Internet Access to Spacecraft

James Rash
Goddard Space Flight Center
Greenbelt, MD 20771 (301) 286-5246 James.Rash@gsfc.nasa.gov

Ron Parise, Keith Hogie, Ed Criscuolo, Jim Langston
Computer Sciences Corp (CSC)
7700 Hubble Dr. Lanham-Seabrook, MD 20706 (301) 286-3203
Ron.Parise@gsfc.nasa.gov, Keith.Hogie@gsfc.nasa.gov, Ed.Criscuolo@gsfc.nasa.gov, jlangsto@csc.com

Chris Jackson
Surrey Satellite Technology Ltd. (SSTL)
University of Surrey
Guildford, Surrey GU2 5XH, U.K. (011) 44-1483-259141 c.jackson@surrey.ac.uk

Harold Price
VyTek Wireless Inc.
1121 Boyce Road, Suite 400, Pittsburgh, PA 15241 (724) 942-1085 hprice@vytek.com

Abstract. The Operating Missions as Nodes on the Internet (OMNI) project at NASA's Goddard Space flight Center (GSFC), is demonstrating the use of standard Internet protocols for spacecraft communication systems. This year, demonstrations of Internet access to a flying spacecraft have been performed with the UoSAT-12 spacecraft owned and operated by Surrey Satellite Technology Ltd. (SSTL). Previously, demonstrations were performed using a ground satellite simulator and NASA's Tracking and Data Relay Satellite System (TDRSS). These activities are part of NASA's Space Operations Management Office (SOMO) Technology Program. The work is focused on defining the communication architecture for future NASA missions to support both NASA's "faster, better, cheaper" concept and to enable new types of collaborative science. The use of standard Internet communication technology for spacecraft simplifies design, supports initial integration and test across an IP based network, and enables direct communication between scientists and instruments as well as between different spacecraft.

The most recent demonstrations consisted of uploading an Internet Protocol (IP) software stack to the UoSAT-12 spacecraft, simple modifications to the SSTL ground station, and a series of tests to measure performance of various Internet applications. The spacecraft was reconfigured on orbit at very low cost. The total period between concept and the first tests was only 3 months. The tests included basic network connectivity (PING), automated clock synchronization (NTP), and reliable file transfers (FTP). Future tests are planned to include additional protocols such as Mobile IP, email, and virtual private networks (VPN) to enable automated, operational spacecraft communication networks. The work performed and results of the initial phase of tests are summarized in this paper. This work is funded and directed by NASA/GSFC with technical leadership by CSC in arrangement with SSTL, and Vytek Wireless.

Introduction

This paper will discuss the use of standard Internet applications and routing protocols to meet the technology challenge of providing dynamic communication among heterogeneous instruments, spacecraft, ground stations, and constellations of spacecraft. The objective is to characterize typical mission functions and automated end-to-end transport of data in a dynamic multi-spacecraft environment using off-the-shelf, low-cost, commodity standards. This capability will become increasingly significant in the years to come as both Earth and space science missions fly more and more sensors and the present labor-intensive, mission-specific techniques for processing and routing data become prohibitively expensive. This work is about defining an architecture that allows science missions to be deployed “faster, better, and cheaper” by using the technologies that have been extremely successful in today’s Internet.

Internet Protocols in Space Overview

The goal of the OMNI project is to define and demonstrate an end-to-end communication architecture for future space missions. The authors have combined their knowledge and experience in Internet technologies and space communication systems in developing the following end-to-end data communication concept.

End-to-End Network Concept

The key to the whole architecture is the use of the Internet Protocol[1] (IP) to provide a universal communication capability among all space and ground systems for future missions. IP is the technology that the entire Internet runs on. It provides a basic standardized mechanism for end-to-end communication between applications across a network. The protocol provides for automated routing of data through any number of intermediate network nodes without affecting the endpoints.

Network addresses define endpoints. A network needs a mechanism for maintaining routing tables that direct the flow of data across the network. Routing protocols such as Router Information Protocol[2] (RIP), Open Shortest Path First[3] (OSPF), Border Gateway Protocol[4] (BGP), and Interior Gateway Routing Protocol[5] (IGRP) are currently used to automatically maintain the routing tables in the Internet. These protocols assume that nodes on the Internet are stationary, and that the only reason for learning new routes is to adjust to individual link outages.

Spacecraft environments seem to pose numerous additional challenges over earth-bound networking, such as:

- continually intermittent links
- multiple mobile nodes forming a dynamic network topology
- maintaining a single address for a spacecraft as it uses different ground stations
- highly asymmetric or unidirectional communication links

However, there are parallels to the spacecraft environment in the developing wireless networking community. For example, the increasing popularity of laptop computers, handheld digital assistants, and Internet cell phones has driven the development of protocols to handle mobile nodes, such as Mobile IP[6] (MIP) and mobile routing. They are also driving the development of new protocols such as Cellular IP[7] and Dynamic Source Routing[8] (DSR) and other ad-hoc-networking protocols.

This paper will examine standard Internet applications and protocols specified by the Internet Engineering Task Force (IETF), and map them to spacecraft applications. It will also describe how standard, commercially available communication hardware and software was used to test some of these concepts with an orbiting spacecraft.

Benefits

Increasingly, future space missions featuring multiple spacecraft are being proposed. This includes constellations (disjoint or formation-flying), sensor-webs[9] of heterogeneous spacecraft, and collaborative science between spacecraft belonging to different missions. As the number of spacecraft involved increases, the number of possible end-to-end routes for data goes up geometrically. The present methods utilizing manual data routing, non-interoperable protocol options, and custom data handling applications are not scalable and rapidly become unaffordable due to the large amount of manpower and custom development required.

Using standard Internet protocols will allow robust, dynamic, and automated end-to-end data delivery. Using standard Internet applications, such as FTP[10], will allow exchange of data without designing custom data handling and translation software. These will reduce the risk and development time for space missions, increase the accessibility of science data, and enable cost-effective realization of new science capabilities such as:

- Data exchange directly between spacecraft
- Correlation of data in space
- Collaborative science among unrelated sensors within and across missions
- Easy reconfiguration and deployment of new types of collaborative science across multiple missions
- Direct access to science payloads and data by principal investigators or collaborators

Along with these new capabilities, significant benefits are envisioned across all phases of a mission life-cycle. Significant cost savings and risk reduction are achievable in all the following areas:

- Mission Design
- Software Development
- Hardware Development
- Testing and Integration
- Flight Operations

Mission design will be faster and simpler by using much more standardized communication interfaces. This will allow designers to spend less time doing data communication system design and focus more on the specific science and mission details. Minimizing mission specific interface control documents (ICDs) will expedite design and reduce costs.

Software design and development will be able to utilize a large amount of software and services already defined, documented, implemented, and tested in commercially available operating systems and applications.

Using a common suite of standard communication interfaces for spacecraft subsystems will allow development of radiation hardened, standard local area network components. This will simplify development and integration of spacecraft systems.

Using Internet technology as a common communication mechanism from spacecraft subsystems to the end user will allow very early functional testing of subsystems to identify problems long before traditional integration and test. Catching problems earlier can provide significant cost savings and risk reduction.

All of the advantages from the earlier mission phases also carry over into the mission operation phase. Using the same communication interfaces from initial design and development through operation allows the same monitoring and control software and knowledge bases to be used across the entire mission life-cycle. This avoids developing and testing new interfaces and software systems for initial development, integration testing, and operation.

Prototype Implementation

The OMNI project had been looking for opportunities to demonstrate IP in space for the last year. However, many of the spacecraft candidates were deemed unsuitable due primarily to their onboard communication hardware. The key issue was to find a spacecraft that could support HDLC[11] framing in hardware. Based on its near-universal use on the terrestrial Internet, the OMNI project had chosen to use HDLC framing for the link-level protocol on space-to-ground links. This allowed simple, straightforward interfacing with existing commercial routers. By choosing the IETF encapsulation for multi-protocol over frame-relay/HDLC, we could insure interoperability and avoid being tied to one manufacturer's implementation. These choices made UoSat-12 (figure 1) an ideal test platform, as it already used HDLC framing to carry its AX.25 protocol. Since HDLC I/O hardware was already present on-board, only flight software changes would be required to adapt UoSat-12 to use IP. Changes to the ground station would also be minimal, requiring only the addition of a standard commercial router and a programmable switch. See figure 4.

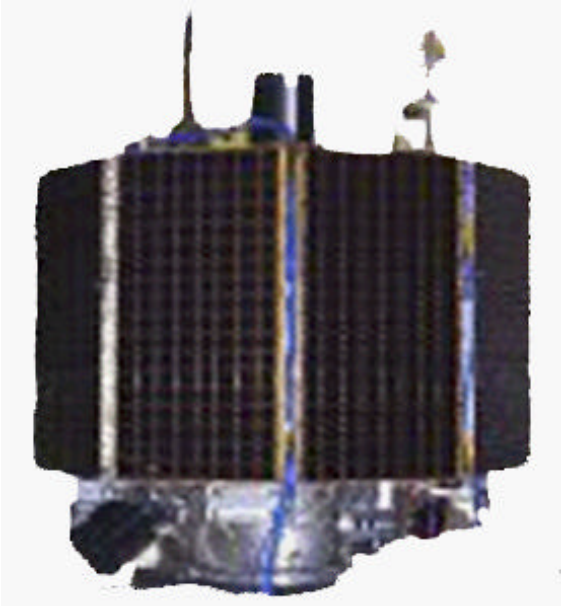

Figure 1 – UoSAT-12 Spacecraft

In December 1999, The OMNI project contractor, Computer Sciences Corp. (CSC), began negotiations to have an IP stack ported to one of the spacecraft's onboard processors, using HDLC/Frame-Relay for the link-layer protocol over the RF communication system. This would allow the ground station to directly connect the receiver to a standard low-cost router, providing end-to-end IP connectivity between an IP address on the spacecraft and any node on the Internet. Further discussions covered porting File Transfer Protocol (FTP) and Network Time Protocol[12] (NTP) to the spacecraft's operating system.

In February 2000, CSC let an initial contract to VyTek Wireless of Pittsburgh, PA, formerly VyTek LLC, to supply the software porting, spacecraft time, and ground-station time for a series of flight tests. Initial tests of IP connectivity were scheduled for early April 2000, with tests of spacecraft clock synchronization, reliable file transfer, and blind commanding to follow in May/June 2000.

Follow-on work is planned to demonstrate http file delivery, mobile IP, security, and store-and-forward commanding and data delivery using Simple Mail Transfer Protocol[13] (SMTP). These tests are expected to be performed in 3Q/4Q 2000.

Spacecraft Implementation

Since the UoSAT-12 spacecraft was already in orbit, that part of the IP implementation could not require any hardware changes. The spacecraft was built to be a flexible prototype test environment and it was easy to upload and test new software to support IP and HDLC.

HDLC at the Physical Layer

The UoSat-12 spacecraft uses hardware to perform the HDLC framing. At the physical layer, HDLC framing is extremely simple, consisting of only a 1-byte flag pattern, a variable number of data bytes, and a 2-byte CRC. See figure 2. The end of the frame is identified by another 1-byte flag pattern. This is allowed to be the flag pattern for the start of the next frame. During any idle time, successive flag bytes are output until the next frame begins. Thus, HDLC frames are always separated by one or more flag bytes.

Flag bytes consist of a zero bit, 6 one bits, and a zero bit. In order to prevent this pattern from occurring in the data, the HDLC hardware performs "bit stuffing" when sending data. Any sequence of 5 one bits in the data automatically has a zero bit inserted after it, thus insuring that any sequence of 6 consecutive one bits *must* be a flag byte. When receiving, these extra zero bits are automatically removed from the data.

While the primary purpose of "bit stuffing" is to ensure the uniqueness of the flag byte, it also has an additional benefit. It ensures that a long unbroken sequence of one bits in the data does not produce a signal to the transmitter that does not have periodic transitions. These periodic transitions are important at the receiver, where a bit-synchronizer depends on them to extract the clock and data bitstreams from the raw signal. Along the same lines, UoSat-12 also uses standard NRZI coding for the HDLC output. NRZI will insure that an unbroken sequence of zero bits in the data stream becomes transformed into an alternating sequence of ones and zeros. Thus, the use of "bit stuffing", idle flag bytes, and NRZI coding insures that the transmitter will never send an unmodulated carrier, and the receiver will see a transition *at least* once every 6 bit times. It is important to note that these requirements were able to be met by standard COTS hardware and protocols without inventing any "space specific" solutions. It should be further noted that these solutions are isolated to the lowest layer and are transparent to the upper layers. The application layer does not have to concern itself with generating "fill packets" or "fill frames".

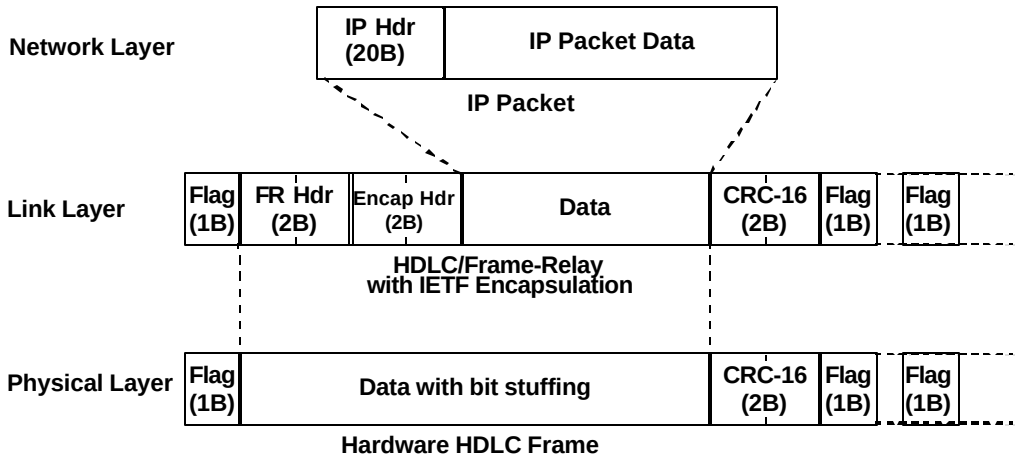

Figure 2 - HDLC/Frame Relay/IP packet formats

HDLC at the Link Layer

As previously mentioned, the OMNI project had selected HDLC/Frame-Relay with IETF multi-protocol encapsulation as the link layer protocol of choice. In order to modify the UoSat-12 flight software to use IP, an IP stack had to be ported to UoSat-12 and a link-layer driver written and added to that stack. Because of the hardware HDLC support, the link layer driver was fairly simple, requiring only the addition of the frame-relay and encapsulation headers, most of which consisted of fixed bit patterns to indicate a fixed Data Link Channel Indicator (DLCI) and an encapsulation of IP over frame-relay.

IP Stack in SCOS

The UoSat-12 spacecraft uses the SCOS operating system, developed by VyTek Wireless. SCOS has been part of 34 small spacecraft missions, 16 are currently operational, and 8 are being prepared for launch. SCOS is a preemptive dispatch multitasking operating system with support for dynamic reloading of modules from the ground, or from onboard "ram disk" storage.

To add IP capability to the UoSAT-12 spacecraft, VyTek ported the relevant sections of FreeBSD version 3.2, which is based on Berkeley 4.4 BSD developed by the University of California, Berkeley and its contributors. This consisted of approximately 12,000 lines of C code, including the NTP client and FTP server. Most of these lines were unmodified but the porting did include dealing with 16 bit vs. 32 bit integers. The majority of the work was in:

- developing a device interface for the FreeBSD IP stack that made use of the "raw HDLC" interface that was added to the existing SCOS AX.25 drivers.
- Adding a "sockets" interface callable by user programs.

The FTP server was recoded to fit better in the SCOS environment. The FreeBSD code forked a new process for each concurrent user and was not an efficient use of SCOS resources. The result of the port was about 140K bytes of executable code (including FTP and NTP) and 60K bytes of data (including packet buffers).

The software was tested using a spacecraft simulator at VyTek. The simulator is an I/O co-processor card from IBM called ARTIC. It uses an 80186 CPU and two Zilog 8030 SCC chips (providing hardware HDLC). From the point of view of the SCOS kernel, this is similar to the UoSAT-12 hardware (80386 and Zilog ISCC). The various test scripts developed at GSFC were first tested through the Internet using the simulator at VyTek, as shown in figure 3.

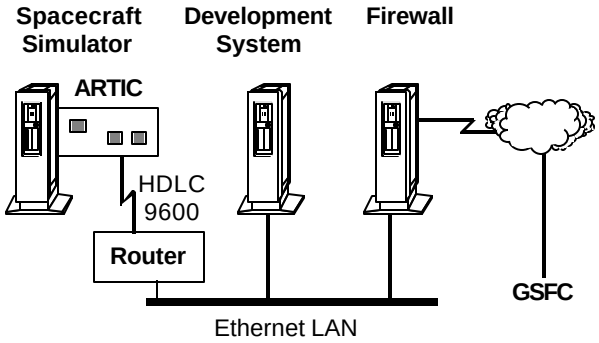

Figure 3 – VyTek Development Environment

Once the testing was complete, the executable files were send to SSTL, where they were uploaded to the spacecraft by the operators. When major changes were made, SSTL would first check the software by loading it on a engineering model of the UoSAT-12 computer, small tweaks were generally uploaded without additional testing. UoSAT-12, like all of the UoSATs, can be quickly reloaded, in many cases from onboard file storage. In cases of low risk, testing is sometimes done on board, greatly reducing the overall costs of updates. This is especially important on research and development missions like UoSAT-12.

The SCOS design philosophy and a TCP/IP stack fit nicely together. Many of the TCP protocols are peer to peer, meaning a program on one host computer somewhere on the net is interacting with a program running on a host computer elsewhere on the net. There is no single gatekeeper program running on hosts in the Internet environment to parcel out multiplexed access to the communications links. The TCP/UDP/IP stack performs all of the routing and multiplexing functions needed.

Likewise, an SCOS software stack is usually made up of independent (though interacting) processes (called tasks), each interacting with a peer on the ground, for example, telemetry tasks, GPS, attitude control, imaging control, etc. Each task has its own address on the uplink/downlink, and can be separately addressed on the link. In past missions, the address features of the AX.25 protocol were used. In the UoSAT-12 OMNI tests, the port number features of TCP and UDP were used.

Addressing processes on a host using ports is very natural way of communicating through the Internet or an intranet, and therefore make maximum use of existing COTS hardware and software, and more importantly, the many years of experience and work embodied in the Internet protocols.

Ground Station Implementation

Since the SSTL ground station already supported HDLC framing, a standard Internet router was the only addition needed. Figure 4 indicates the basic components of the ground station and where the router was added in parallel with the existing AX.25 communication front-end.

The serial interface on the router uses a standard RS-530 connection with balanced, synchronous clock and data lines for transmit and receive data. The modem/receiver lock signal is connected to the DCD line on the router. This allowed the receiver lock indication to be monitored remotely by someone logged into the router. During a normal UoSAT-12 pass, the SSTL transmitter is in standby and a push-to-talk mode is used as needed. When the station is configured for IP mode the transmitter is on all the time to provide continuous, full-duplex operation.

Since the basic HDLC framing is identical for both AX.25 and frame relay, the receive clock and data lines are simply split off to both the AX-25 front-end and the routers. When UoSAT-12 is sending AX.25 format data, the router doesn't recognize the AX.25 headers inside those HDLC frames, so it just counts the HDLC frame and discards the data. The AX.25 front-end treats the IP/frame relay packets similarly. By using standard HDLC for both modes there are no receiver reconfigurations needed to switch between modes.

The only station reconfiguration required is to select which system is connected to the transmitter. This is done with a controllable switch and supports fully automated passes for either the IP or AX.25 mode.

The SSTL ground station is built on an Ethernet LAN with firewalls and router connectivity to the Internet. Two addresses were used on the ground station LAN to support these tests. One address was used for the Ethernet interface on the router and the other address was assigned to the spacecraft. The router was configured to route data for the spacecraft address out the serial port. This enabled Proxy ARP for that address the . This means that when any system on the Surrey LAN sent out an Address Resolution Protocol (ARP) packet to find the Ethernet address that corresponded to the spacecraft address, the router would respond and data would be sent to it. The router would then forward the packets on the serial interface. Any data coming in the serial interface would be forwarded based on its destination address using standard IP forwarding.

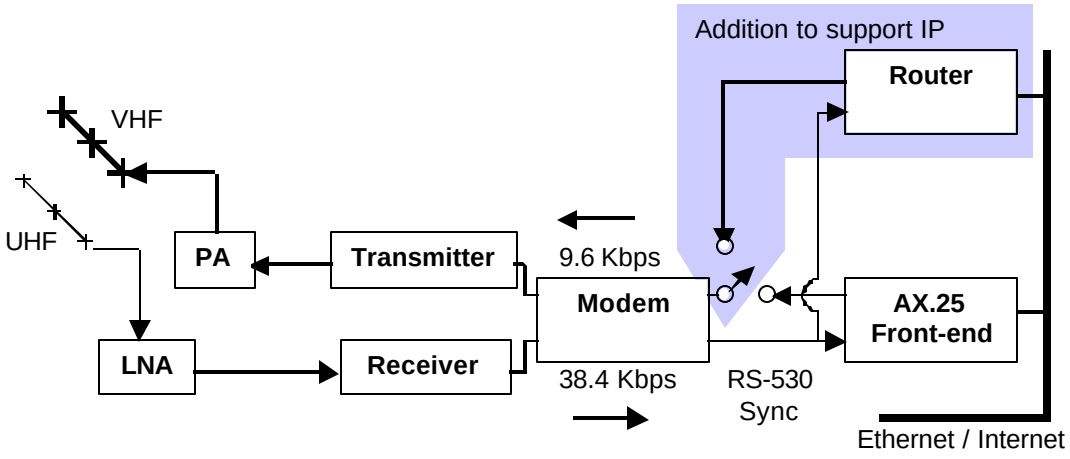

Figure 4 - SSTL ground station configuration

Current Tests and Results

The following tests were performed to demonstrate functionality of Internet Protocols with an on-orbit spacecraft.

Basic Internet Connectivity

The initial IP tests with UoSat-12 were designed to verify the operation of standard Internet packet routing from the spacecraft's operating system to anywhere on the Internet. The "ping" utility was used to accomplish this and to characterize the basic link propagation delay. The test configuration used is shown in figure 5. Five simultaneous data-capture sessions were run:

1. Local Ping - A ping session was run from the Cisco router at the SSTL ground station to UoSat-12. This demonstrated connectivity and measured the basic link delay.
2. Remote Ping - A ping session was run from a workstation at GSFC all the way to UoSat-12. This demonstrated end-to-end packet routing from user to spacecraft through approximately 20 hops across the open Internet.
3. Router Statistics - The interface, link, and network statistics on the router at SSTL were captured every 30 seconds.
4. Ground Internet Delay - A ping session was run from a workstation at GSFC to the router at SSTL. This provided a measurement of the terrestrial Internet's latency and provided a cross-check on the differences between the local ping and remote ping sessions.
5. Groundstation Log - During the pass, groundstation parameters such as antenna az/el, signal strength, and range data were captured.

All captured data was timestamped, and the time clocks in SSTL and GSFC were kept synchronized by using NTP to synchronize to the US Naval Observatory's timeserver in Washington DC.

The results for the test run on April 11, 2000 are shown in figure 6. The scheduled UoSat-12 pass had an AOS of 16:38:21 UTC and an LOS of 16:57:43. This was a very clean test, with pings beginning right at 0 deg. elevation, and no significant anomalies other than the expected dropouts near LOS due to antenna masking. This high-elevation pass (71 deg. max) produced a clearly visible 15 ms. shift in the round-trip times between minimum and maximum range.

The chart in figure 6 has five separate quantities plotted. Starting from the bottom, the bottom (orange) data series is antenna elevation in degrees vs time. The scale is shown on the right. All other series use the seconds scale on the left. The second (green) data series is the theoretical minimum round-trip-time (RTT). This is calculated based on the packet size, uplink and downlink rates, and the slant range to the spacecraft. The third (blue diamonds) data series is a scatter plot of the 3000+ raw PING times from the SSTL router to UoSat-12. The fourth (light blue) line is a 5^{th} order polynomial fit through the raw PING times. Note the constant 40 ms. offset from the theoretical minimum RTT. This represents onboard processor latency. The fifth (red X) data series is a scatter plot of the raw PING times from GSFC to UoSat-12. Note that, in general, the ground station to satellite RTT is less than half of the GSFC to satellite RTT.

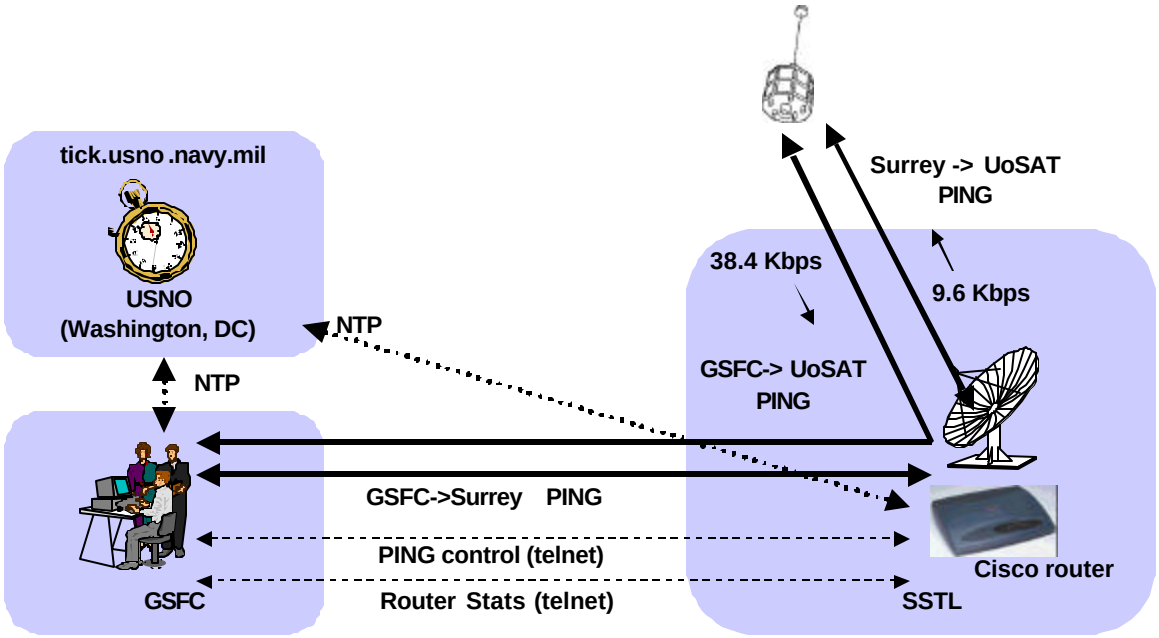

Figure 5 - PING test configuration

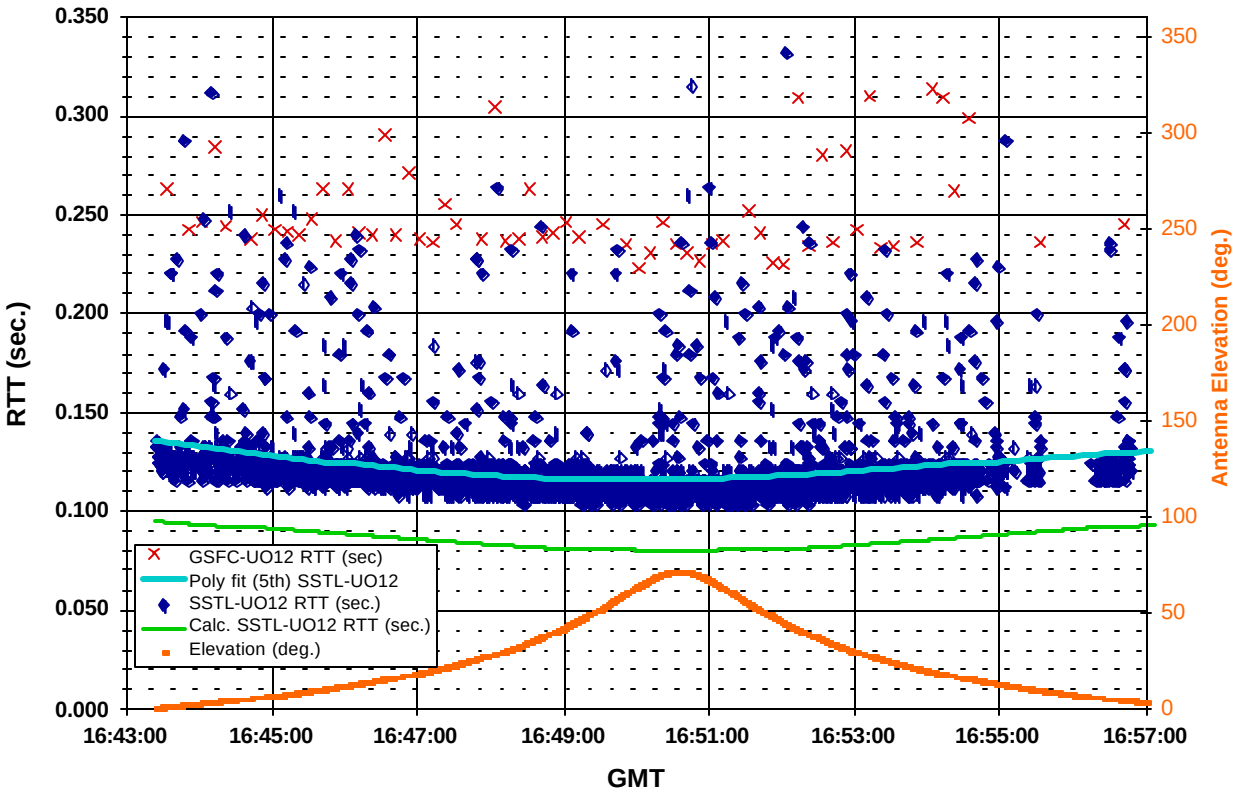

Figure 6 - PING test results

Automated Spacecraft Clock Synchronization

For the clock synchronization tests, a standard NTP client was ported to the UoSAT-12 spacecraft. It was used to automatically synchronize the onboard clock to UTC. On the ground, the US Naval Observatory's timeserver (tick.usno.navy.mil) was used as the reference timeserver. This configuration is shown in figure 7. This represents somewhat of a worst-case test, as the USNO is a quarter of the way around the world, over 20 router hops, from the UoSat-12 ground station in Surrey, UK. In a real operations environment, a timeserver of the required accuracy would be located at the groundstation to minimize the network latency and variation that NTP has to factor out. However, NTP is designed to deal with these factors, and the resulting levels of accuracy might be quite adequate for many space missions even under worst case conditions.

Two tests were performed, both following the same scenario. The test starts out with the onboard NTP server running, but disabled from actually changing the spacecraft's clock. The onboard server periodically negotiates with the USNO timeserver to factor out network delay. If it is successful, the onboard server calculates the offset it thinks it has to apply to the spacecraft's clock. This value is sent to the ground in a UDP telemetry stream, where it is logged for later analysis. For purposes of this testing, the NTP negotiation period is set artificially low to 30 seconds so that a reasonable number of data points can be collected during the 14 minute pass. A short time into the test, a command is sent to the spacecraft to enable NTP to actually change the onboard clock. NTP will require two successful offset calculations before it will adjust the clock. Later in the test, a command is sent to the spacecraft to manually set the onboard clock in error by a large amount (2-3 seconds). After two successful offset calculations, NTP should again reset the clock. If the time is off by more than one second, the spacecraft NTP client adjusts to the proper second during one adjustment period, and adjusts for fractional seconds on the next adjustment period.

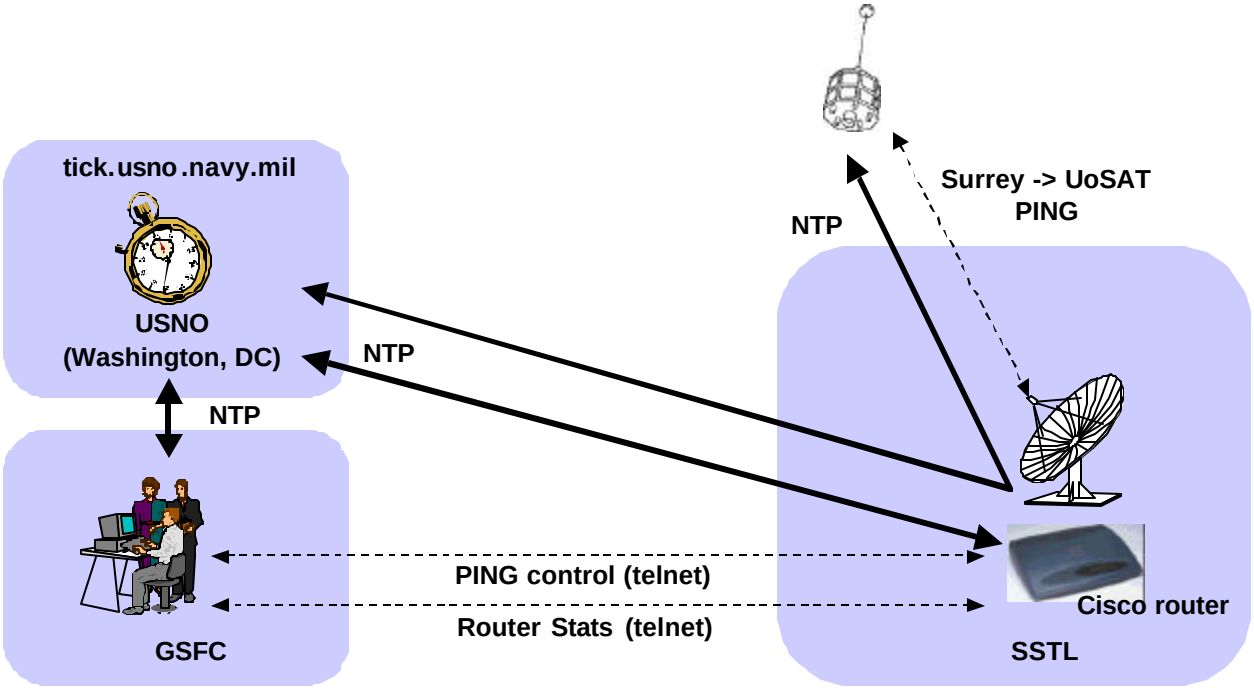

Figure 7 - NTP test configuration

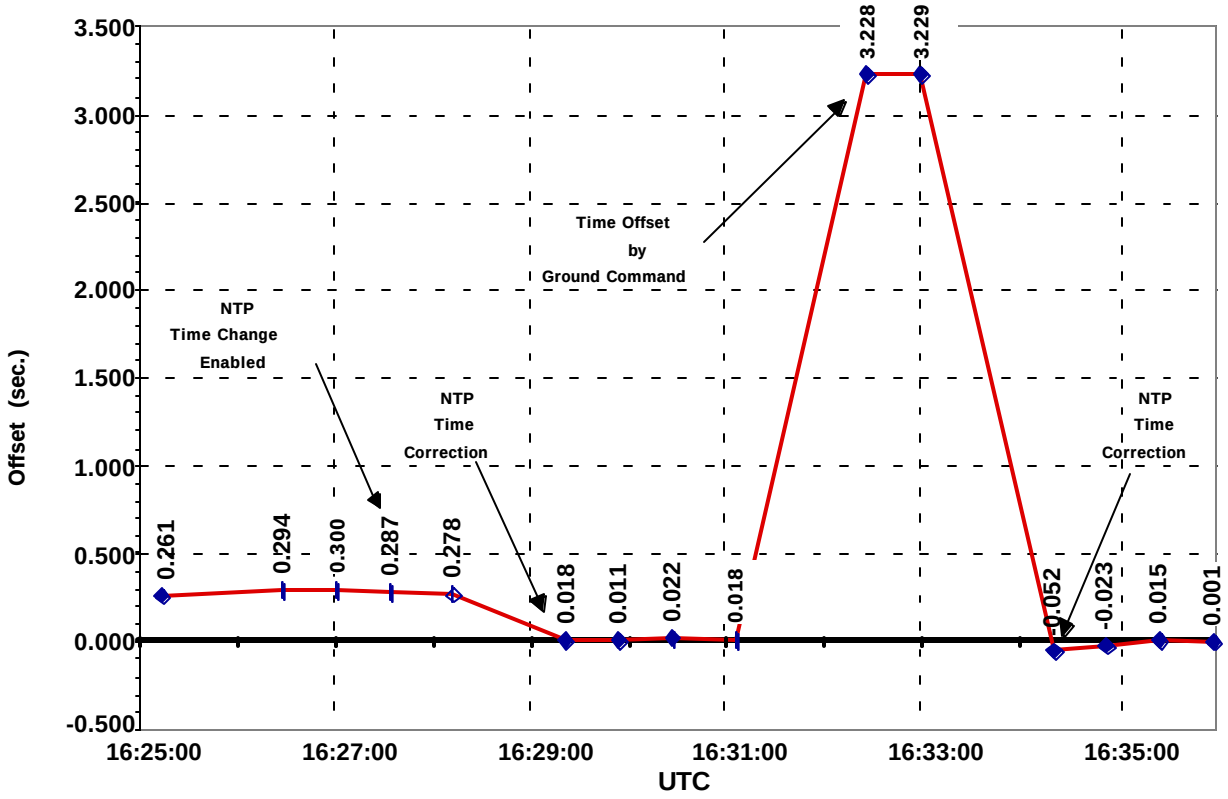

Figure 8 - NTP test results

The results for the test run on April 14, 2000 are shown in figure 8. The scheduled UoSat-12 pass had an AOS of 16:24:16 UTC and an LOS of 16:38:36. No anomalies occurred during the pass.

The pass began with a spacecraft clock offset from UTC of approximately +300 ms. Two calculations after NTP time-changing was enabled, the calculated offset dropped to less than two clock ticks (20 ms) and stayed there until it was manually set in error from the ground. A ground command was used to set the clock ahead by approximately 3.25 seconds. Two offset calculations after that, NTP had reset the clock to within six clock ticks of UTC, taking an additional two offset calculations to settle within two clock ticks of UTC.

Future Work

Testing of the File Transfer Protocol (FTP operating over the Transmission Control Protocol (TCP)) was initiated after the NTP tests. Files were successfully transferred and work is currently underway to adjust some of the standard TCP tuning parameters to optimize performance. At the completion of the FTP tests, additional software will be uploaded to UoSAT-12 to demonstrate delivery of standard telemetry and science data using UDP packets all the way from the spacecraft to a control center.

The current activities have demonstrated that standard Internet protocols will function in a space environment. However, the activities so far have been done in fairly simple and controlled configurations. More work is needed to investigate additional protocols required to properly deploy Internet protocols in world-wide, operational space communication networks. Implementing HDLC framing and IP packets on spacecraft and installing routers at ground stations are not major problems. The main issues are to deal with network security and the highly mobile aspects of spacecraft.

The OMNI project is in the process of expanding its test environment to include multiple spacecraft simulators and ground nodes for testing mobile IP and mobile routing protocols. These investigations plan to use the Linux and VxWorks operating systems on the

spacecraft simulators and Cisco IOS 12.1 or newer software on the ground routers.

Security solutions based on Internet security protocols[14] (IPsec) and virtual private networks[15] (VPNs) will be configured and tested along with the mobile IP environment.

The mobile IP and security work will focus on issues for deploying IP in operational space communication networks. Additional work is also planned to identify spacecraft control and data delivery applications to use over a space IP network. One of the main application areas to be investigated will be reliable file transfer in space environments. This will focus on file transfer applications that operate over UDP and that can then operate in communication environments with extremely long round-trip times and link bandwidth asymmetry.

A workshop to discuss Internet protocols in space will be held at GSFC in November 2000. This will be an opportunity for anyone interested in space Internet concepts to discuss issues and propose solutions.

Information on test results and future activities will be posted on the OMNI project web site at http://ipinspace.gsfc.nasa.gov/.

Conclusions

The last year of tests and demonstrations has shown that HDLC framing and IP packets provide a very simple and flexible communication mechanism for space communication. HDLC framing is well supported in a wide range of COTS products and has been used on spacecraft for over 10 years. Using the Internet Protocol as a network layer allowed easy integration and testing of our end-to-end scenarios. Also, both HDLC and IP required no modifications to operate in intermittent space link conditions.

HDLC framing provides a minimal byte overhead along with a link level error check. The variable length of HDLC framing also results in very simple data packing and unpacking since one IP packet normally ends up in one HDLC frame. A large UDP packet can be sent which will result in IP fragmentation but this is under the application programmer's control and can be completely avoided if desired. The biggest benefit of using HDLC is that it is supported on most any communication hardware that has serial interfaces.

Using the IETF multiprotocol encapsulation over frame relay has proven to be very robust and supported on every piece of communication equipment we have worked with. We have mixed equipment from different vendors on serial links, and there have been no compatibility problems. Frame relay equipment can also be used to provide basic forwarding of frames without any IP processing involved. This provides additional flexibility in deploying communication systems.

While many of the Internet protocols (i.e. TCP, FTP, NTP) work in full-duplex communication scenarios, we have also successfully used others (i.e. UDP) in either receive-only or transmit only scenarios. During the tests described in this paper, a one-way UDP based telemetry stream was used for diagnostics and statistics data. These one-way data transmission modes must be supported in order to deal with spacecraft contingencies when a full-duplex link is not available. This is just one more case of the Internet protocols being flexible enough to support a wide range of requirements.

Finally, introducing a network protocol like IP in the communication architecture has allowed us to easily support a wide range of communication scenarios and mission scenarios. Using IP has allowed us to communicate around the world and introduce new applications very quickly and easily. Most of the traditional interface control documents (ICDs) are eliminated since the Internet standards are already well specified, highly interoperable, and widely available.

Acknowledgments

The research described in this paper was carried out by personnel from Computer Sciences Corporation working under contract to NASA's Goddard Space Flight Center. The work was funded by NASA's Space Operations Management Office (SOMO) Communication Technology Project headed by Tom Costello. The authors would like to thank Cisco Systems for the loan of a Cisco 1601 router for use in the SSTL ground station. Also, the NTP time servers at the US Naval Observatory proved very useful in the NTP tests on UoSAT-12.

References

1. "Internet Protocol, DARPA Internet Program Protocol Specification", Internet Engineering Task Force RFC-791, September 1981

2. "Router Information Protocol (RIP) Version 2", Internet Engineering Task Force RFC-2453, November 1998

3. "Open Shortest Path First (OSPF) Version 2", Internet Engineering Task Force RFC-2328, April 1998

4. "BGP4/IDRP for IP---OSPF Interaction", Internet Engineering Task Force RFC-1745, December 1994

5. "Internetworking Technologies Handbook: Enhanced IGRP", Cisco Systems, Macmillan Publishing USA, 1997

6. "IP Mobility Support", Internet Engineering Task Force RFC-2002, October 1996

7. "Cellular IP - A New Approach to Internet Host Mobility," ACM Computer Communication Review, January 1999

8. "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks", Internet Engineering Task Force, Internet-Draft Mannet-DSR-03 , 22 October 1999

9. "Real-Time Information Technology Challenges for NASA's Earth Science Enterprise", G. Prescott S. Smith K. Moe, The 20th IEEE Real-Time Systems Symposium, Dec. 1-3, 1999 Phoenix, Arizona, USA

10. "File Transfer Protocol", Internet Engineering Task Force RFC-959, October 1985

11. "High-level Data Link Control (HDLC) - Frame Structure", ISO-3309

12. "Network Time Protocol (Version 3) Specification, Implementation and Analysis", Internet Engineering Task Force RFC-1305, March 1992

13. "Simple Mail Transfer Protocol", Internet Engineering Task Force RFC-821, August 1982

14. "Security Architecture for the Internet Protocol", Internet Engineering Task Force RFC-2401, November 1998

15. "Virtual Private Networks Identifier", Internet Engineering Task Force RFC-2685, September 1999