A Web Based Architecture for Remote Access of Satellite Emulation Services

Kathy J. Liszka
The University of Akron
Dept. of Computer Science
Akron, Ohio 44325-4003

Allen Holtz
NASA Glenn Research Center
21000 Brookpark Rd.
Cleveland, Ohio 44135

Abstract

We propose a web-based architecture using standard protocols to provide a user-friendly interface and remote access to satellite emulation test-bed facilities on a parallel computing cluster. Researchers can remotely control an emulation to evaluate performance of proposed space-based Internet architectures, backbone networks, formation clusters and satellite constellations. The objective is to define the framework for an open distributed system to make a satellite emulation testbed accessible through the Internet. We use XML and XML compatible technologies as the basis for a platform-independent system.

Keywords: XForms, XML, satellite emulation

1. Mission Planning

Gaze into the sky on a clear night and you might see a tiny light that looks like a distant star traveling across the sky. It's most likely one of thousands of commercial, scientific or defense satellites orbiting the earth. They provide us coverage for telephony and television services, global positioning coordinates, as well as track hurricanes, forest fires, and collect scientific data on precipitation, solar radiation and volcanic effects. With increasing demand for these types of services, modern satellite design includes cooperating platforms to provide a backbone network [1].

Coordinated missions for data collection and transmission require protocol development. Not only do tasks performed by satellites vary, but their data capture and transmission requirements range from constant low-bandwidth to bursty, high data rate information collection. Standard network protocols such as IP fall short of these requirements although there is much to be gained for incorporating this standard into future satellite platforms. At present, most satellites work alone or in small groups to collect and/or transmit data to grounds stations.

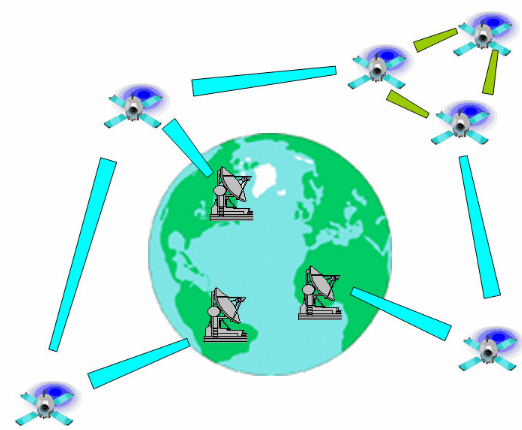

Figure 1. Complex Multi-satellite Emulation Scenarios

Building and launching satellites for the purposes of testing feasibility isn't realistic, particularly in the case of multi-platform configurations. Advanced facilities are needed to provide capabilities for a multi-user, distributed system to emulate space based Internet architectures, backbone networks, formation clusters and constellations as in Figure 1. Cooperative satellite missions planned for the near future and beyond require new models for communication to be successful and cost effective. Many concepts of the terrestrial-based standard Internet protocols can be taken advantage of. However, the dynamic and complex infrastructure of disparate missions and hardware create the need to re-evaluate and optimize the communication protocols for space. We propose a web-based architecture rooted deeply in XML technologies with an intelligent interface to help design such complex systems.

In section two, we explain the need for emulation capabilities and facilities. In section three, we develop the XML-based interface for remotely accessing such a facility. Section four further develops the architecture so that a rich set of tools may be used to study and evaluate the performance of multi-platform satellite constellations.

2. Network Emulation

Ground stations and data relay satellites use communication protocols similar to Internet protocols used on Earth with exceptions to accommodate unique situations and limitations that arise from sending data through space. Communication signals take significantly longer to travel hundreds or thousands of miles through space as opposed to a few miles between ground stations. Scheduling algorithms for transferring data must consider unique parameters such as line-of-sight, i.e., that time when a transmitting or receiving satellite will be in range of a ground station antenna. Hardware designs, and more notably launches, are expensive, so researchers must rely on emulation facilities for designing and analyzing complex multi-satellite communication architectures.

Software simulations can help model the effects of orbital motion of satellite constellations. It is important to understand how intercommunicating spacecraft change their topology, and thus, routing decisions as the result of path delay as they draw near and then pull away from each other in their individual orbits. However, an emulation tool can enable scientists to study the more subtle interaction between the hierarchical protocol layers and the impact of other issues such as power limitations, bandwidth, delay, and bit error rates. Optimization of a particular protocol layer for ground based communications may impair network performance and quality of service in the space environment where operating conditions differ significantly. For example, standard Internet routing algorithms do not take into account the inherently ad hoc nature of spacecraft that fly in different orbits. Mobile ad hoc routing protocols have been developed but they are not optimized to take advantage of orbital calculations that can predict when two platforms will be within communication range.

Parallel cluster computing facilities have been developed to test multi-platform satellite scenarios, with each computing node acting as an individual satellite, hosting the actual software that is proposed for the spacecraft platform [2, 3]. We have worked specifically with software originally developed at the University of Kansas that provided a set of libraries and application integration tools in a distributed testbed environment [4]. Our primary goal was to provide researchers the ability to test and evaluate their proposed space communication architectures remotely using an XML-based interface for scenario design.

3. Web Based Services

It is critical to capture the requirements of specific emulation scenarios in a complete and reasonable manner. The purpose of using XML for the emulation is to provide support for portable encoding of data for transfer between dissimilar systems. The original method of entering data for an

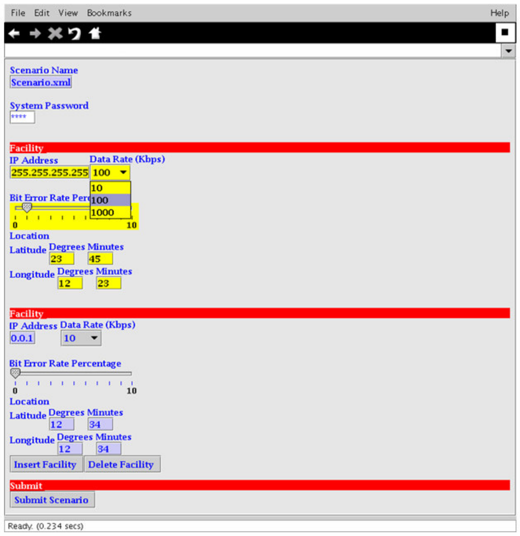

Figure 2. X-Smiles Display of a scenario

emulation scenario was through manual creation of an XML (Extensible Markup Language) file. This is the first step in creating the web-based service. A typical mission scenario may consist of several ground facilities, a dozen satellites and numerous data collection/communication instruments per satellite, all with multiple parameters to define. The size of the resultant data file may be quite large. To reduce the chances of human error, we need tools to provide a user-friendly interface and automatically encode the data into a well-formed, complete XML document. We chose XForms, the markup language intended to be the successor to the more familiar HTML Forms [5]. It uses XML to exchange data in a platform independent manner. Thus, the science is prevalent, not the format. Unlike HTML forms, the data is separated from the web page, stored in an XML format defined by the user and forwarded to the emulation facility encoded as Unicode bytes.

XForms was designed to be integrated into another document format rather than be a standalone markup language. We start with a basic XHTML documenting using X-Smiles, an open source, Java based XML browser for our working environment. X-Smiles currently supports all of the XForms features that we need for our interface design [6]. The XML file created for input to the emulation contains specific information about the satellite scenario. This includes run-time control parameters, network routes, interface bandwidths, delay and bit error rate. Satellite and ground stations specifications are included as well. With XForms we apply our XML schema to validate data input before forwarding it to the emulation facility and submitting it for execution.

Using the many form controls for data entry, such as menus, radio buttons, sensitive text, check lists and range control, we quickly provide a straightforward interactive user interface. In the prototype, we used Cascading Style Sheets (CSS) to define the style sheet for the XHTML web page to format our text, colors, and element positions. Figure 2 shows one of the input screens with the following sample input elements and features:

- scenario name – input dialog
- password – masked input
- IP address – input dialog
- data rate – select drop down
- bit error – sliding bar
- initial latitude & longitude – input dialog
- insert facility – trigger
- delete facility – trigger
- submit – trigger

Other screens include input for number, type and capabilities of antennae, data rates, orbital parameters, power information, routing tables, ground station assignment, etc. The amount of information describing a single satellite is lengthy. Creating an XML file directly is not a viable option.

In the XForms Model, an instance element defines the information that will be collected and either points to or directly contains XML. The contents must be formatted such that if they were contained in a separate file, it would be well-formed XML. The content of an instance element is for reference only and does not invoke any processing. One important XForms Model property we include is *submission*. This is used to specify the CGI file to execute when we are ready to transmit to the remote facility and run our script. In short, the XForms Model provides a non-visible definition of instance data, constraints on that instance data, and any other run-time aspects. Figure 3 shows the currently implemented components.

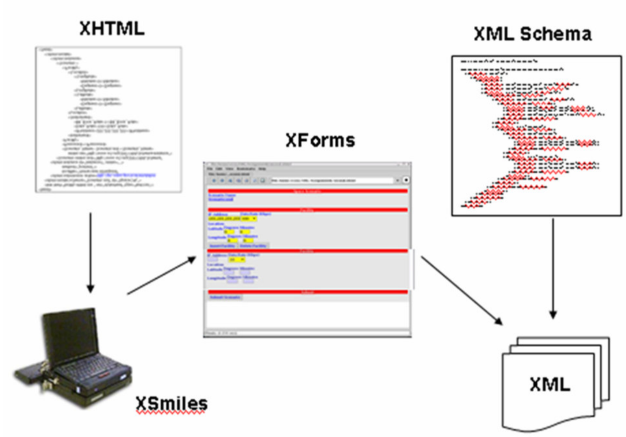

Figure 3. XML-based components of a scenario

4. Architecture Extensions

XForms is not a stand-alone technology. XLINK is used to form the basis for graphs and providing a semantic representation of the objects defined in the form and their relationship to each other. For example, Earth Observation Satellites (EOS) work together to provide global coverage for observation of phenomenon such as hurricanes and forest fires [7]. They coordinate the tasks of data sourcing (gathering data) and switching (routing) to deliver data to ground stations when they are within sight. We are using XLINK to define these relationships to each other, to the ground station facilities they are associated with for downlink and possibly to other cooperating satellite formations. In addition, associations may be made between sensors on board a satellite or between sensor suites on multiple satellites.

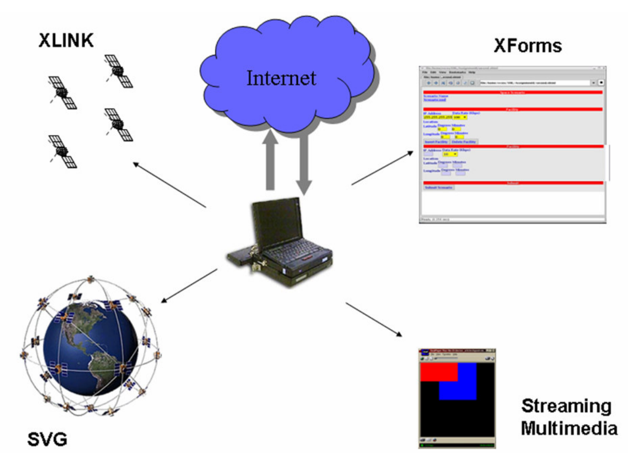

Figure 4. The Complete Web-based Architecture

Scalar Vector Graphics (SVG) can then be incorporated to add visual dynamics to the form. This is a language for describing two-dimensional graphics in XML. It can be used interactively and dynamically. Animations showing orbital positions and "live" data links can be defined and triggered either declaratively or they can be scripted. A color system for the data links will be used to give an immediate visual indication of data and error rates. These animations will be created and fed through streaming multimedia for immediate visual inspection. Figure 4 shows the final architecture design.

5. Conclusions and Future Work

Web-based access is one component to the complex task of emulating satellite constellations. It is essential for researchers to be able to access the system in a standard, user-friendly way that provides a powerful interface. We are continuing development of the XLINK component design and graphics.

6. References

[1] Internetworking and Computing over Satellite Networks, Yongguang Zhang (ed.), Kluwer Academic Press, March 2003.

[2] "Architecture and Design for a Space Based Internet Emulation System", G. J. Minden, J. B. Evans, L. S. Searl and P. J. Wagh, *Earth Science Technology Conference (ESTC) 2001*, August 2001.

[3] "Space Based Internet Network Emulation for Deep Space Mission Applications", S. Endres, M. Griffith, B. Malakooti, K. Bhasin and A. Holtz, *22nd AIAA International Communications Satellite Systems Conference and Exhibit 2004 (ICSSC)*, May 2004.

[4] "Design of a Space Based Internet Emulation System," S. R. Baliga, ITTC, University of Kansas, Lawrence, KS, Tech. Rep. ITTC-FY2003-TR-24350-05, Dec 2002.

[5] W3C XForms 1.0 Recommendation, http://www.w3.org/TR/xforms/ .

[6] X-Smiles, http://www.x-smiles.org/ .

[7] NASA's Earth observing System (EOS) Home Page, http://cospso.gsfc.nasa.gov .