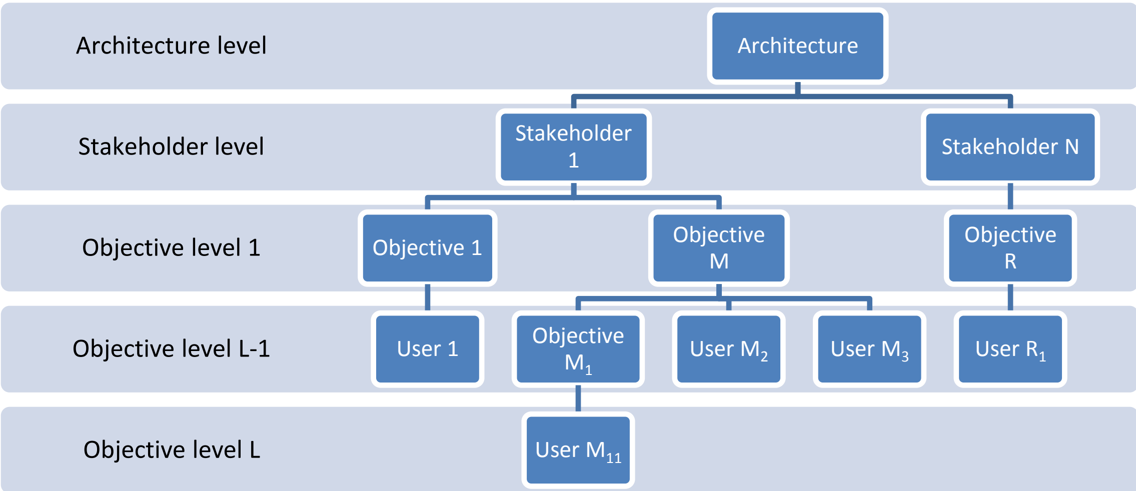

Figure 4. Stakeholder/Objective/User hierarchical structure

3.3. High Level Structure of the Model

The overall structure of the performance model is based on the VASSAR architecture presented in section 1.2.2.3. Based on an input database of facts, and first set of *database*, *manifest* and *attribute inheritance rules*, the model will assert the architecture being analyzed and ensure that its information is consistent for the following computations. Then, a performance model will simulate the behavior of the network in order to assess what users and services can be successfully scheduled in a given time horizon. Next, the *requirement satisfaction rules* will assess the satisfaction of the users and this information will be propagated through the objective, stakeholder and architecture level through the *value aggregation rules*.

Figure 5 presents a high level flow diagram of the performance model used to evaluate a given space communication architecture. Contrary to the VASSAR methodology, the performance of a network requires numerically expensive computations that render the rule-based system alternative inefficient. This fact can be mitigated by using a mixed approach where multiple types of computational programs interact and take advantage of their own strengths: STK to compute the orbital motion of the network nodes; Jess to encode the rules that define logical restrictions among different parts of the system; and Matlab to perform scheduling and link budget computations over a large set of nodes and time intervals.

The program starts by importing the input information into the Database Module where it is stored in the form of facts which use the templates and structures presented in the previous sections of this chapter. This information is queried by the Manifest Module in order to import only the necessary data to perform an end-to-end evaluation of a given the network architecture. This process is divided into three main tasks: compute the network topology over a representative time interval (typically one day); assess the capacity of the links given the communication properties of the payloads carried by the network nodes; and calculate a network schedule so as to understand how resources can be optimally allocated.

The Constellation, Link, RF Spectrum and RF/Optical Link Budget Modules perform the first aforementioned task. In order to define a dynamic evolvable network topology, three different types of restrictions have to be considered:

(1) Geometrical restrictions: As ground stations, relay satellites and customers revolve around the Earth (in its surface or orbiting at different altitudes), there are contact opportunities driven by existence or inexistence of direct line of sight between them. These contact opportunities can be computed taking into consideration three main facts: the geometrical occultations caused by the Earth and the orbital dynamics of the nodes; the main direction at which the communication antennas are pointed; and the FOV (due to steering or beamwidth limitations) for these antennas. These restrictions are treated on the Constellation Module.
(2) Radiofrequency/Optical restrictions: A wireless communication between a source and a destination is only possible if sufficient power is received. This received power is inversely quadratic with the distance between the two nodes which, in turn, varies as they move around the Earth. Moreover, a RF link has to take into account national and international spectrum regulations. They indicate what fraction of a band is useable for a particular application, thus restricting the bandwidths and frequencies that can be used in a transmission. These restrictions are treated in the RF Spectrum and RF/Optical Link Budget Module.

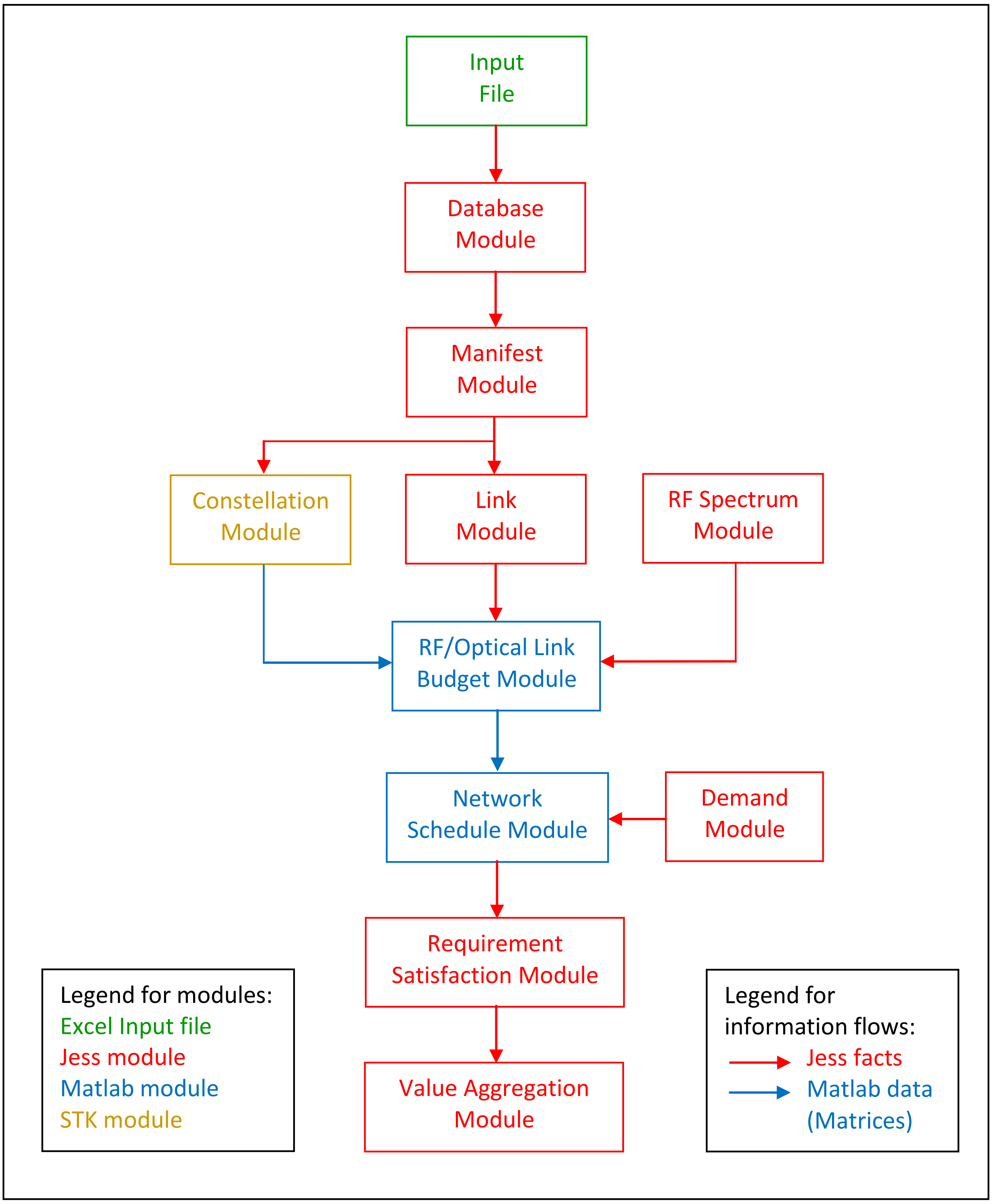

Figure 5. High level structure of the performance model

(3) Logical restrictions: Even if a link is viable because there is direct line of sight and enough received power, logical restrictions can directly invalidate its operation. Depending on the origin for these limitations, two types can be envisioned: communication-based restrictions, i.e. a link can only be established if the two nodes have the same physical and data link layer parameters configured (same modulation, polarization, coding scheme, multiple access scheme); and system-based restrictions, that is, a communication payload might be designed to do a specific type of link (for

instance an intersatellite link or a space to ground link). These considerations are taken into account in the Link Module.

The output of these four modules is a three dimensional binary matrix $NT \in M_{NxNxT}$ where $NT_{ijt} = 1$ if the time interval $[t, t + \Delta t)$ holds a successful communication between node *i* and node *j*. In this notation, constants *N* and *T* capture respectively the number of nodes in the network and the number of time intervals being simulated. In turn, Δt is computed as:

$$\Delta t = \frac{Total\ simulation\ time = \ \Delta T}{Number\ of\ time\ intervals = T} \qquad \textbf{Eq. 1}$$

The *NT* matrix is the first input of the Network Schedule Module. The second input comes from the Demand Module as a list of jobs that need to be serviced. Each job contains information regarding the user that is demanding it, as well as the desired concept of operations. The latter is used to differentiate between services that typically use intermittent short duration contacts (such as telemetry or command transmissions) and almost continuous two-way connections (such as the ISS dedicated link for voice and video services with the astronauts).

Once the network schedule has been computed, the model uses the Requirement Satisfaction Module in order to infer the degree of satisfaction for all the users of the system. These user satisfactions are finally aggregated by means of averaged sums to the objective, stakeholder and architecture level on the Value Aggregation Module.

3.4. The Performance Model

This section provides a detailed description of the main modules that compose the performance module. For each of them, it clearly states the expected inputs and outputs as well as the calculations that take place on them. Examples of the rules and/or equations used are presented in order to clarify how the tool estimates the performance of the system.

3.4.1. The Database Module

The Database Module is used to store the input information for all the elements of the model. Each time an architecture needs to be evaluated, copies of the required database facts are replicated on the adequate modules. For instance, the Database Module will contain all the possible communication payloads that are available for any network architecture. When a particular one is manifested, only the communication payloads that belong to it are copied into the Manifest Module for further analysis.

The rules of the Database Module have two main purposes: first, compute additional parameters from the initially input information. As an example, the definition of a communication payload might include its transmit power and gain at a particular band. From them, the Database Module will automatically calculate the EIRP (Figure 6). The second purpose of the Database Module is to ensure the consistency of the input information. The rules will check the different values of the asserted facts and ensure that (1) they are within the expected bounds (Figure 7) and (2) they are compliant with structural relationships of other parts of the system (Figure 8).

```
(defrule DATABASE::calculate-payload-EIRP-from-power-and-gain
    ?f <- (DATABASE::PAYLOAD (EIRP nil) (gain ?gain&~nil)
                             (tx-power-dB ?tx-power&~nil))
    =>
    (bind ?eirp (+ ?tx-power ?gain))
    (modify ?f (EIRP ?eirp))
)
```

Figure 6. Database Module rule to compute the EIRP from the transmit power and gain

```
(defrule DATABASE::check-optical-payloads-for-wavelength
    (declare (salience -10))
    ?f <- (DATABASE::PAYLOAD (id ?id) (payload-type optical)
                             (wavelength ?wl&~1.55e-6))
    =>
    (modify ?f (wavelength 1.55e-6))
    (printout t (str-cat "The wavelength slot for optical payload " ?id "
                          has been automatically set to 1550 nm") crlf)
)
```

Figure 7. Database Module rule to set the wavelength of an optical communication payload

```
(defrule DATABASE::check-antenna-payloads
    (declare (salience -10))
    (DATABASE::ANTENNA (id ?id) (payloads $? ?pay $?))
    (not (DATABASE::PAYLOAD (id ?pay)))
    =>
    (bind ?rule "check-antenna-for-payloads")
    (bind ?msg (str-cat "Antenna " ?id " has payload " ?pay " that is not
                          defined. Check the payload definition"))
    (throw (new jess.JessException ?rule ?msg ""))
)
```

Figure 8. Database Module rule to check that an antenna does not contain a non-defined payload

3.4.2. The Manifest Module

The Manifest Module is in charge of containing all the information to evaluate a single architecture. Based on the initial architectural fact, the *manifest rules* are used to automatically assert all the elements of the system. They start by creating a fact for each constellation and then they propagate this information to assert the satellites and orbits that define them. Next, the antennas and communication payloads for the satellites are declared, along with the ground stations, users and services. Figure 9 presents an exemplary rule of such a task. Given an existing manifested satellite with a non empty set of antennas, the rule asserts all the required antenna facts as well as the communication payloads that depend from them.

```
(defrule MANIFEST::assert-antennae-and-payloads-from-satellite
    (declare (salience 10))
    ?sat <- (MANIFEST::SATELLITE (antennae $?ant&:(notempty$ $?ant))
                                 (id ?name) (payloads $?current))
    (DATABASE::ANTENNA (id ?id) (payloads $?payls))
    (test (contains$ ?id ?ant))
    (test (not-contains-all$ $?payls $?current))
    =>
    (modify ?sat (payloads (append-all$ $?current $?payls)))
    (assert (MANIFEST::ANTENNA (id ?id) (payloads $?payls)
                               (purpose payload) (satellite ?name)
                               (name (eval (str-cat ?name "-" ?id)))))
    (foreach ?payl $?payls
        (assert (MANIFEST::PAYLOAD (id ?payl) (satellite ?name)
                                   (parent ?id)
                                   (name (eval (str-cat ?name "-" ?id)))))
    )
)
```

Figure 9. Manifest Module rule to assert the antennas and payloads from a satellite

3.4.3. The Constellation Module

The main goal of the Constellation Module is to assess what links are available at a certain instant of time due to geometrical restrictions (see section 3.3). In order to do so, the tool performs the following actions:

(1) Determine all the baseline architectures that are feasible based on the number of possible planes, satellites per plane and orbits. Each baseline architecture will contain only one constellation and it will be assumed that its relay satellites carry all the possible antennas and communication payloads.
(2) Create a STK scenario and configure it to the desired simulation time and time step.
(3) Populate the STK scenario with satellites and facilities that model relay satellites, customer spacecraft and ground stations.
(4) Append conical sensors to all the satellites and facilities, one for each antenna beam that can be supported. A single access communication payload will have only one associated beam while a multiple access payload will vary depending on the number specified on the payload template.
(5) Configure the FOV of the conical sensors according to the field of view of the communicating antennas.
(6) Compute the access duration and distance for all the links between two sensors of the STK scenario. This action is done following a two step process: first, the access time is

used in order to infer the best orientation of the transmitting and receiving antennas (being nadir and zenith the two main possibilities for each of them); second, information regarding the access distance at multiple time steps is stored for further processing.

The output of the Constellation Module is a $GL \in M_{NxNxT}$ matrix where $GL(tx, rx, t)$ indicates the distance between the transmitting beam *tx* to the receiving beam *rx* at the time interval $[t, t + \Delta t)$. This information is obtained by parsing the *AER reports* that STK produces once the scenario has been properly created and configured.

3.4.4. The Link Module

The Link Module is used to determine what links are logically viable. For each pair transmitter-receiver, this module asserts a fact LINK that contains the following information regarding both ends of the communication link:

- Type of payload for the transmitter: user, isl, icl or gs (see section 3.2.3)
- Technology used on the transmitting payload: RF, optical.
- Type of parent object containing the transmitting payload: relay satellite, ground station or user.
- Constellation holding the parent object for the transmitting payload. For ground stations and users this field is ignored.
- Type of payload for the receiver: user, isl, icl or gs (see section 3.2.3)
- Technology used on the receiver payload: RF, optical.
- Type of parent object containing the receving payload: relay satellite, ground station or user.
- Constellation holding the parent object for the receiving payload. For ground stations and users this field is ignored.

Once a fact LINK has been asserted, the Link Module contains a set of rules that indicate whether it is logically viable. These rules are used to encode the following generic constraints:

- A user payload can only communicate with a payload that is contained by a user.
- An isl payload can only communicate with an isl payload of another satellite of the same constellation.
- An icl payload can only communicate with an icl payload of another satellite of another constellation.
- A gs payload can only communicate with a payload contained by a ground station.
- A link is valid only if the transmitting and receiving payloads have interoperable parameters: same modulation, same coding scheme and same frequency band.
- A link is valid only if the transmitting and receiving payloads use the same technology.

Figure 10 presents a paradigmatic rule of the Link Module. Given a fact that represents an initially viable link, this rule checks that the communication payloads of the transmitter and receiver are compatible (use the same technology) and invalidates it if necessary. Similar rules are used to ensure that all the aforementioned criteria are met.

```
(defrule LINKS::invalidate-links-between-different-types-of-payloads
    (declare (salience 5))
    ?f <- (LINKS::LINK (tx-payload-type ?tx-pay-type)
                       (rx-payload-type ?rx-pay-type)
                       (viable yes))
    (test (neq ?tx-pay-type ?rx-pay-type))
    =>
    (modify ?f (viable no))
)
```

Figure 10. Link Module rule to invalidate links between RF and optical payloads

Once a link is found to be viable, the Link Module automatically asserts a new fact RF-LINK or OPTICAL-LINK that represents the link both from a logical and a communication point of view. It contains the same information as the original LINK fact plus the communication parameters intrinsic to the transmitting and receiving payloads and antennas. In particular, the following information is specified:

- RF link: transmitter band, modulation, coding scheme and EIRP. Receiver band and G/T.
- Optical link: transmitter EIRP, wavelength, modulation and coding scheme. Receiver gain and quantum efficiency.

This set of facts is the output of the Link Module and serves as the inputs of the RF Spectrum Module and ultimately to the RF/Optical Link Budget Module.

3.4.5. The RF Spectrum Module

The RF spectrum module is used to determine the frequency and available bandwidth for a particular link based on the national (National Telecommunications and Information Administration and Federal Communications Commission) and international (International Communication Union) regulations. Table 6 has been extracted from NASA's Organization of Spectrum Management and indicates the assigned bandwidths and transmitting frequencies at multiple bands for different types of links. The information herein presented is limited to the Near-Earth domain while a similar table can be found for deep space missions.

Band	Ground Network	Space Network
S-band	Uplink: 2025 - 2110 MHz Downlink: 2200 - 2290 MHz	Forward: 2025 - 2110 MHz Return: 2200 - 2290 MHz
X-band	Uplink: 7190 - 7235 MHz	Forward: N/A

	Downlink: 8025 - 8500 MHz	Return: N/A
Ku-band	Uplink: 14.60 – 15.25 GHz Downlink: 13.40 – 14.05 GHz	Forward: 13.75 - 14.0 GHz Return: 14.8 - 15.35 GHz
Ka-band	Uplink: N/A Downlink: 25.5 - 27 GHz	Forward: 22.55 - 23.55 GHz Return: 25.25 - 27.5 GHz

Table 6. Near Earth Spectrum Allocations

The RF Spectrum Module builds upon the RF-LINK facts that have been asserted by the Link Module. It uses the information from Table 6 in order to perform two main tasks:

(1) Determine the transmit frequency and available bandwidth for a particular type of link.
(2) Invalidate links that are trying to communicate at bands where there is no granted allocation.

Figure 11 presents an exemplary rule of task (1). The rule looks for a RF-LINK fact between a relay satellite and a customer where the transmitting and receiving payloads operate at Ka-band. It then computes the available bandwidth for the link as the difference between the upper and lower allocated frequencies. This approach assumes that there is neither a safeguard band nor a channelization scheme for a given band allocation.

```
(defrule RF-SPECTRUM::get-BW-for-Ka-band-forward-link
    "This rule gets the BW for a Ka-band forward link based on NTIA red-
    book allocations"
    ?f <- (LINKS::RF-LINK (rx-payload ?rx-id) (tx-type SATELLITE)
                          (rx-type USER) (tx-band Ka))
    =>
    (bind ?upper-freq 23.55e9)
    (bind ?lower-freq 22.55e9)
    (bind ?BW (- ?upper-freq ?lower-freq))
    (modify ?f (bandwidth ?BW))
)
```

Figure 11. RF Spectrum Module rule to compute the bandwidth of a Ka-band forward link

Task (2) is carried out through the rule presented in Figure 12. The rule indicates that an RF-LINK where the transmit frequency and bandwidth have not been computed cannot be valid and therefore must be discarded. In order to work properly, this rule is given the lowest priority so as to ensure that it only fires when there is no spectrum allocation for a particular link and frequency band.

```
(defrule RF-SPECTRUM::invalidate-links-without-spectrum-allocation
    (declare (salience -1))
    ?f <- (LINKS::RF-LINK (frequency ?freq) (bandwidth ?BW) (viable yes))
    (or (test (eq ?freq nil)) (test (eq ?BW nil)))
    =>
    (modify ?f (viable no))
)
```

Figure 12. RF Spectrum rule to invalidate links without band allocation

Finally, it is important to emphasize that OPTICAL-LINK facts are not treated by the RF Spectrum Module since there is currently no standard national or international regulations for the management of the optical frequency bands. Instead, it is assumed that the bandwidth is not a restriction for an optical link and the transmit wavelength is fixed to 1550 nm for American communication payloads (the current standard).

3.4.6. The RF/Optical Link Budget Module

The RF/Optical Link Budget Module main goal is to determine what links are feasible given the communication characteristics of the transmitting and receiving payloads. The first input for this module is the set of RF-LINK and OPTICAL-LINK facts that contain all the required information for link budget calculations: equivalent isotropically radiated power (EIRP), antenna G/T, modulation, coding scheme, carrier frequency and available bandwidth. The second input comes from the Constellation Module and is expressed through the distance matrix *GL*.

For radiofrequency links, the tool computes the feasible data rate for a communication link from Eq. 4. As such, the transmission speed R_b is computed as the minimum between what is physically realizable given the transmitter power (Eq. 2) and the available bandwidth (Eq. 3).

$$R_{b1} = 10^{\frac{EIRP+\frac{G_R}{T}-L_{FS}-L_{pt}-L_{at}-L_{im}-L_{pol}-10\log k-\frac{E_b}{N_0}}{10}} \qquad \textbf{Eq. 2}$$

$$R_{b2} = \Gamma \cdot BW \qquad \textbf{Eq. 3}$$

$$R_b = min\{R_{b1}, R_{b2}\} \qquad \textbf{Eq. 4}$$

The particularities for the terms of these equations are as follows:

- $EIRP$ is the transmitting EIRP [dBW]. It is inherited from the RF-LINK fact.
- G_R/T is the receiver antenna gain to system noise temperature ration [dB/K]. It is inherited from the RF-LINK fact.
- L_{FS} are the free space losses due to the signal propagation on the vacuum [dB]. They are computed as $10\log\left(\frac{\lambda}{4\pi d}\right)^2$, where d is inferred through the *GL* input matrix.
- L_{pt} are the antenna pointing losses [dB]. They have a fixed value of 1 dB
- L_{at} are the atmospheric losses [dB]. They are only accounted for ground to satellite links with a fixed value of 5 dB.
- L_{im} are the implementation losses [dB]. They have a fixed value of 2 dB.
- L_{pol} are the polarization losses [dB]. They have a fixed value of 2 dB.
- k is the Boltzmann constant [J/K].
- $\frac{E_b}{N_0}$ is the required link bit energy to noise density. It is computed based on a bit error rate of 10^{-5}, the selected modulation and coding scheme, and a 1.5 dB link margin.

- Γ is the modulation and coding scheme spectral efficiency. It is estimated as $\Gamma = \frac{\log_2 M}{(1+\alpha)} \cdot \eta_{cod}$ where *M* is the number of levels in the modulation, α is the roll-off factor and η_{cod} is the coding scheme efficiency.

For optical links, the calculation of the feasible data rate is based upon a non-coherent M level Pulse-Position Modulation (M-PPM). This approach is grounded on NASA's optical link final report [58] and expert interviews suggesting that only one optical payload will be available to the space agency on the 2020 – 2030 timeframe. Next, the fundamental equations to compute an optical link are presented along with an explanation of their parameters:

$$P_R = EIRP + G_R + L_{FS} - L_{pt} - L_{at} - L_{im} \quad \textbf{Eq. 5}$$

$$E_{ph} = h \cdot \frac{c}{\lambda} \quad \textbf{Eq. 6}$$

$$E_{pulse} = \frac{n_s \cdot E_{ph}}{\eta_{quantum}} \quad \textbf{Eq. 7}$$

$$Pulse\ rate = \frac{P_R}{E_{pulse}} \quad \textbf{Eq. 8}$$

$$R_b = Pulse\ rate \cdot b \quad \textbf{Eq. 9}$$

- $EIRP$ is the transmitting EIRP [dBW]. It is inherited from the OPTICAL-LINK fact.
- G_R is the receiver antenna [dB]. It is inherited from the OPTICAL-LINK fact.
- L_{FS} are the free space losses due to the signal propagation on the vacuum [dB]. They are computed as $10\log\left(\frac{\lambda}{4\pi d}\right)^2$, where *d* is inferred through the *GL* input matrix.
- L_{pt} are the antenna pointing losses [dB]. They have a fixed value of 2 dB
- L_{at} are the atmospheric losses [dB]. They are only accounted for ground to satellite links with a fixed value of 4 dB.
- L_{im} are the implementation losses [dB]. They have a fixed value of 8.5 dB.
- h is the Planck constant [J·s].
- $\eta_{quantum}$ is the receiver quantum efficiency. It is inherited from the OPTICAL-LINK fact.
- b is the number of bits per symbol in the M-PPM modulation: $b = \log_2 M$.
- n_s is the number of signal photons required at the detector. It is computed based on a BER equal to 10^{-5} and a number of noise background photons n_b equal to 5.

The output of the RF/Link Budget Module is a set of two matrices. The first one $LB \in M_{NxNxT}$ is a binary three dimensional matrix where $LB(tx, rx, t) = 1$ indicates that the link from node *tx* to *rx* is available at time interval $[t, t + \Delta t)$. Similarly, the capacity matrix $C \in M_{NxNxT}$ stores the achievable data rate for a given link and time interval.

3.4.7. The Demand Module

The Demand Module is used to compute the list of jobs that the network will have to serve in a given time interval (typically one day). For each job, the module estimates its priority based on the input information from the user/objective/stakeholder hierarchy presented in section 3.2.10.

$$Job\ priority = stp \cdot \left(\prod_{\forall objective} op_i \right) \cdot up \cdot sp \qquad \textbf{Eq. 10}$$

where *stp* is the stakeholder priority, *op* is the objective priority, *up* is the user priority, *sp* is the service priority. The product term of the equation captures the fact that an objective can have subobjectives at multiple levels of dependencies.

Once the job priority has been the computed, the last step is the store all the information that will characterize a contact. To that end, the tool gathers the number of desired contacts, data volume per contact and minimum time between contacts for the service being considered, as well as its preferred payload. This last piece of information is only used on users that carry more than one communication payload so as to ensure that their data is routed through the appropriate links.

3.4.8. The Network Schedule Module

The Network Schedule Module uses the information from the RF/Optical Link Module and Demand Module in order to compute a possible schedule for the whole system. This schedule will indicate what users are being serviced in a particular time interval, as well as how many and for how long the network resources are being used.

In section 2.3 multiple approaches for creating scheduling algorithms were investigated: heuristic algorithms, static algorithms and dynamics algorithms. While the former represent the best alternative from a computational perspective, it is also clear that they are suboptimal and therefore can underestimate the capacity of the network. Alternatively, the latter can optimally allocate the resources of the network but are usually grounded on linear programming techniques with exponentially increasing complexity with the number of network nodes.

In order to select one of the aforementioned approaches, it is necessary to benchmark their pros and cons against the main objectives for creating a system architecting tool. In particular, two main considerations must be leveraged: what degree of fidelity is required in order to capture the value delivery process to the system stakeholders; how large is the tradespace being explored. As a first approximation, it has been considered that a moderate level of fidelity suffices for the system architecting tool. Moreover, the plausible architectural decisions have led to, at least, a 500,000 architectural tradespace. Therefore, it is clear that the computational time per architecture has to be limited to less than one minute per architecture, resulting in the choice of a customized heuristic scheduler.

Figure 13 shows the high level structure of the greedy scheduling algorithm implemented on the tool. For each time interval $[t, t + \Delta t)$, the algorithm starts by computing the current status of the network. In order to do so, it first checks each link and, if it is valid, finds how many time steps $n\Delta t$ it will remain so. Then, it computes the total data volume that can be returned over the link by multiplying $n\Delta t$ times the link capacity. The result of this process is a data volume matrix $DV \in M_{NxN}$ that indicates the total amount of bits that can be routed on the network given its current static topology.

Once the network status has been computed, the scheduler tries to serve as many jobs as possible without overloading any link on the network. In order to do so, the algorithm requires three pieces of information:

(1) Who is the originating node: This data is stored in the job list that has been computed on the Demand Module.
(2) Who is the addressee node: The tool assumes that any ground station can be the sink of the data being transferred. Relay system's like TDRSS have proprietary ground networks that connect any of their ground stations to the customer's mission control center (MOC). This is currently not modeled in the tool.
(3) How much data has to be sent over the contact: This data is stored in the job list that has been computed in the Demand Module. It is part of the concept of operations of the service being required by the user.

Additionally, the user can also specify a constraint for the time between scheduled contacts. In this case, a job will only be served if the time between the current and last contact is greater than a specified value. This constraint accounts for the fact that most missions want to schedule a contact only after their instruments have had enough time to record some amount of information.

Finding a viable path between (1) and (2) is done by means of a shortest path algorithm. Being that the case, a suitable metric for estimating the cost of a link has to be defined (Eq. 11).

$$Link\ cost_{ij} = max\{DV_{ij} - (3), 0\} \qquad \textbf{Eq. 11}$$

The intuitive idea behind this metric is that of minimizing the total unused capacity of the system. For instance, a telemetry service that can be routed through a multiple access S-band payload will use it instead of a high data rate Ka-band single access. Furthermore, if a link does not have enough capacity to hold a contact, it will be automatically invalidated and considered inexistent for the shortest path algorithm.

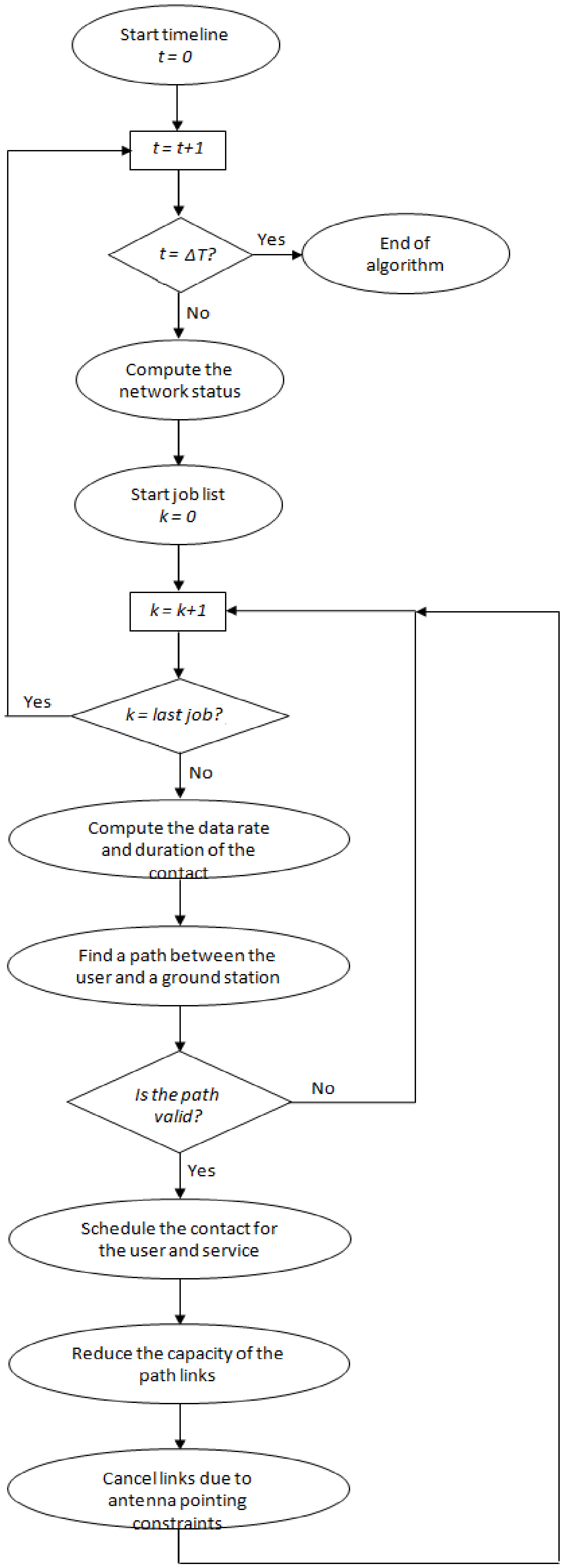

Figure 13. High level structure of the scheduling algorithm

It is also possible that a transmission from a source to a destination has multiple viable paths, one to each ground station of the system. As an example, consider an augmented TDRS system that had two ground stations (White Sands and Guam) and intersatellite links. In this case, the scheduling algorithm would find two optimal paths for serving a user: one would route the data to a single spacecraft and then download it to its nearest ground station; another one would have the user contact one relay satellite, which would send the data to another satellite and this last one would download it to its in-view ground station. In this case, selecting the best option is done by comparing the aggregated cost of all the links in a path, and choosing the smallest one.

Finally, once a viable path from (1) to (2) has been computed, the last step in the scheduling process is to update the status of the network during the time intervals while the job will be served. Two main actions take place at this point: first, the available data rate for the links belonging to the selected path is decreased according to the amount of resources used by the job. Second, some links are invalidated due to antenna pointing constraints. In particular, if two or more communication payloads are sharing the same antenna and one of them is used for a particular contact, it is then assumed that the other communication payloads can only communicate to that same user.

Figure 14 presents an example of the output of the schedule algorithm. It presents the computed schedule for the International Space Station (ISS). Three services have been specified on the simulation, telemetry, video and science data return. The first one operates on an S-band single access and typically requires short bursts of data. Alternatively, the video and science data return services operate through a Ku-band single access that is held exclusively for almost an entire day. This is consistent with the current concept of operations of the actual ISS, where a single access of the TDRSS system is reserved for its exclusive use.

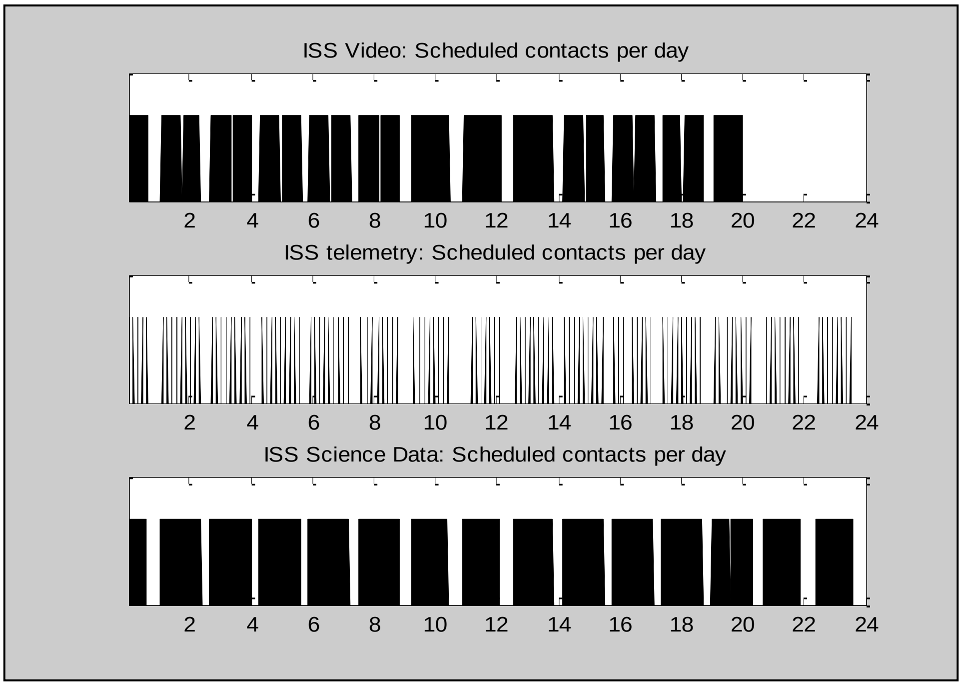

Figure 14. ISS computed schedule

The output of the Network Schedule Module is a set of Jess facts that indicate what fraction of the desired contacts have been satisfied by the network (Eq. 12)

$$Service\ metric\ = \frac{Number\ of\ scheduled\ contacts}{Number\ of\ desired\ contacts} \qquad \textbf{Eq. 12}$$

These facts will be processed by the Requirement Satisfaction Module in order to assess the overall satisfaction of a user.

3.4.9. The Requirement Satisfaction Module

The Requirement Satisfaction Module is used in order to assess the satisfaction of a user based on its demanded services and the number of contacts that he has scheduled. To that end, the module takes in two different inputs, the Jess facts from the Network Schedule Module and the requirement satisfaction input information.

The requirement satisfaction information comes in the form of multiple non-overlapping intervals from 0% to 100% with an associated satisfaction (one for each service that a user wants to support). For instance, an Earth observation mission can specify the following ranges for the science data return service:

- Thresholds: [10; 25; 50; 75; 100]
- Scores: [0;5;15;40;80;100]

In this example, the mission has no satisfaction if the scheduler is only able to grant less than 10% of the desired contacts because most of the collected data by the instruments is lost. Alternatively, if the schedule contains 75% to 100% of the desired contacts, then the science return service for the mission will be almost fully satisfied (80%). This process is automatically done by means of the rules presented in Figure 15.

```
(defrule CAPABILITIES::compute-requirement-satisfaction
    ?f <- (CAPABILITIES::REQUIREMENT (FOM-value ?val&~nil)
                                     (thresholds $?t) (scores $?s))
    =>
    (bind ?sat (compute-requirement-satisfaction ?val $?t $?s))
    (modify ?f (satisfaction ?sat))
)

(deffunction compute-requirement-satisfaction (?v ?t ?s)
    (if (eq ?v nil) then
        (return 0)
    )
      (if (>= ?v (nth$ 1 ?t)) then
        (return (nth$ 1 ?s))
    elif (< ?v (nth$ (length$ ?t) ?t)) then
        (return (nth$ (length$ ?s) ?s))
    )
    (bind ?p 2)
    (bind ?i 1)
    (while (< ?i (length$ ?t) )
      (bind ?ub (nth$ ?i ?t))
        (bind ?lb (nth$ (+ ?i 1) ?t))
        (if (and (< ?v ?ub) (>= ?v ?lb)) then
            (return (nth$ ?p ?s))
        )
        (bind ?i (+ ?i 1))
        (bind ?p (+ ?p 1))
    )
)
```

Figure 15. Requirement Satisfaction Module rule

3.4.10. The Value Aggregation Module

The Value Aggregation Module performs the last task of the performance module. Based on the satisfaction of the services required by a user and the information from the stakeholder/objective/user hierarchy (see section 3.2.10), the module computes a single metric for the benefit of the overall architecture. This is done by means of a four step process:

(1) Services are aggregated through their relative importance to obtain the user satisfaction (Figure 16).

```
(defrule SATISFACTION::compute-user-satisfaction
    "This rule computes the satisfaction of a user"
    ?f <- (SATISFACTION::USER (id ?id)
                              (services-id $?services-id)
                              (services-weight $?services-w)
                              (services-satisfaction $?services-sat)
                               (satisfaction nil))
    =>
    (foreach ?service $?services-id
      (bind ?sat (get-requirement-satisfaction ?id ?service))
      (bind ?p (member$ ?service $?services-id))
      (bind $?services-sat (replace$ $?services-sat ?p ?p ?sat))
    )
    (modify ?f (services-satisfaction $?services-sat)
               (satisfaction (dot-product$ $?services-sat $?services-w)))
)
```

Figure 16. Value Aggregation Module rule for a user

(2) Users and subobjectives are aggregated through their relative importance to obtain the objective satisfaction (Figure 17)

```
(defrule SATISFACTION::compute-objective-satisfaction
      "This rule computes the satisfaction of an objective"
    ?f <- (SATISFACTION::OBJECTIVE (users-satisfaction $?users-sat)
                                   (users-weight $?users-weight)
                                   (objectives-satisfaction $?objs-sat)
                                   (objectives-weight $?objs-weight)
                                   (satisfaction nil))
    (test (eq (member$ -1 $?users-sat) FALSE))
    (test (eq (member$ -1 $?objs-sat) FALSE))
    =>
    (if (neq (length$ $?users-weight) 0) then
      (bind ?user-sat (dot-product$ $?users-sat $?users-weight))
    else
        (bind ?user-sat 0)
    )
    (if (neq (length$ $?objs-weight) 0) then
      (bind ?obj-sat (dot-product$ $?objs-sat $?objs-weight))
    else
        (bind ?obj-sat 0)
    )
    (modify ?f (satisfaction (+ ?user-sat ?obj-sat)))
)
```

Figure 17. Value Aggregation Module rule for an objective

(3) Objectives are aggregated through their relative importance to obtain the stakeholder satisfaction (Figure 18).

```
(defrule SATISFACTION::compute-stakeholder-satisfaction
      "This rule computes the satisfaction of an stakeholder"
    ?f <- (SATISFACTION::STAKEHOLDER (satisfaction nil)
                                  (objectives-weight $?objs-weight)
                                  (objectives-satisfaction $?objs-sat))
    (test (eq (member$ -1 $?objs-sat) FALSE))
    =>
    (bind ?obj-sat (dot-product$ $?objs-sat $?objs-weight))
    (modify ?f (satisfaction ?obj-sat))
)
```

Figure 18. Value Aggregation Module rule for a stakeholder

(4) Stakeholders are aggregated through their relative importance to obtain the architecture benefit (Figure 19).

```
(defrule SATISFACTION::compute-architecture-satisfaction
      "This rule computes the benefit of an architecture"
    ?f <- (MANIFEST::ARCHITECTURE (benefit nil)
                                 (stakeholders-weight $?stks-weight)
                                 (stakeholders-satisfaction $?stks-sat))
    (test (eq (member$ -1 $?stks-sat) FALSE))
    =>
    (bind ?benefit (dot-product$ $?stks-sat $?stks-weight))
    (modify ?f (benefit ?benefit))
)
```

Figure 19. Value Aggregation Module rule for the architecture

The output of the Value Aggregation Module is also the end of the performance model. By automatically inferring the benefit of the architecture, one can directly assess the overall average degree of satisfaction for all its stakeholders. If a more detailed representation of the results is required, the module also keeps track of the partial satisfaction of services, users, objectives and stakeholders. This ensures that the performance model is traceable and one can understand who is extracting more benefit from the system and why.

4. Conclusion

4.1. Summary

This thesis has provided an introductory view into the field of space communication networks from a system architecture perspective. The overall goal has been to describe the work done in creating an automated software tool that is able to compute the value of a network architecture based on its architectural decisions. Emphasis has been put on assessing the performance of the system, although similar endeavors have been simultaneously undertaken from a costing standpoint.

Chapter 1 has been entirely devoted to summarize basic concepts in the three main required areas of expertise: system architecting, rule-based expert systems and space communication networks. It has justified the need for computer based tools that aid the system architecting process and allow exploring large tradespaces. It has also presented VASSAR, a generic framework for assessing the value of an architecture based on a rule-based expert system. Finally, it has provided an overview of the current elements of a space communication network. For each of them, a qualitative trade-off between multiple options has been introduced as a motivation for automating this process through a piece of software.

The second chapter has been dedicated to literature review in the field of space communication network. Its main goal has been to understand how past researchers have tackled the problem of assessing the performance of an overall space communication network. To that end, the chapter has broken down the task in multiple parts: estimating the orbital moment of the network nodes, computing a schedule to assess how network resources are allocated, and model the particularities of the traffic flows being supported. These discussions have lead to multiple approaches to program end-to-end network simulators, which have been benchmarked against the needs and specificities of a system architecting tool. Results indicate that current models for space communication networks do not have the appropriate level of fidelity or are too stringent in the architectures they simulate.

Finally, chapter 3 has presented the performance model developed for architecting space communication networks. It has first indicated the overall structure of the tool, emphasizing the need for three main elements: an architecture enumerator, a heuristic search algorithm and an architecture evaluator. It has then introduced the modeling elements of the system based on the rule-based engine Jess: constellations, satellites, ground stations, users, antennas, communication payloads, stakeholders and objectives. Next, it has presented the high level structure of the performance model and has indicated its similarities with the VASSAR architecture. Finally, it has provided a detailed description of each module of the tool, clearly stating its inputs, outputs and interfaces, as well as the fundamentals of the computations being carried out.

4.2. Future Work

4.2.1. Modeling the Space Communication Network

Future work in modeling the space communication network includes validating the tool against a current already existing system such as NASA's TDRSS. This step will bring confidence to the model and reinforce the validity of the results obtained in later tradespace studies. It is also expected that the performance model will continue to evolve in order to increase the fidelity of the output results. The scheduling algorithm is the most ad-hoc part of the model and therefore requires more attention and validation. The current trends of research indicate that this task is nowadays usually undertaken by means of linear programming algorithms. Substituting the heuristic scheduler by this alternative will clearly delimitate the differences between the two approaches, both from an optimization and a time-consuming perspective. In this context, if the linear programming scheduler proves to be fast enough from a computational standpoint, it will clearly replace the current heuristic algorithm.

Other future work includes creating rules to evaluate differences in the communication strategy for the onboard payloads. As an example, a set of rules can be used to estimate the capacity and required signal to noise ratio based on the payload architecture being selected: bent-pipe, circuit-switched or packet-switched. In particular, literature has suggested that an OBP circuit-switched architecture improves the link performance in 3 to 5 dB compared to a bent-pipe strategy. This fact can reduce the power requirements for the communication payload, thus reducing the overall cost of the spacecraft. Similarly, placing packet-switched nodes allows statistical multiplexing of the traffic flows being carried which in turn reduces the overall needed capacity for a link. Depending on the burstiness of the traffic, this reduction can range from 5% to more than 50%, therefore clearly impacting the required power for the transmission and the cost of the relay spacecraft.

Finally, another possible improvement of the tool is to enlarge its scope to an interplanetary network. In this case, the model would allow placing relay nodes and customers at multiple places of the solar system in order to create a deep space extended network. This approach would very much align with the current plans of NASA's Space Communication and Navigation Program (SCaN), which include orbital communication elements in Mars and the Moon. This would increase the capacity of the global system and allow reliable robotic and human exploration activities in these planets.

4.2.2. Exploring Future Space Communication Network Architectures

Once the tool has been validated, it will be exercised in order to identify a set of architectures suitable for the next generation of TDRSS. Although the current system has been in place for more than twenty years and is still being upgraded (TDRS-K will be launched January 2013 and TDRS-M is planned on 2014), SCaN has been actively engaged in finding less costly, more flexible and improved architectures for future systems. Results from the tool will complement their ongoing effort by clarifying the trades, performing high level what-if studies and suggesting best sets of candidate architectures.

On the other hand, once the tool has been upgraded to an interplanetary level, it will also be exercised for analyzing the value of having relay assets placed in Mars, Moon or, potentially, other further locations (Sun-Earth or Jupiter-Earth Lagrange points). Assuming a 2020-2030 timeframe, the expected deep space missions will indicate whether developing such a complex system is justified and maximizes the value delivered to NASA and its stakeholders.

5. Bibliography

[1] N. Program, "Space Communications and Navigation (SCaN) Network Architecture Definition Document (ADD) Volume 1 : Executive Summary." 2011.

[2] J. L. Hayden and K. Bashin, "Space Internet Architectures and Technologies for NASA Enterprises 1," 2010.

[3] K. Bhasin and J. Hayden, "Developing Architectures and Technologies for an Evolvable NASA Space Communication Infrastructure," no. July, 2004.

[4] S. Applications, I. International, C. Schaumburg, and C. Nas, "Satellite Communications Database Technology."

[5] M. W. Maier and E. Rechtin, *The Art of System Architecting*. 2009.

[6] D. Selva Valero, "Rule-Based System Architecting of Earth Observation Satellite Systems by," Massachusetts Institute of Technology, 2012.

[7] E. F. Crawley, "System Architecture." .

[8] C. Grosan and A. Abraham, *Intelligent Systems: A Modern Approach*. 2011.

[9] C. L. Forgy, "Rete: A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem." p. 21, 1982.

[10] W. J. Larson and J. R. Wertz, *Space Mission Analysis and Design*. 1999, p. 929.

[11] G. Maral, *Satellite Communication Systems: systems, techniques and technology*. 2009.

[12] K. Y. Jo, "Satellite communications with Internet Protocol (IP) efficiency," *MILCOM 2009 - 2009 IEEE Military Communications Conference*, pp. 1–7, Oct. 2009.

[13] M. Bever, J. Freitag, S. Linsky, J. M. Myers, R. M. Nuber, J. L. Prieto, and E. R. Wiswell, "Fast-packet vs circuit switch and bent pipe satellite network architectures," *Internation Journal of Satellite Communications*, vol. 17, no. 2–3, pp. 83–105, 1999.

[14] J. Hant and D. Lanzinger, "Effect of dynamic topology on circuit and packet-switched satellite networks," *Geoforum*, 2005.

[15] N. Blefari-melazzi, M. Femminella, and G. Reali, "Dynamic Bandwidth Allocation in a Circuit-Switched Satellite Network: Provision of Deterministic and Statistical QoS Guarantees," *IEEE INFOCONF*, vol. 1, pp. 225–234, 2000.

[16] National Communications & Information Administration, *MANUAL OF REGULATIONS AND PROCEDURES FOR FEDERAL RADIO FREQUENCY*, no. May. 2011.

[17] K. S. Sebacher, S. G. Lambert, J. A. Pautler, and J. P. Carter, "Laser Crosslink Configurations for RF Satellite Communications Systems," *MILCOM 1985 - 1985 IEEE Military Communications Conference*, 1985.

[18] M. Toyoshima, "Trends in satellite communications and the role of optical free-space communications," *Journal of Optical Networking*, vol. 4, no. 6, 2005.

[19] E. Sein, G. Planche, B. Laurent, J.-P. Bouzinac, G. Oppenhauser, and T. Tolker-nielsen, "Space applications of optical communications," *Annales Des Télécommunications*, vol. 58, no. 11–12, 2003.

[20] Globalstar, *Description of the Globalstar System*. 2000.

[21] N. A. and S. Administration, *Space Network Users ' Guide (SNUG)*, no. August 2007. 2007.

[22] T. Lewis, "2nd TDRSS Workshop," vol. 4594, 1996.

[23] "ESA Telecommunications & Integrated Applications." [Online]. Available: http://www.esa.int/esaTE/SEM5GGKTYRF_index_0.html.

[24] A. K. Erlang, "Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges," 1917.

[25] T. R. Henderson and R. H. Katz, "Network simulation for LEO satellite networks." 200AD.

[26] X. Wang, H. Yang, and H. Zhao, "The Modeling, Simulation and Effectiveness Evaluation for Communicatoin Networks of Multi-layer Satellites Constellation." 2006.

[27] "The network simulator - ns-2." [Online]. Available: http://www.isi.edu/nsnam/ns/.

[28] L. P. Clare, C. Wang, and M. W. Atkinson, "Multiple satellite networks: performance evaluation via simulation." 1987.

[29] J. S. Baras and T. Charuhas, "Hybrid (Satellite and Terrestrial) Communication Networks: Object Oriented Generic Tools for Simulation and Management." pp. 1–7, 1992.

[30] E. Jennings and D. Heckman, "Performance Characterization of Space Communications and Navigation (SCaN) Network by Simulation," *2008 IEEE Aerospace Conference*, pp. 1–9, Mar. 2008.

[31] E. Jennings, R. Borgen, S. Nguyen, J. Segui, T. Stoenescu, S. Wang, and S. Woo, "Space Communications and Navigation (SCaN) Network Simulation Tool Development and Its Use Cases," *American Institute of Aeronautics and Astronautics*, no. August, pp. 1–11, 2009.

[32] B. Barritt, W. Eddy, and S. Matthews, "Unified Approach to Modeling & Simulation of Space Communication Networks and Systems Integrating network stack and astronautical physics simulators," 2010.

[33] E. Jennings and D. Heckman, "Architecture Modeling and Performance Characterization of Space Communications and Navigation (SCaN) Network Using MACHETE."

[34] S. Feng, H. Zhu, and G. Li, "Dynamic modeling and simulation for LEO satellite networks," *2008 11th IEEE International Conference on Communication Technology*, pp. 37–40, Nov. 2008.

[35] L. Baranyai, E. G. Cuevas, S. Davidow, C. Demaree, and P. DiCaprio, "End-to-end network modeling and simulation of integrated terrestrial, airborne and space environments," *2005 IEEE Aerospace Conference*, pp. 1348–1353, 2005.

[36] "AGI." [Online]. Available: www.agi.com.

[37] C. Ahara and J. Rossbach, "The Scheduling Problem in Satellite Communication Systems." 1967.

[38] D. L. Brandel, W. a. Watson, and a. Weinberg, "NASA's advanced tracking and data relay satellite system for the years 2000 and beyond," *Proceedings of the IEEE*, vol. 78, no. 7, pp. 1141–1151, Jul. 1990.

[39] A. K. Mainzer, P. Eisenhardt, E. L. Wright, F. Liu, W. Irace, I. Heinrichsen, R. Cutri, and V. Duval, "Preliminary Design of The Wide-Field Infrared Survey Explorer." .

[40] P. Kit, D. Steitz, G. Diller, D. Ainsworth, and M. Neiman, "Terra : Flagship of the Earth Observing System," no. November. 1999.

[41] Y. Fang and Y. Chen, "Constraint Programming Model of TDRSS Single Access Link," no. August. 2006.

[42] M. Adinolfi and A. Cesta, "Contributed Paper Heuristic Scheduling of the DRS Communication System," vol. 8, no. 2. 1995.

[43] J. Munkres, "Algorithms for the assignment and transportation problems," *Society for Industrial and Applied Mathematics*, vol. 5, no. 1, p. 32, 1957.

[44] M. L. Balinski, "Notes on the assignment and transportation problems of linear programming," in *The University of Michigan Engineering Summer Conferences*, 1963.

[45] G. Schorcht, U. Freund, H. Salzwedel, H. Keller, and I. Germany, "A Hierarchical Object-Oriented Global Traffic Model for Simulation of Mobile Satellite Communication Networks." 1997.

[46] M. Werner, A. Jahn, E. Lutz, and A. Bottcher, "Analysis of System Parameters for LEO/ICO-Satellite Communication Networks," vol. 13, no. 2. 1995.

[47] T. Stoenescu and L. Clare, "Traffic Modeling for NASA's Space Communications and Navigation (SCaN) Network," in *2008 IEEE Aerospace Conference*, 2008, pp. 1–14.

[48] "Strategic Network Technologies." [Online]. Available: http://www.scalable-networks.com/content/.

[49] ISO/IEC, “Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model.” p. 1996.

[50] G. Xiantai, Y. Hongchao, Y. Feng, and L. Gang, “Modeling and simulation of small satellite constellation networking using multi-topology routing,” *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, no. Iccasm, pp. V12–143–V12–147, Oct. 2010.

[51] “OPNET.” [Online]. Available: http://www.opnet.com/.

[52] B. L. Murphy, “High Resolution Satellite Communication Simulation.” 2000.

[53] Z. Tsai, C.-C. Chuang, C. Jin-fu, and C. Huang, “Performance of A Global Circuit-Switched Satellite Communication Network.” 1994.

[54] J. Alonso and K. Fall, “A Linear Programming Formulation of Flows over Time with Piecewise Constant Capacity and Transit Times piecewise constant capacity and transit times.” 2003.

[55] V. V Gounder, R. Prakash, and H. Abu-Amara, “Routing in LEO-based satellite networks.” 1999.

[56] J. He and G. Li, “Routing strategy research based on ISL states and topology snapshot in LEO satellite constellation,” *2008 11th IEEE International Conference on Communication Technology*, pp. 13–16, Nov. 2008.

[57] E. Friedman-hill, *JESS IN ACTION*. 2003, p. 482.

[58] NASA, “Optical Link Study Group. Final Report.” p. 164, 2012.