WEB-ALTAIRIS: AN INTERNET-ENABLED GROUND SYSTEM

Funded under contract with NASA / GSFC / Information Systems Center

Phil Miller, AppNet, Inc.
Jason Coleman, AppNet, Inc.
Darren Gemoets, AppNet, Inc.
Kevin Hughes, AppNet, Inc.

Abstract

This paper describes Web-Altairis, an Internet-enabled ground system software package funded by the Advanced Automation and Architectures Branch (Code 588) of NASA's Goddard Space Flight Center. Web-Altairis supports the trend towards "lights out" ground systems, where the control center is unattended and problems are resolved by remote operators. This client/server software runs on most popular platforms and provides for remote data visualization using the rich functionality of the VisAGE toolkit. Web-Altairis also supports satellite commanding over the Internet. This paper describes the structure of Web-Altairis and VisAGE, the underlying technologies, the provisions for security, and our experiences in developing and testing the software.

Motivation

A typical satellite ground system involves, at minimum, an operations staff present twenty-four hours a day in one or more control center facilities, performing monitoring of spacecraft health and safety. This staff might include five to ten operators, working in shifts, fixing problems as they arise. This traditional approach, while reasonable and logical, is costly.

In recent years, there has been a trend in the NASA community towards "lights out" control centers that can operate without on-site operators. There is also the hybrid model where, during the prime shift, the control center is staffed and during the other shifts, the system is purely "lights out". When unattended, a lights out system detects anomalous conditions and, if no automatic recovery has been prescribed, the software contacts a human operator via telephone, email, or messaging. But in the nominal case, such a system works autonomously and tirelessly, significantly reducing the number of operators and specialists needed for a control center.

While the lights out approach reduces costs, the operators must still travel to the control center to analyze and resolve anomalies. It is desirable for the control center to allow the operator to address an anomaly remotely: to connect to the control center, investigate the problem with a user-interface comparable to that of the control center, and perhaps take corrective action, all from the convenience of a remote site (home, office, shopping center[1], etc.)

Background

The Microwave Anisotropy Probe (MAP[2]), is one of the medium-class explorers (MIDEX) at NASA's Goddard Space Flight Center. MAP's principal scientific mission is to measure temperature differences in the cosmic microwave background radiation. A secondary goal of the MAP mission is to assess new technologies for autonomous lights out spacecraft operations. To that end, MAP employs two separate ground systems, one well-established and one modern. The traditional ground system is an off-the-shelf product called the Advanced System for Integration and Spacecraft Test

[1] We are in the early stages of running VisAGE clients on palmtop computers with wireless connections to the Internet.

[2] MAP is scheduled to launch in early 2001. For more information, see http://map.gsfc.nasa.gov/.

(ASIST[3]). ASIST consists of a database of command and telemetry mnemonics and a STOL (Spacecraft Test and Operations Language) window for manipulating the mnemonics.

The modern ground system is a commercial product, the Altairis Mission Control System (MCS[4]). Altairis MCS differs from traditional control software, employing Finite State Modeling (FSM) to monitor and command the spacecraft. The *state model* is based upon a hierarchy of mnemonics, in which the value of each mnemonic depends upon the values of mnemonics lower in the hierarchy. States are defined for each subsystem's hierarchy node to reflect operating norms and anticipated anomalies. A properly configured[5] Altairis system contains a model of all known states of the satellite. System engineers define *state transitions*, each telling how to command the satellite from one state to another desired state. With Altairis, the operator executes pre-defined transitions rather than composing and executing complicated command procedures. Moreover, engineers can configure the system to execute transitions automatically from one state to another so that, when the satellite enters an anomalous state, the recovery is automatic.

While the increased autonomy of Altairis brings the MAP mission closer to lights out, it does not solve the accessibility problems. If Altairis cannot resolve an anomaly and the system is unattended, Altairis notifies a remote operator. The operator may receive certain information describing the anomaly, but usually must travel to the control center to investigate and resolve the problem.

Therefore, a remote access solution is needed that gives the operator the ability to obtain the full status of the mission, and possibly even command the mission, from a remote location. With the advent of technologies such as Java, wide availability of access to the public Internet, secure communication, and powerful personal computers, the technology exists to implement such a solution.

The remote access solution described in this paper is "Web-Altairis" in conjunction with the VisAGE remote visualization software. Its is currently in use in the integration/test phase of the MAP mission.

Description of Web-Altairis

Web-Altairis (WA) provides access to an Altairis control center from any remote computer. Its client/server architecture (see Figure 1) distributes ground system data to a user's machine. The server interfaces with the ground system and manages client requests for data. The client uses advanced visualization tools to allow the user to display and manipulate the data. Data distribution and visualization are implemented using the Visual Analysis And Graphical Environment (VisAGE[6]). The VisAGE toolkit provides both secure data transmissions and flexible data visualization capabilities. How the remote computer connects to the control center depends upon the network architecture of the particular mission, but modem dial-up connections and Internet connections are both supported by the software.

---

[3] For more detailed information regarding ASIST, see http://rs733.gsfc.nasa.gov/ASIST/.

[4] For more detailed information regarding Altairis, see http://www.altaira.com/AMCS/amcs.html.

[5] The Altairis FSM is developed by domain specialists, not software developers, using an intuitive graphical user interface.

[6] VisAGE is also a product of NASA/Goddard's Code 588. See http://tidalwave.gsfc.nasa.gov/avatar/visage/.

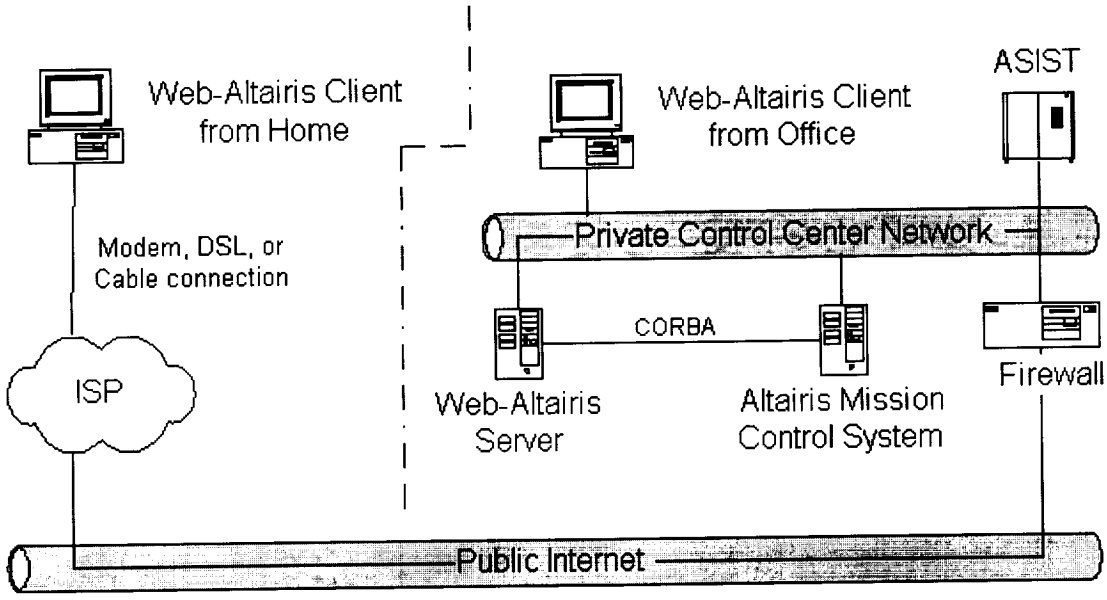

Figure 1: Web-Altairis Architecture

In a typical usage, an operator starts the Web-Altairis client, creates a connection to the Altairis ground system, and is then presented with a user interface that mimics that of the Altairis console. From this point forward the operator has access to a majority of the research and commanding capabilities available from the console. Furthermore, because Web-Altairis is built on top of the VisAGE toolkit, it provides numerous data visualization capabilities such as two- and three-dimensional charts that are not available from the Altairis console.

While the Altairis MCS ground system was not specifically designed for remote access, it is well-suited to the task for the following reasons:

- The Altairis Finite State Model consists of high-level states which synthesize the status of an entire subsystem, such that data transmission over the network is minimized.
- Altairis provides a CORBA interface to allow authorized processes to "listen in" on data as it passes through the system.

The Web-Altairis server utilizes the CORBA interface to receive a copy of all data that is directed to the Altairis console. Since the volume of this data can be large, the server compresses text messages and performs other data optimizations to minimize network traffic before sending the data to the client.

Once the data reaches the Web-Altairis client it is uncompressed and displayed to the user. The user interface closely resembles the Altairis console (see Figure 2).

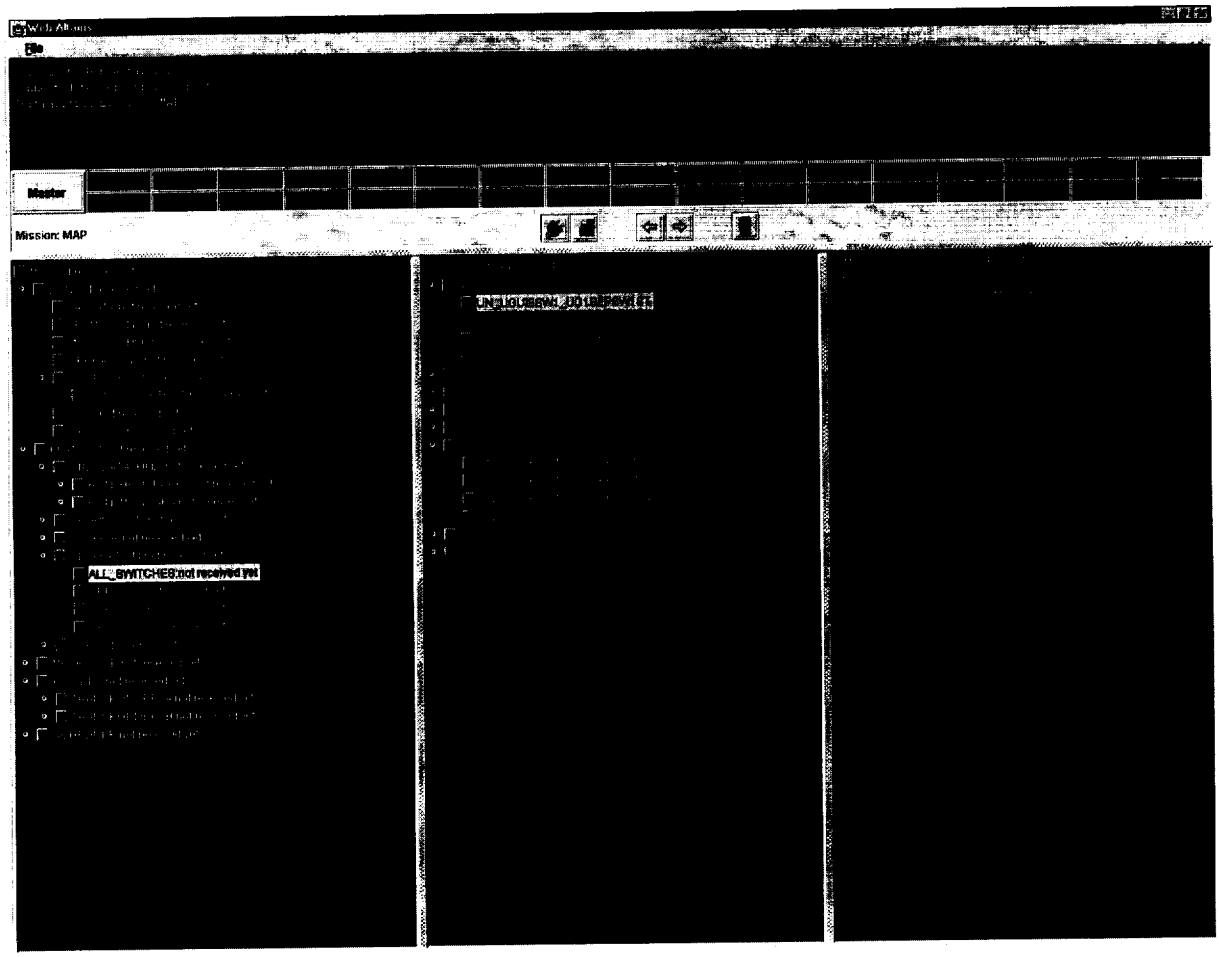

Figure 2: Web-Altairis Console

The user interface includes:

- a message window that provides system-level messages,
- a status bar that provides a high-level summary of the state of the major subsystems,
- a subsystem browser to allow the user to drill down into the state hierarchy and find the state of all the subsystems, and
- a window that can host detailed visualizations.

The Web-Altairis server runs in the control center, co-located with the Altairis host system, and therefore must run on the same computer platform as the control center. Likewise, the Web-Altairis client must be able to run on whatever platform a remote operator may have available. Since both the client and server are implemented in Java, both run on many platforms that an operator or ground system is likely to employ. The benefits of Java go beyond platform independence, though. Java's support for distributed computing and network security will be discussed in detail in the section entitled "Security of Transmitted Data".

Issues and Challenges

This section highlights the major issues and challenges that we faced during the design and development of Web-Altairis:

- How do we provide a new interface to an existing ground system?
- How can we protect the security of data transmitted over a public network, especially usernames and passwords?
- How can we securely authenticate remote users?

- What restrictions do we place upon remote users? For example, who is actually allowed to command the satellite?
- Can we optimize client/server communication so that a client workstation running over a dial-up link to the Internet will be reasonably responsive?

Interfacing With the Ground System

To be accessible remotely, a ground system must (a) support a remote Application Programmer Interface (API), and (b) provide the hardware connectivity for remote access.

The API concern is already addressed by many modern ground systems. For instance, the ITOS ground system includes a process– the Data Point Server– that provides ground system information to external processes via a published protocol[7]. Other existing systems provide APIs based on sockets or CORBA.

The connectivity requirement is non-trivial because the computers comprising a ground system are generally heavily protected from external access. They are often on private networks that either have no connection to the outside world or else are behind firewalls which strictly control what data passes into and out of the ground system.

There are several approaches to provide connectivity to these systems, including:

- A dial-up capability where the system accepts, perhaps via a callback mechanism, modem connections from the outside.
- A validation procedure to allow connections through a firewall.

Connectivity to the MAP mission ground system will be accomplished in two different ways over the course of the mission. At the time of this writing, the MAP mission is still in the integration and test phase. During this period, the ground system is on an engineering network that is accessible to the PCs in the operators' offices and that also supports a bank of modems for dial-up connections from a private residence.

Before the mission goes operational, the ground system will run on NASA's private NASCOM network. NASCOM enforces tight security guidelines covering the transfer of data on or off of the network. Connectivity during this phase of the mission has not yet been implemented, but early plans for connectivity center around the use of a callback mechanism whereby operators dial into a modem that records their phone number and disconnects, and then the ground system initiates a modem connection back to the same number on a secure phone line. Once connected on the secure phone line, the user would then be required to authenticate themselves with a username and password before receiving access to the ground system.

Security of Transmitted Data

One major concern when transferring sensitive data over a public network like the Internet is ensuring that no one can intercept the information. This practice, often called *sniffing*, occurs when curious parties inspect, copy, and save packets being sent over a network. Unscrupulous individuals may even use the same means to discover the usernames and passwords that are sent between clients and servers during logins.

A second concern when using a public network is that a malicious outside party may try to fool the ground system server into accepting data packets that hold malicious content. This is called *spoofing* and occurs when a malicious party creates data packets that appear to have originated from the appropriate client and can insert them into the data stream running between the client and the server. This may fool the server into executing unexpected or malicious commands. If these threats to privacy and security cannot be eliminated in the case of Web-Altairis, then this software would not be suitable for access over a public network.

---

[7] For more detailed information regarding ITOS, see http://itos.gsfc.nasa.gov/

Web-Altairis addresses concerns like sniffing and spoofing by encrypting its data via the Secure Socket Layer (SSL) protocol for client/server communication. SSL uses public- and private- key encryption technologies to encrypt all TCP packets on the user's machine before they are sent across the network. Curious parties are still able to intercept and sniff the packets themselves, but are unable to decipher the encrypted data. Further, since the data packets are encrypted using a private key, when a third party tries to insert a packet into the data stream, the server immediately recognizes it as a bogus packet. This eliminates the threat of successful spoofing.

As with any encryption technology currently available[8], it is theoretically possible to break SSL encryption given enough computing power and time. Threats of this nature can be managed, however, by periodically changing the encryption algorithm's private keys.

Authenticating Remote Users

Establishing a connection from a Web-Altairis client to a Web-Altairis server requires the remote operator to log in with a username and password. Although the server receives the username and password from the client, it does not itself perform the authentication. Instead, the server forwards the username and password to the ground system, which in turn validates the login and notifies the Web-Altairis server. If validation fails then the server terminates its connection to the client.

Deferring user authentication to Altairis is a design that ensures that Web-Altairis will never compromise the security of the ground system: the ground system itself must grant permission for each connection made by Web-Altairis. This means that accounts are managed from within the ground system alone, not in two different places. It also means that the operators using the ground system can place restrictions on remote connections while they work, and those restrictions take effect immediately.

It is possible that an operator's username and password may accidentally fall into the wrong hands. This is a pitfall for any password-based system, but is manageable. For instance, there are One Time Password technologies (OTP) that require the user to provide a different password at each login.

Commanding

There are technological and ideological reasons that both support and contraindicate the notion of commanding a ground system remotely. Ideologically, remote commanding is buttressed by the fact that NASA has already placed an emphasis on the need for a ground system that can operate in a "lights out" mode. If operators of future ground systems will not *always* need to be present, it seems a natural progression that operators should *hardly ever* need to be present. However, due to the security risks involved, the prevailing sentiment among mission control staff is that commanding over a network is intrinsically unsafe and inadvisable.

The Web-Altairis client provides an interface for passing commands to the ground system. The user must re-enter his or her password with each command. Web-Altairis does not itself process commands, but simply forwards them to the Altairis ground system, along with the operator's username and the newly entered password. Altairis then authenticates the user and verifies that the user possesses sufficient privilege to execute the given command. If both of these requirements are met then Altairis executes the command.

This scenario only occurs if the ground system is configured to accept commands. The option to command from Web-Altairis is provided, but can be disabled. Altairis can also be configured to delineate between the privileges available to various remote users. Some remote operators, for instance, may be allowed only to use the Web interface as a tool to view the current state of the ground system. Other more privileged users may be allowed to execute commands. Again, this is determined by the Web-Altairis server in conjunction with the Altairis ground system.

---

[8] We used the IAIK implementation of SSL. For information about IAIK, see http://jcewww.iaik.tu-graz.ac.at/products.htm.

The risk with remote commanding is that an unauthorized party will be able to send commands to a given system. There is no technological way of absolutely preventing this. The security provided by passwords fails if the passwords fall into the wrong hands. The security provided by encryption fails when the encryption algorithm is broken; and all modern encryption algorithms are breakable given sufficient time and resources. At the current time, physical security procedures that are meticulously observed are the only certain means of thwarting all malicious attempts.

Performance

There are several aspects to the performance of a system such as Web-Altairis. The one that has driven most of our research is the performance perceived by the remote user: we want remote users to be able to view the data that is important to them in realtime or near-realtime. The element that has the most effect on the realtime performance of the Web-Altairis client is the effective speed of the network between the client and server.

Other elements of the system that affect realtime performance include:

- the user's computing platform,
- the execution speed of the Java Virtual Machine running the client, and
- network constraints between the ground system and the Web-Altairis server.

Most users realize that in periods of high utilization, any network– public or private– will drop in perceived performance. Furthermore, Web-Altairis from a dial-up connection has its bandwidth limited by the speed of the user's modem.

Because modem usage was envisioned for Web-Altairis since the early stages of its design, the server and client both compress transmitted data. These compression algorithms require Web-Altairis to use more CPU power than would otherwise be necessary, but enables it effectively to reduce the amount of bandwidth required to deliver realtime data.

The client platform will naturally have an effect on the performance of the software. A Java Virtual Machine running the Web-Altairis client requires between 20 and 30 megabytes of memory. This is not an unusually high memory requirement for a Java program, but performs slowly on a limited (32 megabyte) platform. Web-Altairis was developed on PCs with 128 MB of memory and CPUs running at 450 MHz; these machines are more than sufficient to run the Web-Altairis client smoothly.

Another factor that might eventually limit the performance of Web-Altairis (both client and server) is the fact that it is written in Java. Java is a highly platform-independent language, but tends to execute more slowly than programs compiled specifically for a particular platform (like programs written in C and C++).

Currently WA has only encountered Java-related sluggish performance in the area of 3D data visualizations. Rendering 3D visualizations in realtime is computationally intensive and sometimes the user can notice an overall drop in the client's performance when a 3D visualization is running. There is no doubt that a C++ Web-Altairis implementation would allow these sections of code to execute faster, but the concomitant portability issues would be prohibitive.

Maintainability

Maintainability is an important issue for any ground system; and accessing a ground system across a network brings about its own additional, unique challenges. In the case of Web-Altairis, the most significant maintenance issue arose from the fact that VisAGE mnemonic configuration depends upon the Altairis Finite State Model. As the model evolved, so must the mnemonic configuration file. We addressed this issue by developing a utility which converts the Altairis state model into the VisAGE configuration file. This was not a trivial endeavor, but was required not only to connect Altairis to VisAGE in the first place, but to facilitate iterative propagation of changes in the state model into VisAGE. This utility greatly improved the maintainability of Web-Altairis, although even with the

utility, changes in the state model still require that ground system staff shut down the server, apply the conversion utility, and restart Web-Altairis.

A further maintenance issue was more fundamental to Internet-enabled ground system access. Quite simply, because the Web-based interface is designed to resemble the Altairis ground system console, changes to the Altairis user interface must be reflected in the Web interface. However, for Web-Altairis, this was not a significant issue– the reason for this is two-fold:

- Web-Altairis was implemented after the Altairis user-interface was complete and stable. Clearly, developing a Web-based interface to resemble a ground system interface which is itself constantly evolving is an unsound development strategy.
- Much of the Altairis user interface is dedicated to browsing the Finite State Model; therefore the interface is always synchronized with the current state of the model as received from the server.

Technologies

Several technologies contributed to the feasibility of Web-Altairis:

- the Altairis ground system provides an innovative Finite State Model which reduces the amount of data that needs to be transmitted to a remote client;
- the Java platform provides a rich, portable, network-enabled software infrastructure;
- SSL provides data encryption and other security measures; and
- VisAGE provides data distribution capabilities and visualization tools.

Any Internet-accessible ground system will benefit from the continuous maturation of Internet technology. Web-Altairis will also benefit from improvements to the VisAGE product itself. As new visualizations are added to the VisAGE visualization suite, they will become available to any VisAGE-based solution. Currently, VisAGE development is underway on visualizations suitable for viewing an entire constellation of satellites. When this becomes available in a future release of the VisAGE product, such visualizations can be applied to future products which provide remote access to a satellite constellation ground system.

Conclusions

The viability of an Internet-enabled ground system is directly related to the maturity of the technologies upon which it is based. For example, Web-Altairis is only as secure as Java SSL and only as portable as the Java platform itself. Similarly, as the Java2D and Java3D graphics packages improve in capability and speed, a user interface employing these technologies can be brought to increasingly low-profile client platforms such as PDAs and cellular phones. Simultaneously, these same platforms will themselves increase in capability and speed. As these technologies converge, remote access to ground systems will increase in feasibility, usability, and security. Coupled with the trend toward lights out, autonomous ground systems, we believe Internet-enabled ground systems will gain increasing acceptance within the aerospace community.