Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Networks 47 (2005) 603–650
Computer Networks
www.elsevier.com/locate/comnet

Using standard Internet Protocols and applications in space ☆

Keith Hogie *, Ed Criscuolo, Ron Parise

Computer Sciences Corporation, 7700 Hubble Dr., Lanham-Seabrook, MD 20706, USA

Available online 16 September 2004

Abstract

This paper discusses approaches for using standard Internet technologies to meet the communication needs of future space missions. It summarizes work done by the Operating Missions as Nodes on the Internet (OMNI) project at NASA/GSFC since 1997. That project arose from a small group of engineers who had been involved with building NASA communication systems for over 20 years. Since NASA needed communication systems for space long before the Internet evolved, NASA developed many custom protocols and communication techniques to meet its "space specific" communication needs. However, as the Internet evolved, it needed to address all of the same communication issues of errors, delays, and intermittent links. Those challenges may not have seemed space related, but the solutions developed can be used to address space communication issues. The key is to select the appropriate Internet Protocols that can support space communication while also providing direct interoperability with the terrestrial Internet.

This paper uses a layered approach to discuss all aspects of using Internet technologies in space. It starts with the low-level physical, data link and data routing issues related to using Internet Protocols to support basic spacecraft communications. After identifying options for supporting basic datagram delivery in space, the paper describes issues for selecting transport protocols and applications to meet various mission data delivery needs. Information is provided throughout the paper to identify key implementation issues and provide information on the current status of products in each area. Finally, current implementation and usage of these protocols in both spacecraft and ground systems are discussed.

Keywords: Spacecraft networking; Internet in space; Space communication protocols; Internet space missions; Space shuttle STS-107

1. Introduction

This paper discusses issues related to the use of standard Internet Protocols for satellite communication. It covers all protocol layers including

☆ US Government work not protected by US copyright.

* Corresponding author. Tel.: +1 301 794 2999.
E-mail address: keith.hogie@gsfc.nasa.gov (K. Hogie).

1389-1286/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2004.08.005

- lower layer protocols that deliver data onboard the spacecraft and over the space link,
- network protocols that provide global addressing and data routing among systems,
- transport protocols to support end-to-end delivery,
- application protocols to support operational needs.

These all work with the Internet Protocol (IP) [1] to provide a universal, end-to-end data communication architecture for space. This standard datagram delivery approach using off-the-shelf, low-cost, commodity-level standards will become increasingly significant in the years to come as space missions become more complex. Both Earth and Space science missions plan to fly more and more sensors and have them interact to form a "SensorWeb" [2]. Manned space flight missions in earth, lunar, and other planetary orbits are also becoming more complex and need more interoperable communication systems. The present labor-intensive, mission-specific techniques for processing and routing data do not scale well and will become prohibitively expensive. This paper 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.

The goal of the Operating Missions as Nodes on the Internet (OMNI) project at NASA's Goddard Space Flight Center (GSFC) is to define and demonstrate end-to-end communication architectures for future space missions. The authors have combined their knowledge and experience in Internet technologies, space communication, command/control, and data processing systems in developing the following end-to-end data communication concepts.

2. IP in space architecture overview

Thirty years ago, spacecraft communication was a very special business with organizations such as NASA breaking new ground and designing new protocols for the "special" space environment. However, communication technology has undergone huge changes over the last 30 years. Communication applications such as cell phones, international Internet telephone calls, and worldwide network access from a handheld computer, that once would have seemed impossible, are now in common use. Spacecraft environments still pose numerous challenges but most of these have direct analogs and solutions in the ground-based mobile IP and wireless networking industries, such as:

- intermittent communication links,
- highly asymmetric or unidirectional communication links,
- bit error rates higher than most hardwired links,
- multiple mobile nodes forming a dynamic network topology,
- maintaining a single address for a spacecraft as it uses different ground stations.

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 (MIP) [3] and mobile routing. They are also driving the development of new protocols such as IPv6 [4]. Cellular IP [5] , Dynamic Source Routing (DSR) [6], and other ad hoc networking protocols.

2.1. End-to-end network concept

Fig. 1 shows the current space communication architecture with Internet technologies added in parallel. This shows how existing missions can continue using their legacy communication techniques while also adding Internet communication support for future missions. Most NASA ground communication is carried across IP backbones so only minimal additions are needed to extend IP to the spacecraft. The major change from today's satellite communication systems is to change the format of the data onboard the spacecraft and on the space-to-ground link. Changing the spacecraft data format to match formats used by standard Internet devices allows the extensive use of COTS (Commercial Off The Shelf) networking devices in ground stations. This greatly reduces

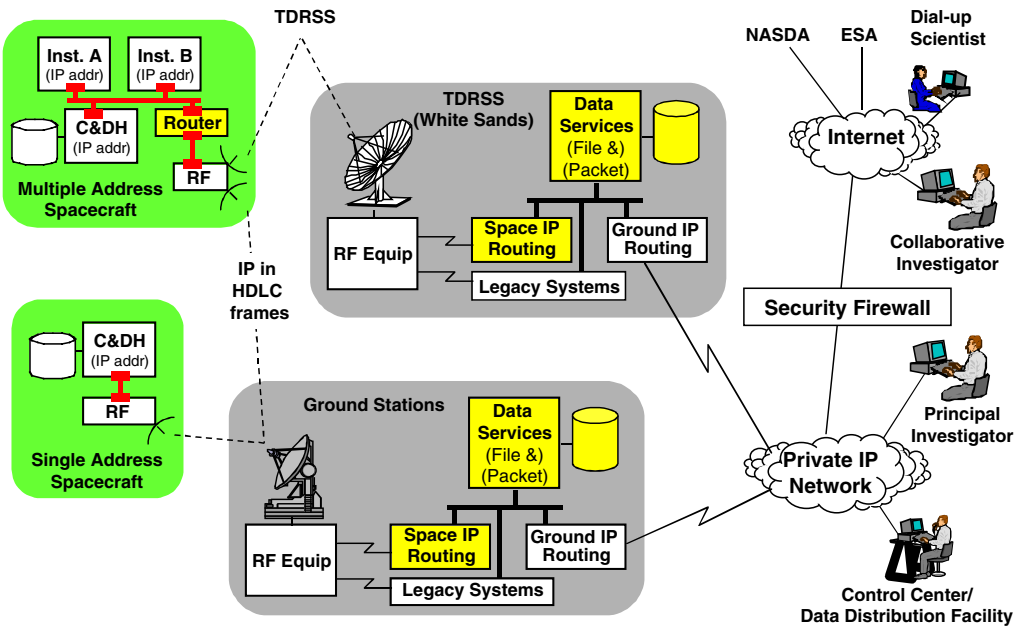

Fig. 1. End-to-end architecture using IP.

equipment cost and allows much easier upgrades to accommodate future mission needs.

The left side of the figure shows two spacecraft architecture options. The bottom one is a simple architecture that has been implemented on some spacecraft. The spacecraft only has a single IP address that is associated with its serial interface to the space link. This is easy to add to existing spacecraft designs since it only affects the interface from the onboard processor or Command and Data Handling (C&DH) system to the space link. No other onboard systems need any changes but the spacecraft can now utilize a wide range of Internet Protocols to support its operational needs and it can easily communicate with ground systems using Internet technologies.

The upper left spacecraft shows a more advanced implementation with IP extending to other subsystems onboard the spacecraft. This impacts the design of more spacecraft systems since they all need to support IP communication protocols but it also provides benefits to the overall mission. These benefits include the ability to prototype and even test subsystem interactions much earlier in the mission life cycle by allowing subsystems to perform initial interface tests across the Internet. These functional tests can now occur years before they normally would and problems can be identified and corrected sooner. The goal is to actually have space qualified routers that can perform functions such as routing, mobility, and security just like ground routers.

Ground stations need to keep supporting existing missions but can be upgraded to support Internet communication in parallel with legacy missions. Communication from ground stations to users already uses commercial technology and circuits. The ground networks may be implemented on private circuits for security reasons but they can still leverage Internet technologies to provide low-cost, high-bandwidth, scalable communication. The ground station may also be either a passthrough site or a store-and-forward site. The passthrough operation is implemented using standard Internet routing with data flowing through the ground station in real-time and with no local storage. This is how routers pass data across the Internet today. Store-and-forward operation would involve data being stored at the ground station and passed to other nodes on the

ground at a later time. This could be used to stage commands for later upload and to capture data at the station for later dissemination on the ground. Store-and-forward concepts can also be used to capture high rate data from space for later distribution across lower rate ground links.

The data communication requirements of many advanced space missions involve seamless, transparent connectivity between space-based instruments, investigators, ground-based instruments and other spacecraft. The key to an architecture that can satisfy these requirements is the use of applications and protocols that run on top of the Internet Protocol (IP). IP is the technology that drives the public Internet and therefore draws billions of dollars annually in research and development funds. Most private networks also utilize IP as their underlying protocol. IP provides a basic standardized mechanism for end-to-end communication. The following sections break down network communication into smaller pieces and discuss how each area relates to space communication.

2.2. ISO protocol layer model

The ISO OSI model defines a protocol stack with seven layers for any network. Those seven layers and their associated areas of standardization are:

1. physical—bit stream details, signal voltage, cable specifications, modulation techniques,
2. data link—supports transfer of data across the physical link in frames,
3. network—hides data transmission details from upper layers, end-to-end addressing,
4. transport—data stream multiplexing, reliable and unreliable transparent data transfer between end points,
5. session—provides control structure to establish, manage and terminate connections,
6. presentation—performs data services (ASCII-EBCDIC, encryption, security),
7. application—provides services to users of the OSI environment (file transfer, remote login, network management).

Fig. 2 shows the layers with some of the most common protocols at each layer. It also shows the more common Internet version of the layer model with layers 5–7 grouped into a single layer. The diagram also shows some of the Consultative Committee for Space Data Systems (CCSDS) protocols and how they relate to the Internet model.

The important thing to note in the layer diagram is the commonality of IP as the network layer. This allows everything above the network layer to operate independently of the type of link used. Similarly, in the lower layers, different hardware drivers can be substituted without affecting

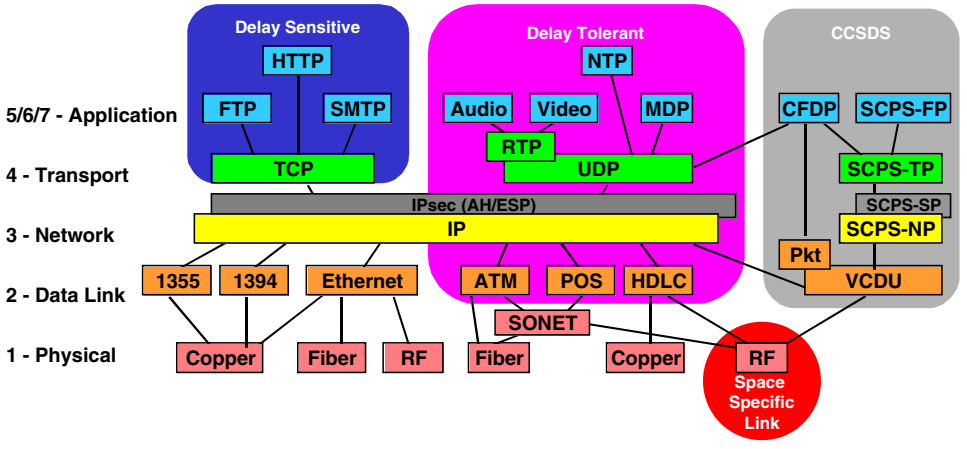

Fig. 2. IP and CCSDS Protocol families.

the specific applications being used at the upper layers.

It also shows how the physical details of the space link are handled at the lowest layer. This includes the basic processes of delivering bits across the RF link using appropriate coding and forward-error-correction mechanisms. This makes the space link perform similarly to other links and allows the use of standard upper layer protocols.

3. Proposed architecture

Recognizing the clear benefits of IP as an end-to-end networking protocol, the OMNI project developed a reference system architecture for the space and ground segments of future IP missions. The goals were to maximize the use of commercial-off-the-shelf (COTS) hardware and protocols while avoiding creating any new "space-specific" solutions. A high-level view of this architecture appears in Fig. 3.

Of course, as spacecraft become more easily accessible network nodes, communication security becomes more of an issue. But as the Internet becomes more critical to everyday business operations, extensive security solutions such as firewalls and Virtual Private Networking [7] (VPN) are being developed which can also be applied to satellite missions.

The key to this whole architecture is that it is built upon the protocol layering concepts of the ISO OSI network reference model. A key component of the diagram is the single network layer protocol (IP) that provides an end-to-end, common denominator that ties together all the other protocols. This feature has allowed the Internet to grow to hundreds of millions of users while still supporting data delivery between all users. The layer isolation has been critical in allowing evolution in both the lower and upper layers with minimal change and reconfiguration of existing users.

Some of the primary protocols identified for space related use are shown in the bottom three layers of Fig. 3. They include standard Ethernet LANs in space and on the ground with High-level Data Link Control (HDLC) [8] framing over the RF link. However, getting IP to the spacecraft cannot be seen as the ultimate goal. We must look beyond just moving IP packets around on a RF link and continuing to operate spacecraft mission systems in the same old manner. The true power of IP to the spacecraft lies in its global addressing and datagram delivery and the use of higher level protocols to change the way that the ground and the spacecraft interact. This allows the use of off-the-shelf solutions from other disciplines, such as industrial networks, and large-scale system and network management systems.

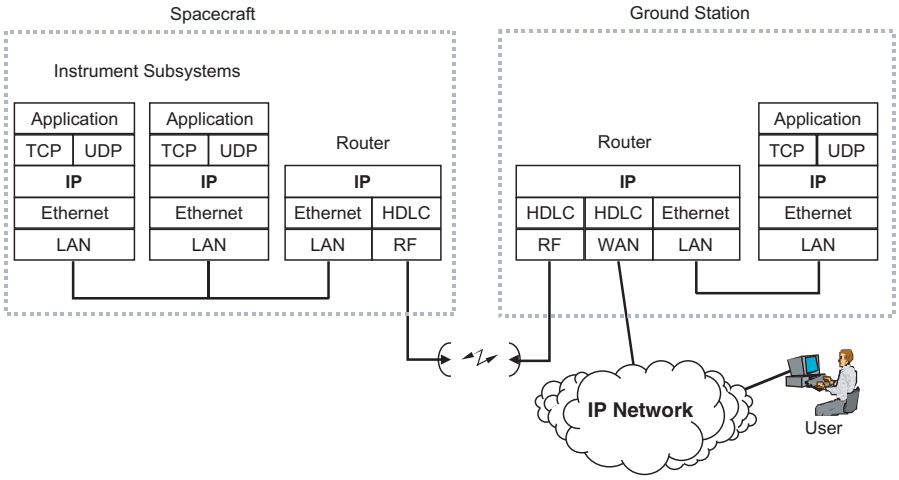

Fig. 3. Protocol stacks for an IP mission.

The key points of this diagram are that the upper layer network services, layer 3 and above, operate on an end-to-end basis and are independent of the various physical media and data link protocols at layers 1 and 2. The end systems do not even know what sort of other data links, in this case an Ethernet LAN, an RF link, and ground LANs and WANs are in the path between the end users. Network components, such as routers or bridges, between the end systems operate in only the lower 2 or 3 layers of the protocol stack and are completely unaware of what upper layer protocols are passing over them. This concept of upper layer independence from the lower layers has been exploited extensively in the Internet. This layered and modular approach is what allows the Internet, and potentially space networks, to make changes and upgrades in one area without any impact on other areas. In the Internet the lower layer communication links are constantly being upgraded to higher rates and different link protocols, but no changes are required by the hundreds of millions of Internet users.

In current space-based communications, the data link and physical layers are represented by the RF link from the ground station to the spacecraft and by framing mechanisms such as time division multiplex (TDM) major/minor frames and CCSDS transfer frames. Current missions do not implement a standard network layer to provide network-wide addressability. Once data leaves the space-to-ground link other mechanisms such as dedicated circuits or address translation gateways are used to direct the data to its next destination. The application layer is implemented with concepts such as CCSDS packets. However, the intervening layers change with the medium that is being used for transferring the data. For the spacelink itself, the transport and network layers are non-existent and the session layer is defined by acquisition of signal (AOS) and loss of signal (LOS) and commands sent by the end-user to start and stop transmissions. In the ground link, the network and transport layers are satisfied with custom NASA data formats and, in many cases eventually delivered over IP protocols anyway.

In contrast, a terrestrial network has a much more orderly progression of layers. The Internet is a strong example of how the layers actually interact. In the transfer of data across the network, the application, presentation, and session layers are defined by protocols such as Hyper Text Transfer Protocol [9], network file system (NFS) [10], Telnet [11], and File Transfer Protocol (FTP) [12], The data formats are preserved throughout the end-to-end transfer of data. The transport layer is either the Transmission Control Protocol (TCP) [13] or the user datagram protocol (UDP) [14] and the network layer is IP. Again, these formats are preserved throughout the network. At the data link and physical layers, the frame headers change as the physical media changes. As data moves from the local area network to a wider area network, the lower levels of the network model change, but the upper models are preserved. This is not the case in the current space mission networks where the change in the network protocol begins at the transport layer.

Extending the ground network to the spacecraft requires some very simple concepts:

- The spacecraft is either a computer or a network of computers with some very specialized peripherals (kind of like a lab computer with measuring devices hanging off of it).
- The ground station antenna (and associated gear) is a RF transmitter/receiver and data router for layers 1, 2 and 3 in the same way that a bridge or router is a media converter between an Ethernet LAN and a serial interface for a wide-area network (WAN).

Fig. 3 indicates a potential stack diagram for an end-to-end communication system using standard network technologies for all ground communication and adding in the necessary modification to fit the space link into the overall networking model. The main purpose for a diagram of this type is to identify the data formats and protocols used on each link and to verify that identical physical and data link protocols are used on the ends of each link. The dotted line at layer 3 indicates an end-to-end network protocol such as IP which provides end-to-end addressing and hides the details of the lower layers from the upper layers. It should be noted that the only space specific part

of this diagram is the RF link between the spacecraft and ground. All other parts use standard Internet technology.

3.1. Onboard spacecraft IP LAN

One of the key features of this architecture is the incorporation of an IP stack in the onboard processor. It may also include the use of peer-to-peer IP networking via an onboard LAN. The use of IP provides end-to-end network addressing between any combination of onboard systems with each other, multiple ground sites, and potentially other spacecraft. An onboard LAN supports distributed processing and "smart" instruments, while IP in an onboard processor supports legacy processor-controlled "dumb" instruments. There is an IP address associated with the processor, each "smart" instrument, and the router, and they are directly reachable from any node on the network. The router takes care of delivering packets to the appropriate LAN address without processor supervision. This is in contrast to the conventional master–slave architecture of a typical IEEE 1553 bus spacecraft, where the processor must be responsible for all bus traffic by managing the bus time-slicing in real-time. Candidate LANs for future spacecraft include Ethernet, CANbus, IEEE-1355 (Spacewire), and IEEE-1394 (Firewire).

3.2. Spacecraft router function

A router is a network device that has a processor and two or more network interfaces and forwards or "routes" IP packets among its interfaces based on their network destination address. At its basic functional level, the router performs simple conversion from one link-level interface to another. For example, the spacecraft router in Fig. 3 could be converting HDLC framing on a serial link into Ethernet framing on a LAN. In small, simple spacecraft, there may be only a small number of "dumb" processor-controlled instruments. In this case, a LAN would not be needed, and the spacecraft would have a single IP address for the processor. The remaining router functions could then be performed in software on the processor. This configuration minimizes costs and spacecraft redesign while still retaining the benefits of end-to-end IP networking.

4. Physical layer issues

Before a spacecraft can transfer any data to the ground or another spacecraft, a communication link must first be established. At the lowest level this consists of activities like tuning transmitters and receivers and pointing antennas. This all assumes that technology for using radio frequencies or optical techniques have been implemented in the space and ground systems that can deliver the necessary bits across the required distance. Future missions are considering scenarios that require gigabit data rates from orbits beyond the moon and at planetary distances. Developing the basic transmission technologies and implementing them within the power, size, and weight restrictions of spacecraft continues to be a major challenge. This paper does not address the lowest level RF modulation and transmission mechanisms but focuses on techniques to be used once bits have been delivered across the link.

A major theme of this paper is that work on developing new transmission technology can and should be done independent of the protocols used over the space communication link. This is part of the layering concept where the interface between the physical bit delivery layer and the data link framing is at the bit level. This section describes some of the techniques used at the physical layer and they include some sort of framing. However, the purpose of that framing is only to operate on a bitstream and improve link quality. There is no addressing or protocol information of any sort at this layer.

4.1. Bit delivery

The basic function of the physical layer is to provide a mechanism to deliver bits across a point-to-point link or between two nodes on a multi-node local area network. Sending bits over a link requires the use of some type of modulation or coding technique to place bits on the physical

media and to extract them on the other end. A simple physical modulation technique is to represent a 0 with a low voltage and a 1 with a high voltage. If the link is a relatively noise-free, a direct connection with a pair of lines with data on one line and a clock signal on the other can be used. The data is recovered by simply sampling the data line at each clock cycle. This type of signaling is used in common serial line protocols such as RS-449/422 and V.35.

However, if the link only has a single line the clock and data must be combined on the link in a form that can be recovered on the other end of the link. This is normally used on media such as Ethernet, optical, and RF links. Bit recovery normally consists of detecting transitions on the line and synchronizing a phase-locked loop to recover a clock signal and then extracting bits from the received signal. But, this can lead to data recovery problems if there is a long string of zeros or ones because the phase-locked loop can drift and random bits may be added or deleted.

4.2. Modulation and coding

There are many bit level modulation and coding techniques available that can provide more reliable data recovery over these serial links with an embedded clock signal. The exact techniques vary widely depending on the physical transmission media being used. Some of the most common media are copper wires, fiber optic cable, and radio frequency (RF) wireless transmissions. Some of the commonly used physical modulation and coding schemes are

- Manchester coding for 10 Mbps Ethernet.
- 4B/5B for 100 Mbps Ethernet and Fiber Data Distributed Interface (FDDI).
- 8B/10B for Gigabit Ethernet and Synchronous Optical Network (SONET).
- Biphase shift keying (BPSK) and quadrature phase-shift keying (QPSK) for RF systems.

The details of these physical modulation techniques are not covered in this paper. However, one point to note is that the same modulation technique must be used on both ends of a physical communication link. This can be seen in protocol stack diagrams by noting that the bottom layer protocol must always match between any two devices on the same link. The main issue is that the modulation and coding technique used is independent of the upper layer framing. This allows the use of any coding technique, including those optimized for space use, with standard data link layer framing and IP protocols.

4.3. Forward-error-correction coding

A common approach to dealing with potential erroneous bit recovery on space links is to include additional bits that the receiver can use to detect and correct damaged bits. This type of coding is referred to a forward-error-correction (FEC) since the error correction information is passed forward with the data. Various FEC coding schemes have been devised over the years. Some of the most common FEC techniques are convolutional coding and Reed–Solomon (R–S) coding.

The major difference between these two coding techniques is that convolutional coding operates on a serial bitstream with no specific byte boundaries while Reed–Solomon coding operates on fixed size blocks of data. A convolutional encoder accepts individual bits, adds additional coding bits based on a predictable algorithm, and passes out the encoded bitstream. A convolutional decoder reverses this process by identifying the original pattern, removing the additional bits, and passing out the original bitstream. The additional bits provide sufficient information so that some errors can be detected and corrected by the decoder.

Reed–Solomon coding does not insert bits into the middle of the data but appends check symbols to a whole block of data. These symbols can later be used to detect and correct errors that may have been introduced in the data. Since RS coding operates on a block of data the receiver must locate the RS synchronization pattern at the beginning of the code block. The CCSDS Reed–Solomon coding specification [15] uses a 4-byte synchronization pattern (`0x1acffc1d`) to delimit the code blocks and a (223, 32) coding scheme. Using a 4-byte pattern and fixed length blocks provides a robust sync detection in more severe bit error environments.

The long sync pattern is less likely to spuriously occur due to bit errors and the fixed length blocks allow the receiver to "flywheel" or assume where a sync pattern should be and continue processing data without dropping lock.

The Intelsat Technical Note TN309.5 specifies a Reed–Solomon code for carriers to use and it has a 4-byte sync pattern (`0x5a0fbe66`) and Reed–Solomon code parameters of (219, 201, 9). It also specifies an interleaving scheme to distribute burst errors over wider areas of data and increase the probability of error correction. A common use of these Intelsat communication links is to provide WAN connectivity between switches and routers transmitting HDLC frames. Another commercial application of Reed–Solomon coding is in Digital Video Broadcasting (DVB) which uses yet another Reed–Solomon coding algorithm. The main point is that many communication applications use forward-error-correction techniques today but it is used to simply provide better link quality and is independent of any data link framing implemented by higher level users.

This is different from many current spacecraft systems where the RS framing is also used as the data link framing. However, this then forces each data link frame to be fixed length to match the RS code block length. The main problem with this is that science and engineering data packets are normally not the same size as the RS frame.

Fitting various length packets into fixed length RS frames means that additional information must be included along with the packets. This information indicates where the first packet starts in a frame and how long each packet is. Since the various packet sizes do not fit evenly into RS frames, packets are also split between frames.

On receipt, if there are too many bit errors in a frame the Reed/Solomon coding will not be able to correct the damaged bits. In this case the frame is discarded along with the part of the packet from the previous and following frames.

4.4. Separation of framing and coding

One of the most important issues in this paper is to note that unlike current space communication systems, commercial network products perform forward-error-correction (FEC) coding, such as Reed–Solomon or convolutional, independently from the data link framing. This is in accordance with the OSI layered model of networking, where framing is carried on at the data link layer and coding is down at the physical layer. The coding simply treats the data link frames as a bit-stream to be protected. This is a key difference between the current data formats used in many space missions and the OMNI architecture.

This separation, as illustrated in Fig. 4, is the standard way Internet connectivity is deployed across commercial satellite links. Commercially available satellite modems support many modulation and coding techniques to improve the bit error rate (BER) of bits passed through communication satellites. However, the inputs and outputs of these modems are simply a clock and data bit-stream. This allows users to connect whatever network equipment they want and use any framing

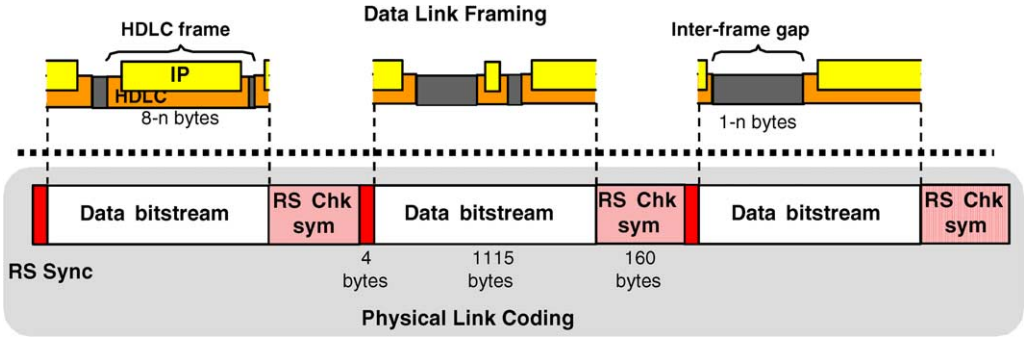

Fig. 4. Separation of HDLC framing and RS coding.

protocol desired. There is no relationship between the users data link framing and any framing that might be used over the RF link. This approach allows future spacecraft to use new and better coding schemes by only changing the FEC processor in their transmitters/receivers and requires no changes in the rest of the installed equipment onboard or in ground systems.

Reed–Solomon coding is also commonly used as a bit level FEC mechanism for many other applications such as cable modems, ADSL, cell phones, direct-broadcast TV, and CD-ROMs. These applications do not use the RS code block for data link framing but simply to provide better data quality to the bitstream being delivered.

Finally, separating the Reed/Solomon code block framing from the data link framing eliminates the current need for fill frames and fill packets. Since the space link uses synchronous clocking, conditions occur where there is no upper layer data to be sent but frames must still be output. Current protocols implement fill packets to be used to fill out frames to meet frame output timing requirements. This added complexity goes away when RS coding is separated from data link framing.

The Reed–Solomon coding simply operates on a bit level and is constantly accepting bits without any relationship to whether the upper layers are sending frames or not. This is the way Reed/Solomon coding is used in all other commercial applications. This is also the way that Reed/Solomon coding has been used on the WIND and POLAR spacecraft for the last 5 years.

5. Link layer issues

The link layer builds on the bit delivery capabilities of the physical layer and provides a mechanism for delimiting a group of bits into an identifiable frame of data. The link layer also adds addressing information, possibly control information, and some type of frame level error detection mechanism, normally a cyclic redundancy check (CRC).

A space mission communication system consists of many different data links to carry data from the science instrument on the spacecraft, down to the ground, and eventually to the scientist. The OMNI project sees great potential for implementing "faster, better, cheaper" satellite communication systems using the link layers that have been very successfully used to build the Internet. Based on the most common COTS technology, this would consist of Ethernet framing on LANs onboard the spacecraft and HDLC framing on the space-to-ground link. On the ground end, COTS network equipment is widely available to support data rates up to 51 Mbps using HDLC framing over clock and data serial interfaces. HDLC framing has also been used at rates up to 100 Mbps in some products. There are no special data rate limits with HDLC framing. The only rate limitation is the speed of the actual hardware implementation.

Some other onboard LAN technologies that are currently being worked on are the IEEE-1355 (Space Wire) and IEEE-1394 (Fire Wire). The mapping of the Internet Protocol into these media is not as well defined as Ethernet but work is underway in the IETF to define IP over IEEE-1394.

5.1. Onboard Ethernet framing

Recent developments in industrial automation have shown great potential for using standard Ethernet technology for data communication in real-time environments like spacecraft. Major efforts are underway to use Ethernet in industrial environments that have always had requirements for real-time, deterministic, reliable, and secure operations. Many companies have come together to form the following groups:

- Online Industrial Ethernet Book—http://ethernet.industrial-networking.com/
- Industrial Automation Open Networking Alliance—http://www.iaona.com/
- GE Cisco Industrial Networks—http://www.gecisco.com

The OMNI project sees great potential for building on the current industrial Ethernet work to develop a cost effective Ethernet solution for use on spacecraft.

This is based on the fact that Ethernet has taken over the majority of the data communication world with a huge number of Ethernet interfaces being deployed. Ethernet also supports a wide range of data rates of 10, 100, 1000, and 10,000 Mbps with a 40,000 Mbps version currently under definition. There is also a tremendous amount of research and development going into standard upper layer protocols for use over Ethernet, ruggidized connectors, and new strategies for using Ethernet in process control environments.

Studies in the late 1990s showed that Ethernet response times can consistently be maintained under 2 ms for a lightly loaded Ethernet network and under 30 ms for a heavily loaded network. The key to successfully using Ethernet is the proper design of the network topology and traffic patterns and using devices such as switches to separate traffic and reduce collisions.

Work is also underway in the industrial Ethernet community to define upper level programming languages and an application programming interface (API) to standardize the software used on real-time Ethernet LANs. The API will address issues related to timing, low-level device control, and real-time response. This work should also be useful for future spacecraft designers.

5.2. RF link HDLC framing

Based on its near-universal use on the terrestrial Internet, the OMNI project chose HDLC framing for the link-level protocol on space-to-ground links. This allows simple, straightforward interfacing with existing commercial routers in the ground station. HDLC has been used in communication equipment for over 30 years and provides basic framing for many serial line protocols such as IBM's synchronous data link control (SDLC), Frame Relay, X.25, and ADCCP.

As indicated in Fig. 5, at the physical link 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. During any idle time, successive flag bytes are output until the next frame begins. Flag bytes consist of a zero bit, 6 one bits, and a zero bit (01111110). 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. On receipt, these extra zero bits are automatically removed from the data by the hardware.

Using data link framing that relies on only a single byte flag pattern to delimit frames is a concern for noisy environments like space. However, using HDLC on top of forward error correcting codes addresses that problem. Before the HDLC sync is even an issue the lower layer coding must be successfully processed. Convolutional and Reed–Solomon coding use much stronger synchronization mechanisms and once they have been processed, any data link framing will be more reliable. This is especially true when using

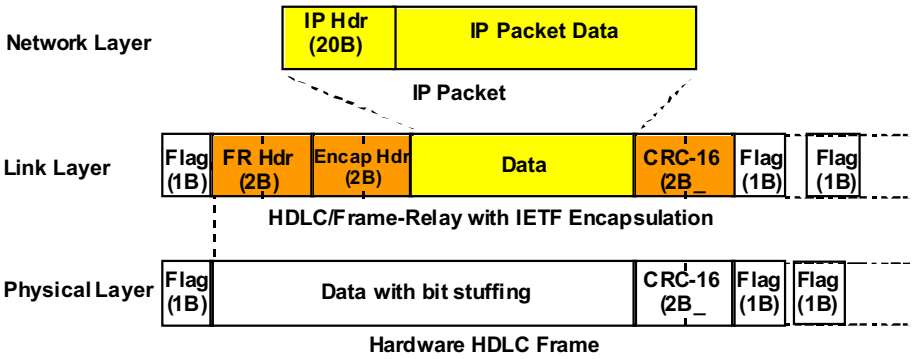

Fig. 5. HDLC/frame relay/IP formats.

Reed–Solomon coding since the result of the R–S processing is normally a perfect bit stream. In present space data processing systems, if the R–S block is so badly damaged that the coding cannot correct it, the block is discarded. Using HDLC on top of Reed–Solomon actually allows the damaged bits to be passed on to the HDLC frame processing to see if it can locate some good frames within the damaged R–S block. HDLC can use its 16-bit CRC to determine if it has extracted a good frame from the R–S code block. This could potentially result in successfully extracting more data from uncorrectable Reed–Solomon blocks than is possible today.

While the primary purpose of "bit stuffing" is to ensure the uniqueness of the flag byte, it also has an additional benefit for space use over RF links. 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, the use of standard non-return-to-zero (NRZI) coding for the HDLC output 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 "space specific" requirements can 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. None of the protocols layers need to worry about generating "fill packets" or "fill frames".

The OMNI project considered various commercially available encapsulation mechanisms for use over HDLC. There were two major criteria for selecting the encapsulation method to use:

- the encapsulation could not require full-duplex links since full-duplex links might not be available during a spacecraft emergency,
- the encapsulation must be interoperable between many vendors routers since no group can ensure that all routers at all ground stations will come from the same vendor.

The first criteria ruled out protocols like Serial Line IP (SLIP) and Point-to-Point Protocol (PPP) [16] because they need full-duplex links for parameter negotiation at startup. The second criteria ruled out protocols such as Cisco's default HDLC encapsulation which uses a Cisco specific HDLC header.

This led to the choice of the IETF encapsulation for multiprotocol over frame-relay/HDLC specified in RFC 2427 [17]. In the OMNI tests with UoSAT-12 the actual header format consisted of simply inserting 4 bytes of fixed information at the start of each HDLC frame. The first 2 bytes are a standard Frame Relay header with a few status bits and a virtual channel number or Data Link Connection Identifier (DLCI). Also, since this is a standard Frame-Relay header, a spacecraft could actually use the DLCI to provide additional channelization and routing in addition to the IP capabilities. This could be used along with standard Frame-Relay equipment at the ground station. The next 2 bytes in the header simply indicate that the contents of this frame are an IP packet. There are also standard IETF definitions that allow the transport of other protocols in the data area of the frame.

This data link framing provides capabilities identical to those used by current spacecraft. An application level science or telemetry packet inside of a User Datagram Protocol (UDP) packet with an IP header and HDLC is delivered through space exactly like current data. The main difference is that by using IP and HDLC headers the data leaving the spacecraft is in a format that can be directly ingested by COTS Internet equipment on the ground.

5.3. High-rate RF link framing

Supporting data rates over 51 Mbps using commercial routers requires using a framing technique other than just HDLC. Commercial routers have interfaces that support data rates up to 51 Mbps

using HDLC framing over High-Speed Serial Interfaces (HSSI) but shift to Synchronous Optical Network (SONET) interfaces for data rates of 155 Mbps, 622 Mbps and 2.4 Gbps. These interfaces have traditionally used Asynchronous Transfer Mode (ATM) cells to frame IP packets over SONET.

One objection to using ATM for science satellite communication is the 10% overhead imposed by the ATM cell format. ATM cells contain 48 bytes of data with an additional 5 bytes of cell header. IP packets must be broken into 48 byte pieces with some additional information added to help the receiver reassemble the packet. This process of splitting the IP packet adds complexity and results in additional error cases where the loss of a single ATM cell results in the loss of the entire IP packet. In an environment like ground fiber links with large amounts of bandwidth these issues have traditionally been accepted. However, as the Internet grows and users want more and more bandwidth, alternatives to ATM cells have arisen.

One of the more popular alternatives to ATM cells for highspeed IP support is to bypass the overhead of ATM and put IP packets into SONET. This format is called Packet over SONET (POS) [18]. There is still some framing needed but the framing has gone back to the traditional mode of using HDLC framing to put one IP packet in one HDLC frame and carry that over SONET. A PPP header is also added and the end result is very similar to the multi-protocol over Frame Relay format described above.

However, SONET implementations have not been widely implemented over RF links. Digital Video Broadcast (DVB) currently supports high-rate data delivery over satellite links at rates up to 200 Mbps. Another format for investigation is the Generic Framing Protocol (GFP). More work is needed to identify the best framing mechanism for higher rate data streams.

However, spacecraft with this type of high-rate downlink normally have multiple transmitters operating at both low and high rates. They also would not normally be attempting any high-rate downlink if the spacecraft was in trouble. A choice of link protocols for data rates above 51 Mbps needs further work but can be addressed independently while using HDLC for rates up to 51 Mbps.

5.4. Framing overhead

A major concern for satellite system engineers is both the processing overhead and byte overhead associated with protocols. This is not a major issue for onboard LAN protocols where bandwidth is not as severely limited. Overhead is an issue on the space-to-ground link where bandwidth is often limited due to standard RF link budgets affected by power, error rate, signal quality, and distance.

The overhead of HDLC is very minimal with only the following fields:

- 1-byte flag or sync byte,
- 4-byte Frame Relay and IP encapsulation header,
- 2-byte CRC for error detection.

This framing overhead is as small as other space framing formats used today.

Another aspect of the HDLC framing is its bit-stuffing. This ensures that on the transmission media there are never more than 5 one bits in a row. Breaking up strings of ones is necessary to avoid patterns that would look like a flag byte and signal the beginning or end of a frame. Inserting ones into the data stream results in added bit overhead for HDLC. The extreme case would be an overhead of 20%, which would result from a frame containing all one bits and a zero bit would be inserted after every fifth bit. However this scenario is very unlikely since sending frames of all ones would be a waste of bandwidth anyway. That sort of data can be easily compressed with major reductions in data volume. Another option is to apply data randomization before passing the data down to the HDLC layer. This further reduces the likelihood of long strings of ones. Some examination of data files from the WIND, POLAR, and SOHO spacecraft indicate a realistic HDLC bit-stuffing overhead is in the 1–3% range.

6. Network layer issues

The network layer is the key to the global connectivity provided by any network and especially the Internet. Frames at the link layer normally contain source and destination addresses but those addresses are only valid at the link layer. Those addresses are only used to deliver frames to the proper device on a single physical link. Once a device receives a frame, the data link headers are discarded and only the network layer information remains. The network addresses are globally unique and remain with the upper layer data to provide the information needed for a network of routing devices to forward data to its final destination.

6.1. Internet Protocol

Fig. 6 shows the basic format of the IP header. The primary fields to note are the 32-bit source and destination addresses. These are used to deliver the datagram to its destination. The source address also acts like the return address on a letter and tells the recipient where this data came from and how to communicate back to the source. This feature of source and destination addresses on each datagram is critical to supporting future cooperative science and constellation space missions. Supporting communication among large numbers of spacecraft requires either a central communication hub that knows the addressing details of each spacecraft or an automated mechanism like Internet addressing and routing.

Many of the other fields (e.g., total length, identification, do not fragment, more fragments, and fragment offset) are involved in fragmenting and reassembling large IP datagrams for transport over data links that cannot carry the whole datagram in a single frame. The "time to live" (TTL) field sounds like a field that might be affected by long delay space links like L1/L2 or Mars and beyond. However, this field is not really a time but a count that decrements each time the datagram passes through a router. Its function is to protect against misconfigured routers with a routing loop where a packet could be passed around the network forever. The TTL causes the packet to age and be discarded after a short time. The "type of service" field provides a mechanism for prioritizing datagrams if needed. Finally, the "protocol" field indicates the next level protocol (e.g., UDP, TCP) contained in the data portion of this datagram.

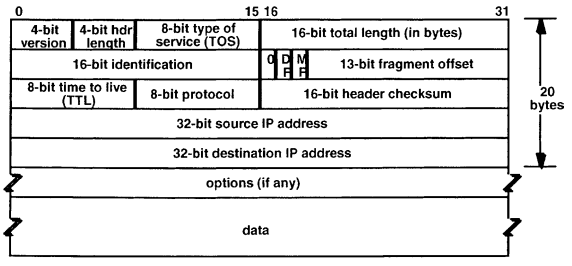

Fig. 6. Internet Protocol header format.

The IP header on each Internet datagram consists of these 20 bytes in a fixed format. There are also standard options that can be added to carry additional control or status information. Many of the options are used to either specify a particular path the datagram should take from router to router, or to record the route the datagram actually took.

Another option is the timestamp option which requests each node that handles the datagram to add their local timestamp to the header. This option was used in tests with the UoSAT-12 spacecraft as a simple way to read the spacecraft clock. Internet Control Message Protocol (ICMP) or PING packets were send from a ground router with the timestamp option set. The ground router entered its local time, the spacecraft entered its time when it handled the PING packet, and then the ground router added a final timestamp when it received the PING response.

6.2. Datagram routing

The basic construct the Internet is built on is a capability for network devices to simply forward datagrams toward a destination address. There is no guaranteed delivery in the network layer. Its basic function is to provide addressing and associated datagram delivery. However, datagram delivery becomes a complex problem as a network grows to the size of the Internet. When a network

is small, the lookup tables that tell each router now to forward datagrams toward their destination are small, simple and relatively static. The maintenance of these routing tables can be performed in a manual or simple automated fashion. This is similar to the manual management of current spacecraft data delivery systems.

When a network reaches the size and continual growth of the Internet, the updating of routing tables must be implemented automatically between routers. There has been a tremendous amount of work done by groups such as the Internet Engineering Task Force (IETF) to develop and deploy robust and automated routing technologies. Many routing protocols such as Routing Information Protocol (RIP) [19], Open Shortest Path First (OSPF) [20], and Border Gateway Protocol (BGP) [21] have been developed to support the Internet. These protocols use the relatively stationary topology of the Internet to determine paths to subnets and inform each router where to forward packets to reach each subnet. Routers also have default routes that are used as a path of last resort when there is no better route information available.

Routing tables are reasonably static but change whenever a link fails and the routers adjust to define new paths for datagram forwarding. The Internet addressing scheme also assumes that a device with an IP address remains attached to its subnet. However, when we start using IP addresses on spacecraft or other mobile devices some new routing issues arise.

6.3. Mobile IP

In today's spacecraft communication, control centers normally send commands to the ground station the spacecraft is passing over and the command is uplinked to the spacecraft. The major issue is that the control center must know where to send the commands and address them accordingly. However, as large constellations of spacecraft are deployed, advance planning and scheduling of contacts becomes more complex and expensive and an automated solution for delivering commands to spacecraft is desirable.

As indicated in Fig. 7, when a spacecraft has an Internet address (e.g., 100.10.10.18), that address will be part of a ground based subnet (e.g., 100.10.10.x). Any IP datagrams addressed to the spacecraft address from anywhere on the Internet will be routed using standard Internet routing and will be delivered to the associated ground subnet. However, since the spacecraft is moving from station to station, the home agent router will not know where to forward the packets. This will not get the commands to the spacecraft. However, this is exactly the same problem encountered by many wireless devices such as mobile laptop computers. The IETF has developed standards called Mobile IP (RFC 2002) to deal with this problem.

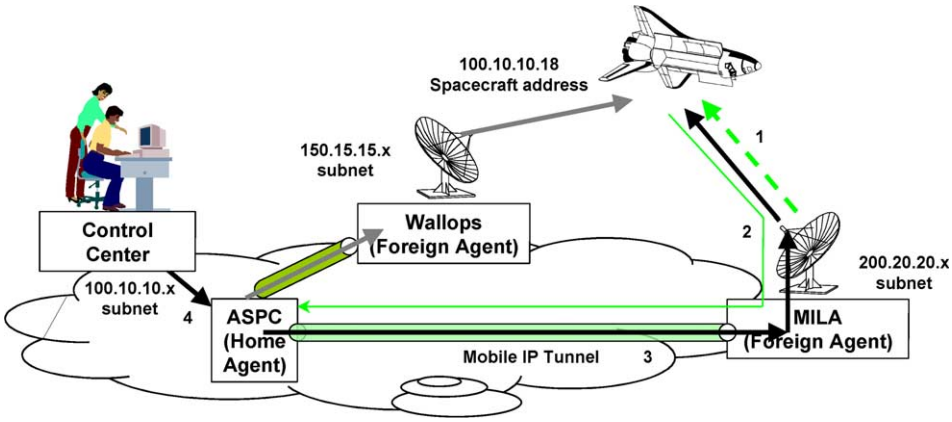

Fig. 7. Mobile IP for spacecraft.

When the spacecraft approaches a ground station (e.g., MILA) it will hear a Mobile IP advertisement-1, from the ground station and respond with a registration request-2 back to its home agent. A new routing tunnel-3 will be established to the current ground station. Then when the control center sends a datagram-4 to the spacecraft address, the packet goes to the home router where the home agent notices that there is a tunnel to the spacecraft via a foreign router. The packet is then sent through the tunnel to the foreign agent which passes it out its serial interface and up to the spacecraft.

This sort of Mobile IP scenario is mainly only an issue for sending data to the spacecraft. When any packets are sent from the spacecraft to any ground station, the ground station simply uses the destination address to forward the packets using standard Internet routing rules. One possible exception is if the foreign ground station has additional routing rules, for security reasons, which prevent it from forwarding packets whose source address is not within the foreign subnet. Then the tunneling features of Mobile IP would be needed to encapsulate the spacecraft packets for delivery to their home subnet.

There are other possible options for solving these mobility issues but Mobile IP currently provides the best solution for automation and scalability.

One other option is to simply give the spacecraft an IP address in the subnet of each of its ground stations. However, this requires both the spacecraft and ground controllers to keep track of which station is currently in use and the corresponding addresses. This solution also gets more complex as more ground stations are added and does not support the use of additional ground stations without spacecraft reconfiguration.

Another option is to use the Dynamic Host Configuration Protocol (DHCP) to let the spacecraft learn an IP address for use during that contact. This allows the spacecraft to downlink data but still has problems with getting commands to the spacecraft. It requires some mechanism for telling the control center the current IP address of the spacecraft.

These cases have only addressed a spacecraft or mobile host with a single IP address. If the mobile device or spacecraft has a LAN with multiple IP addresses then the problem gets more complex. One solution is for each node on the spacecraft with an IP address to perform Mobile IP registration and set up tunnels for each. However, this does not scale well and causes additional traffic for all of the registrations and additional software for each node. The solution currently being worked on in the IETF is called Mobile Routing. It involves a router that performs all of the Mobile IP operations and none of the nodes on the LAN even realize they are mobile. They simply operate just like they do on a fixed LAN. The research and development in this area is being driven by concepts in which all future automobiles will have onboard LANs with Internet addresses and full mobile Internet connectivity. The large size of the automobile market, a potential market for mobile routers, is huge and the commercial research and development investments are substantial.

6.4. Data prioritization

Current spacecraft protocols do not really provide any special support for indicating different priorities for types of data. The virtual channels in CCSDS protocols are sometimes used for this but they only provide a few levels of priority and their meaning is different for each mission.

There are many options available in Internet Protocols for supporting a wide range of priority mechanisms. These options include the following:

- DLCIs in the HDLC/Frame Relay header that can be used similar to the CCSDS virtual channels.
- Type of Service bits in the IP header that have traditionally been ignored but are now beginning to see use to support prioritization of Internet services.
- Priority queuing in routers allows sorting individual packets into multiple priority queues based on fields such at the transport protocol (e.g., UDP, TCP) and transport protocol port numbers (e.g., 1-65535).

These options can be used to prioritize urgent information such as gamma ray burst notifications

at the highest priority. Other timely data such as housekeeping data could be at a lower priority telemetry and playback data at even lower priorities. There are also other protocols being developed for Internet use to deal with specific prioritization and quality of service issues. This work is being driven by applications such as voice over IP and streaming video.

6.5. Network protocol overhead

Today's satellite protocols focus on using minimal overhead formats for delivering data from spacecraft to the ground. The HDLC framing described earlier provides minimal overhead at the link layer. The major overhead from using Internet Protocols comes in the Network and Transport layers.

The IP header consists of a fixed 20-byte header with optional fields up to a maximum of 64 bytes. The majority of packets on the Internet use the base 20-byte header. Packets will also have an additional transport header of 8 bytes for UDP packets and 20 bytes for TCP packets. This results in the following overheads for some selected packet sizes shown in Table 1.

The table indicates high overhead for small packets of user data but that overhead drops to much more reasonable values for larger packets. This would seem to indicate some work needs to be done to try to reduce this overhead if these protocols are to be used for spacecraft. Once again the Internet, and its continually growing range of applications, has already encountered this problem and is working on solutions.

A major driving force for reducing the overhead of Internet Protocols is the rapid growth in voice over IP deployment. Voice over IP sends lots of small, digitized voice samples over IP using the UDP transport layer. The 28 bytes of overhead represent a significant portion of the voice packet. There is significant interest in reducing this overhead to allow voice over IP to grow to support millions of users without having most of the bandwidth used for the protocol header.

Some new header compression algorithms have already been developed in the IETF and are available in some vendors' products. The current standards are RFC 2507 [22] and 2508 [23]. They utilize the fact that most of the fields in the headers for a particular session (e.g., source/destination address and port numbers) remain the same for each packet. These algorithms send some packets with full headers and most of the packets with the UDP/IP or TCP/IP header compressed to contain only the information that has changed from the last packet. This results in these headers shrinking to 6–7 bytes. This results in Internet packets that are equal to or in some cases less than the overhead of current space protocols. There is currently an IETF Working Group called Robust Header Compression (ROHC) (http://www.ietf.org/html.charters/rohc-charter.html) that is working on additional compression options for even smaller headers.

Table 1
Network/Transport Protocol overhead

	User data sizes (bytes)			
	100	500	1000	1400
IP (20)	16.6%	3.8%	1.9%	1.4%
UDP/IP (28)	21.8%	5.3%	2.7%	1.9%
TCP/IP (40)	28.5%	7.4%	3.8%	2.7%

7. Transport layer

The transport layer has the major function of providing stream (or packet) multiplexing of multiple channels into a single link. This logical multiplexing is distinct from any physical resource multiplexing that occurs lower down in the link layer. All Internet transport layer protocols provide this capability, commonly referred to as "ports" or "sockets".

This multiplexing capability is one of the most important aspects of Internet transport layer protocols. It provides the ability to transparently mix thousands of unrelated asynchronous data streams on a single physical link, without the applications that generate the data needing to be aware of each other or the layers below. This "virtual channel" capability even provides for intermixing multiple transport protocols on the same

physical link. Each transport protocol has its own separate set of 65,535 ports.

Prioritization of these "virtual channels" is handled down in the network layer (layer 3) by assigning different queuing priorities to unique port-protocol combinations. This is a standard feature found in most IP routers in use today, and was successfully used by the OMNI project to prioritize mixed streams of data.

These capabilities match well with the telemetry requirements of modern spacecraft, which often have hundreds of "application IDs", representing many separate asynchronous data streams running at different priorities.

Beyond multiplexing, Internet transport protocols have a wide range of different capabilities and limitations. There are two main transport protocols currently in wide use on the Internet: Transport Control Protocol [13] (TCP) and User Datagram Protocol [14] (UDP). Although TCP is the most well known of these protocols, it is important to make the distinction that not all IP is TCP/IP. In addition, a third transport protocol, Real-time Transport Protocol [24] (RTP), is in common use but is generally implemented "on top of" UDP instead of as a separate protocol with its own protocol id. Each protocol has its own strengths and weaknesses.

Selection of a transport protocol for a particular type of spacecraft or instrument data is mission specific, and would be driven by the mission requirements and system engineering tradeoffs. There is no single "one size fits all" answer.

7.1. UDP

UDP is a connectionless transport protocol designed to operate over IP. Its primary functions are error detection and multiplexing. UDP does not guarantee the delivery or order of packets, but guarantees that if a packet is ever delivered with errors, such errors will be detected. Because the UDP format is simple, it has a low overhead. See Fig. 8. It is also fast compared to TCP since there is no connection establishment phase.

UDP provides "atomic packet delivery". This means that the application will never see a partial or fragmented packet (regardless of any fragmentation and reassembly performed by the lower layers). Delivery of a packet to the application layer is all-or-nothing.

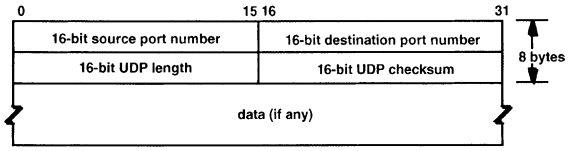

Fig. 8. UDP header layout.

These characteristics make UDP the protocol to use when the timeliness of the data is more important than getting every packet. Examples of this include spacecraft engineering data, health and safety telemetry, and blind commanding.

UDP is a "send-and-forget" protocol. Packets are addressed to their network endpoint and sent on their way without any connection phase or handshaking. This has both advantages and disadvantages. On the plus side, it means that the protocol will work with highly asymmetric or unidirectional links, is completely delay-insensitive, and supports multicast. These characteristics make it well suited for deep-space missions, such as Mars. On the minus side, UDP does not provide flow control or reliable transport. If these features are required with UDP, they must be built on top of it at the application layer, as in today's spacecraft protocols. The following section on the Application Layer discusses several examples of UDP-based applications that take this approach.

7.2. RTP

RTP is used to carry data that has real-time properties. It provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. Those services include payload type identification, sequence numbering, timestamping, and delivery monitoring. Applications typically run RTP on top of UDP to make use of its multiplexing and checksum services, but both protocols contribute parts of the transport protocol functionality. See Fig. 9 for a diagram of the additional 12 bytes of header that RTP adds onto UDP. RTP (and UDP) supports data transfer to multiple destinations using multi-

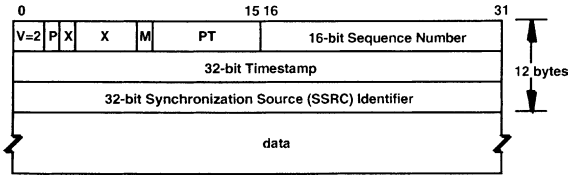

Fig. 9. RTP header layout.

cast distribution if provided by the underlying IP network.

RTP is most well known for streaming audio and video on the Web, in products such as RealVideo and QuickTime, and is used for Voice-over-IP (VoIP). Although originally designed for supporting audio and video over packetnetworks, RTP is also useful for transporting any isochronous data where the timing of the data is important. RTP provides hooks for adding in reliability and flow control if these features are required.

7.3. TCP

TCP is a connection-oriented transport protocol designed to work in conjunction with IP. TCP provides the application layer with the ability to *reliably* transmit a byte stream to a destination, and allows for multiplexing multiple TCP connections on a single host. It provides flow control, and has "out-of-band" handling for priority messages. A diagram of the TCP header is shown in Fig. 10.

Being connection oriented, TCP requires a connection setup phase, followed by a data transmission phase. A connection is terminated when it is no longer in use.

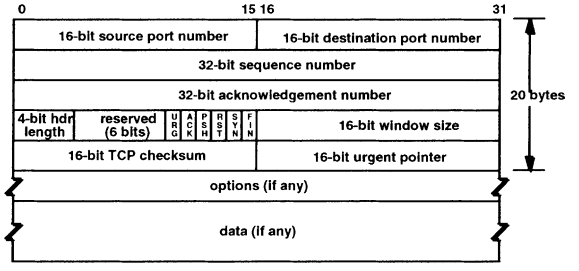

Fig. 10. TCP header layout.

The reliability and flow control of TCP requires that status information be sent with each packet, and acknowledgement be received back from the recipient. This allows TCP to recover from data that is damaged, lost, duplicated, or sent out of order.

These characteristics make TCP a protocol to use when the overriding requirement is for the error-free transfer of data. Examples of this include downloading instrument science data, and uploading spacecraft or instrument command loads. Many off-the-shelf applications are built on top of TCP and can perform this function. In many cases, reliable transfer of instrument data files directly to the scientist can *completely replace* traditional level-0 processing.

Along with TCP's capabilities come some limitations.

Because of the handshaking and flow control, TCP requires a bi-directional link. This link can exhibit only a moderate amount of asymmetry (~50:1) before throughput is affected.

Because TCP is a windowed, buffered protocol, it is sensitive to the Round Trip Time (RTT) delay. With increasing RTT, larger window buffers are required in order to maintain throughput. But larger buffers exact a larger penalty when a packet is lost and has to be retransmitted. In practical terms, what this means is that TCP will perform fine out to about lunar distance. In particular, TCP has been successfully used at geosyncronous distance at over 400 MBit/s [25].

TCP currently has no mechanism for distinguishing loss due to congestion from loss due to noise. This means that increasing noise will reduce throughput as retransmission timeouts increase. The OMNI project has performed experiments which have shown that single-session TCP bandwidth utilization falls to under 60% at an error rate of 10^{-5}; however, through the use of forward-error-correction codes, (FEC), most space missions operate at an error rate of 10^{-7} or better. At these rates, TCP's bandwidth utilization approaches its theoretical maximum of ~92%. In addition, current standards activities, such as Explicit Congestion Notification [26] (ECN), Selective Acknowledgement [27] (SACK), and TCP/Peach

[28], are underway to make TCP more ''satellite friendly''.

8. Application layer

While transport protocols enable the exchange of data, more functionality is required to perform useful work. This is the domain of the application layer.

There are many standard application layer protocols, such as HTTP, SMTP, FTP and Telnet. Most of these are defined in RFCs, are widely implemented, and interoperate universally. Many of them are capable of handling a large variety of space mission requirements. Self-defined protocols are also possible. Designing your own protocol for your application has the advantage of being flexible, lightweight, and efficient, but has the disadvantage of being non-interoperable with other applications. Each mission needs to make that decision based on its own requirements and the cost/benefit tradeoffs.

8.1. UDP-based applications

UDP applications can be divided up into several categories that are relevant to spacecraft operations.

8.1.1. Simple data delivery

In general, a simple custom application would be required to wrap application-specific telemetry or command packets in a self-defined protocol. This is functionally equivalent to current space missions using CCSDS framing and packet protocols. It may even make sense to use the CCSDS packet (not frame) formats inside of UDP, as the CCSDS packet structure is already defined.

8.1.2. Reliable file transfer with UDP

A number of standard applications/protocols are available to perform reliable file transfer via UDP. These include Pacsat Broadcast Protocol [29] (PBP), Multicast File Transfer Protocol [30] (MFTP), CCSDS File Delivery Protocol [31] (CFDP), Network File System [32] (NFS), and Trivial File Transfer Protocol [33] (TFTP). PBP, MFTP and CFDP are of particular interest for deep space missions because they can operate over a mostly unidirectional link. They send out all the file's packets non-stop without waiting for any handshake. Sometime later, (a minute, an hour, a day) a brief return link is required to transmit a list of NAKs for any packets that were lost. These packets then get resent on the next contact. This makes these protocols delay insensitive, just the thing needed for missions at L1 or Mars and beyond.

8.1.3. Time synchronization

The Network Time Protocol [34] (NTP) is a UDP-based protocol and application that is used to synchronize the time of a computer client or server to another server or reference time source. Typical NTP configurations utilize multiple redundant servers and diverse network paths, in order to achieve high accuracy and reliability. Some configurations include cryptographic authentication to prevent accidental or malicious protocol attacks. A space mission using NTP would typically place its primary timeserver right at the groundstation to minimize delay variations and maximize security.

8.2. TCP-based applications

TCP applications can be divided up into several categories that are relevant to spacecraft operations.

8.2.1. Reliable simple data delivery

As with UDP, a simple custom application would be required to wrap application specific telemetry or command packets in a self-defined protocol. The difference here is that TCP takes care of automatically performing any retransmissions required to guarantee delivery of every data byte. In the commanding case, this is similar to current missions using the CCSDS COP-1 protocol. In the telemetry case, there is no current system implemented to perform automatic retransmissions and reliably deliver every data byte. Current missions either tolerate any lost data or manually retransmit the entire data set a second time in an attempt to fill in any losses.

8.2.2. Reliable file transfer with TCP

There are two main application-level TCP-based File Transfer Protocols that are widely available and instantly familiar to anyone who uses the Internet: File Transfer Protocol [35] (FTP) and Hypertext Transfer Protocol [36] (HTTP). FTP is the older of the two, and tends to be more efficient in its use of persistent connections, but is more complex to implement. HTTP, on the other hand, is extremely simple to implement in a small memory footprint, but sets up and tears down a connection for every transfer.

Hundreds of implementations of each of these protocols are available both commercially and as freeware applications. These are a very good match for the needs of science missions, which often have large quantities of prerecorded data that must get shipped to the scientist with maximum fidelity.

8.2.3. E-mail

Simple Mail Transfer Protocol [37] (SMTP), as defined in STD-10/RFC-821, specifies the protocol used to send electronic mail (e-mail) between TCP/IP hosts. E-mail is probably the most widely used TCP/IP application. The basic Internet mail protocol provides mail and message exchange between TCP/IP hosts. Facilities have been added for the transmission of binary data which cannot be represented as 7-bit ASCII text.

SMTP is based on end-to-end delivery; an SMTP client will contact the destination host's SMTP server directly to deliver the mail. It will keep the mail item being transmitted until it has been successfully copied to the recipient's SMTP server. SMTP servers can also be set up as "mail gateways" to implement a "store and forward" delivery system. In either case, the mail is always addressed to the end user.

In space-based applications, SMTP can provide the scientists and spacecraft operators the capability of sending and receiving commands and data when they are not in contact with the spacecraft, and having those files automatically queued and delivered without further human intervention. This "batch" or "bundled" mode of data transfer very closely matches the requirements of many space missions. It can be easily and cost-effectively accomplished with commercial off-the-shelf applications without inventing any new "space-specific" protocols.

8.3. Upper layer protocols and applications

End-to-end IP network connectivity enables new ways to develop, test, and operate future remote systems. This allows the use of a huge number of Internet Protocols and applications that perform remote system monitoring, management, and data handling. However, not all Internet Protocols and applications are well suited to all range and space communication applications.

A basic rule of thumb is that applications using the UDP [14] protocol will work well. Some common UDP-based protocols and usage that have been demonstrated in space are UDP blind commanding, UDP telemetry packets, Network Time Protocol (NTP) [36] and Multicast Dissemination Protocol (MDP) [38]. The UDP transport protocol provides a basic datagram delivery service identical to current frame delivery mechanisms such as TDM and CCSDS. The big difference is the global network addressing capabilities of IP. Also, most missions will be required to use UDP to support communication requirements during conditions such as one-way links, high link errors, high link bandwidth asymmetry, and long propagation delays. Once missions have implemented all their critical functions using UDP, consideration can be given to using TCP protocols in some environments where it's performance is not as seriously impacted by delays, errors, and high rates.

Applications that use the TCP [13] transport protocol need to be examined more carefully for use in range and space applications. TCP provides a mechanism for reliably delivering a stream of bytes but, to provide reliable delivery, it uses a two-way exchange of data and acknowledgements. This feedback loop poses problems when it encounters the following type of communication links:

- one-way links (e.g., bind commanding or TDRSS return-only links),
- high link bandwidth asymmetry (TCP cannot go much beyond 50:1 asymmetry),

- high link error rate (TCP will keep trying until it gets all data through, may not be possible),
- long propagation delays (TCP feedback loop limits throughput with long delays).

Many UDP and TCP based applications are already available for range and space use. Table 4 lists some of the more basic areas that need more investigation for range and space use. These protocols and applications can be used for just onboard communication or all the way from instruments to end users. File transfers can occur from end-to-end in real-time if conditions permit or in a store-and-forward method from one storage system to the next.

9. Space vs terrestrial issues

There are a number of apparent issues for space-based usage of Internet Protocols. These are often misunderstood or misrepresented. A recent quote in Space News stated: "*The environment for the Internet is basically no delays, no errors, continuous connectivity, and pretty symmetric data transfer. If you look at the space environment, it is almost completely reversed. There are high delays and high error rates. The links are not continuous or symmetric.*" If this description of the Internet were true, we would all have continuous 100 MBit/s connectivity to our PCs and cell phones. Instead, we have 56 KBit/s dialup modems, micropower cell phones that run 2400 baud if they can make a connection, and network delays that sometimes run up into the seconds. So, even though on the surface it would appear that "space is special" and has unique problems, upon careful examination, each of these problems can either be found to be a non-problem, or to have a terrestrial parallel that has been solved in the commercial world.

9.1. Long delay

Often it is stated that space missions *must* be carried out with "Round trip delays much greater than ground systems" [39], and that "...long propagation times cause terrestrial protocols to operate sluggishly or fail outright" [39]. For low earth orbit (LEO) missions, which represent the *large majority* of space missions, this is simply not true. A LEO spacecraft is only 200–400 miles away when it passes overhead. Since RF signals travels at the speed of light, this translates into only a 4 ms round trip time! Even at the horizon, which for a spacecraft in a 400 mile high orbit is approximately 3000 miles away, this is about a 32 ms round trip time. Compare this with typical Internet ping times from Baltimore to Los Angeles of 100 ms and the LEO spacecraft should actually run TCP/IP better than coast-to-coast terrestrial links. Even out to geosynchronous orbit, the one way delay time is only 240 ms. Experiments have been performed at the NASA Glenn Research Center [25] using the ACTS satellite, which have operated TCP at over 400 MBit/s at this distance. These experiments used ACTS as a "bent pipe", so a round trip required two hops to geosynchronous distance, or around 480 ms. TCP is limited by its bandwidth-delay product, requiring a transmission window buffer of equal or greater size. This means that low-bandwidth/high-delay TCP connections are similar to high-bandwidth/low-delay ones. Laboratory experiments have suggested that lunar distance, with its 2.5 second round trip time, would require some care in setting up the connection, and represents the practical limit for TCP-based applications. Beyond this distance, deep space missions, such as Mars, should look to using one of the delay-insensitive UDP-based protocols, such as MFTP, PBP, or CFDP.

9.2. Noise

Frequently, it is pointed out that most packet losses on the Internet are due to congestion, whereas most losses on a space-to-ground link are due to noise. TCP has no mechanism for distinguishing packet loss due to noise from packet loss due to congestion, so it always assumes congestion and responds to noise by slowing down. This feature of TCP is often used to imply that all Internet Protocols operate sluggishly or fail outright in the presence of noise. This is not true for UDP-*based*

protocols. UDP does not perform flow control and never attempts to throttle the data.

Many terrestrial environments feature noisy channels that successfully carry TCP traffic. The best example of this is the ordinary dialup telephone line. The telephone line has a bit error rate (BER) that is similar to most spacecraft RF links. The CCITT recommendations for voice circuits that have been conditioned to carry data [40] is a BER of 10^{-5}. Similarly, NASA typically specs its spacecraft RF links at a BER of 10^{-5} . The reason TCP works over the phone lines is that the modem applies error correction down at the physical layer, transparently to the upper layers. This allows the upper layers to behave as though they have a clean link. Similarly, NASA applies error correction to it's space links, achieving operational BERs down to 10^{-7} or better. At 10^{-7}, handshaking protocols, such as TCP/IP, work well.

In addition to this, current standards work is in progress to make TCP itself less sensitive to uncorrected noise loss. These include ECN, SACK (which is already widely distributed), and TCP/PEACH. These efforts are driven, in part, by the explosive demand for Internet-enabled cell phones and wireless devices, which must operate in an inherently noisy, low power environment.

9.3. Power, CPU, and bandwidth constraints

The previously mentioned wireless/cell-phone industry must operate in an environment that is far more severely constrained than that of most spacecraft. Electrical power, CPU processing power, and RF bandwidth are limited to an embedded device that fits in a shirt pocket. Much as in a deep space mission "every bit is precious", so ongoing research and development is being aimed at protocols that are efficient and error tolerant, such as IP header compression [41] and Cellular-IP [42]. These efforts are being coordinated through the Internet Engineering Task Force (IETF), so the resulting non-proprietary standards will interoperate and be available to all. In fact, given the huge potential size of the Internet cell phone market, it seems possible that in a few years, a large amount of Internet traffic will flow over cellular protocols.

9.4. Intermittent connectivity and variable routing

Spacecraft that are not in a geosynchronous orbit cannot maintain continuous direct contact with the ground. Contacts are limited to a brief time when the spacecraft passes within line-of-sight of the ground station. For a low earth orbit, this "pass" is typically no more than 8–15 min long, a few times a day. If more contacts are needed, more ground stations must be used, complicating the routing of data to and from the spacecraft.

This situation is very similar to people with laptop computers. They, and their computers, change locations and intermittently connect to the network at different points. But they want to maintain just one IP address. The Mobile-IP [43] protocol was designed to handle just this problem. Using it, mobile users can maintain a single Internet address while connecting to the network at different locations. Through the actions of a "home agent" and one or more "foreign agents", a "care of" address is established that allows transparent end-to-end addressing of data to and from the mobile host. This protocol does exactly what an IP spacecraft needs in order to send and receive data using multiple ground stations.

9.5. Forward/return path asymmetry

Most spacecraft have a much greater downlink bandwidth than uplink bandwidth. This asymmetry is often incorrectly attributed to the fact that spacecraft are limited by their power and weight budgets, and cannot generally support large steerable high-gain antennas. While this fact is true, it is not what limits the uplink data rate. Up to a point, any shortcomings of the spacecraft antenna or receiver can be compensated for by more power and bigger antennas on the ground. The real limitation is driven in part by physics, but mostly by convention.

In the early years of space exploration, most missions had modest uplink requirements for commanding. As a result, the standard RF systems for the evolving Spacecraft Tracking and Data Network (STDN) came to modulate the up-

link signal on a 16 kHz subcarrier, reserving the main carrier for ranging tones. Although this choice was adequate for the times, today this legacy of "STDN compatibility" limits the uplink channel to about 8 KBit/s instead of the 2 MBit/s that is possible when BPSK modulating the main S band carrier.

In any event, TCP will typically run with asymmetries of up to 50:1 before throughput begins to be affected. For an 8 Kbps uplink this corresponds to a 400 Kbps downlink. Although far below the maximum possible, this data rate is adequate for more than half of the current science missions. And this asymmetry limitation only affects TCP. UDP based protocols can always downlink at the full rate.

Missions that want to use TCP above the 400 Kbps rate will have to use a communication system that is not based on the 16 kHz subcarrier uplink. The NASA Tracking and Data Relay Satellite System (TDRSS) is one such system. Experiments with a ground-based IP spacecraft simulator were able to establish a symmetric 1 MBit/s duplex link through TDRSS using nothing more than a 5 W transmitter and an 18 in. steerable patch antenna [44].

10. IP-based operations scenarios

An IP-based communication architecture can support all existing operations concepts and makes some new, complex concepts realistic.

For example, real-time engineering and housekeeping data can be monitored during a pass using UDP/IP packets. This only requires a uni-directional downlink. On the other hand, if a bidirectional link is available, guaranteed reliable delivery of data packets can be achieved by using TCP/IP. In both cases, only a simple application layer function needs to be written for the flight software. The TCP/IP stack is available as a standard COTS product for current flight operating systems such as VxWorks.

Recorded science and engineering data can be stored onboard in files in a standard COTS file system. These files can later be transferred to the ground with guaranteed complete, time-ordered records using an off-the-shelf application such as FTP. If the round-trip delay times are too great to use a TCP-based protocol such as FTP, a UDP-based protocol, such as Starburst/MFTP, could be used instead.

Onboard clock synchronization, typically a thorny problem, can be handled by using the NTP. NTP can automatically calculate propagation delay times, time variance, and drift rates, relative to one or more reference timeservers. It can set the spacecraft's clock and even perform periodic drift mitigation. The NTP protocol is capable of precision on the order of 240 ps *if* sufficient bandwidth and CPU speed are available.

Store-and-forward commanding, and data delivery, can be achieved by using SMTP to deliver the files as email attachments. For example, in the commanding case, the control center emails the command load to the spacecraft as an attachment. A mail server at the groundstation stores the file until the next contact and then automatically transfers it to the spacecraft for processing. Similarly, in the data delivery case, the C&DH processor, or even a "smart" instrument, emails the data file as an attachment to a message sent to the principal investigator. The onboard mail server stores the file until the next contact and then transfers it to the ground for automatic delivery to the owner of the data.

The IP suite supports various commanding scenarios. When the downlink is not available for acknowledgement, "blind" real-time commands can be sent to the spacecraft using UDP. This is required for emergency situations such as rescuing a spacecraft from tumbling. For nominal operations, reliable commanding can be achieved by using TCP, which automatically takes care of performing the handshaking and any necessary retransmissions.

These basic capabilities, and the end-to-end network addressing capability of IP, can be used to support new, complex scenarios. These include user interaction, spacecraft cross-support, and ad-hoc collaborations.

These scenarios also highlight the capabilities needed for constellations of spacecraft. Formation flyers could send messages back and forth to keep their group navigation within specifications. Constellations of nanosats could message their data

to larger members with the power for delivery to the ground.

11. Ground-based demonstrations

In late 1998, the OMNI project began constructing a "proof of concept" ground-based prototype of an IP spacecraft. Initial demonstrations were performed with instruments in a van sending data back to a prototype control center at GSFC via NASA's Tracking and Data Relay Satellite System (TDRSS) links. These demonstrations used TDRSS Internet connectivity that had been installed and used to provide a communication link to scientists at the South Pole.

Many of the tests consisted of simply sending a one-way data stream in UDP packets from the van, through TDRSS to White Sands, and having them routed back to GSFC with standard Internet addressing. There was no forward link or uplink to provide two-way communication. These demonstrations were both easy to schedule, since they only required minimal TDRSS support for Multiple Access (MA) Return-only service, and they tolerated the intermittent TDRSS connectivity as the van drove around GSFC between buildings and under trees. This also demonstrated the use of Internet Protocols for simple data flows very similar to those used on today's spacecraft with TDM and CCSDS data structures.

When two-way communication service was available, these tests also included other protocols such as file transfers using both FTP and NFS and audio and video streaming.

12. Space-based demonstration

In late 1999 the OMNI project had been looking for opportunities to test these "Internet in Space" concepts on an orbiting spacecraft. 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 framing in hardware to allow simple, straightforward interfacing with existing commercial routers. These requirements made UoSAT-12, a spacecraft launched in May 1999 by Surrey Satellite Technology Ltd. (SSTL), an ideal test platform, as it already used HDLC framing to carry its AX.25 protocol. The AX.25 protocol and HDLC framing have been used on over 20 spacecraft over the last 10 years. Since HDLC interface 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.

12.1. Ground station implementation

Since the SSTL ground station already supported HDLC framing, a standard Internet router was the only addition needed. Fig. 11 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 only station reconfiguration required was to select which system is connected to the transmitter. This is done

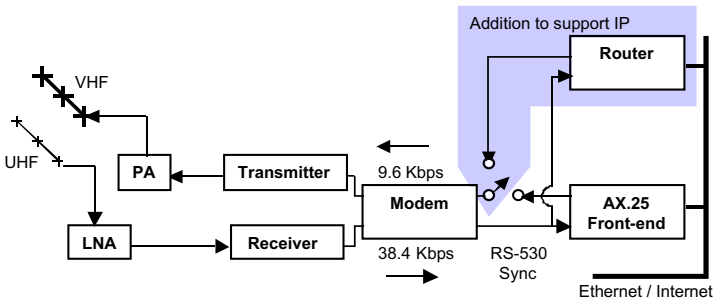

Fig. 11. SSTL ground station configuration.

with a controllable switch, which 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.

12.2. Flight tests

In February 2000 work was initiated to port a standard IP stack to the Spacecraft Operating System (SCOS) used on the UoSAT-12 spacecraft. In April 2000 the first basic connectivity tests using IP to a spacecraft were performed. Standard ICMP echo request (PING) packets were sent from both GSFC and the Surrey ground station to the spacecraft as shown in Fig. 12. The packets passed through a standard router at the Surrey ground station and were transmitted to the UoSAT-12 spacecraft. The standard IP stack onboard UoSAT-12 returned echo response packets addressed to the separate sources. Those packets then passed through the ground station router and were delivered to their respective destinations using standard Internet routing. These tests verified proper operation of both the end-to-end IP routing and the HDLC framing on the space-to-ground link.

The results from a PING test to verify basic HDLC and IP operation are shown in Fig. 13. PINGs were sent to UoSAT-12 continuously from the router at SSTL while a PING was sent once every 10 s from NASA/GSFC. The figure shows the successful replies from the spacecraft from AOS to LOS. The variation in the propagation time to UoSAT-12 on the horizon at AOS (approximately 3000 miles) to the highest elevation overhead (approximately 400 miles) is shown in the slight curvature in the response time plot. The bottom line in the plot shows the antenna elevation. The curve above it shows the theoretical round trip time computed based on data rates and distance. The large number of data points and the curve fit inside of them are the actual round-trip times for each PING response received. The line through them is a curve fit that corresponds very well with the theoretical line with the difference being the processing time on the spacecraft and ground equipment. The top data points are the PING responses recorded at GSFC.

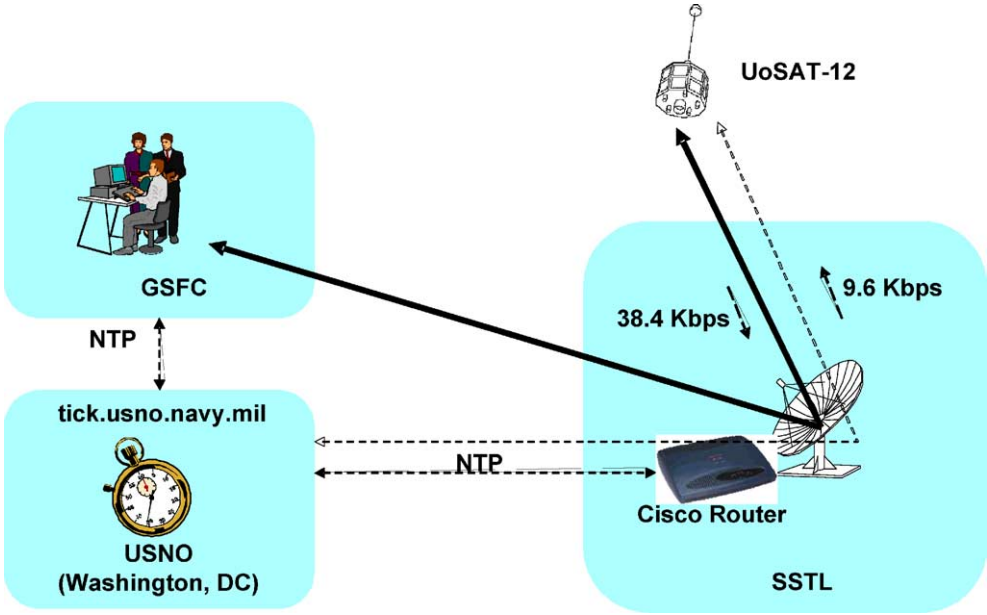

Fig. 12. UoSAT-12 network overview.

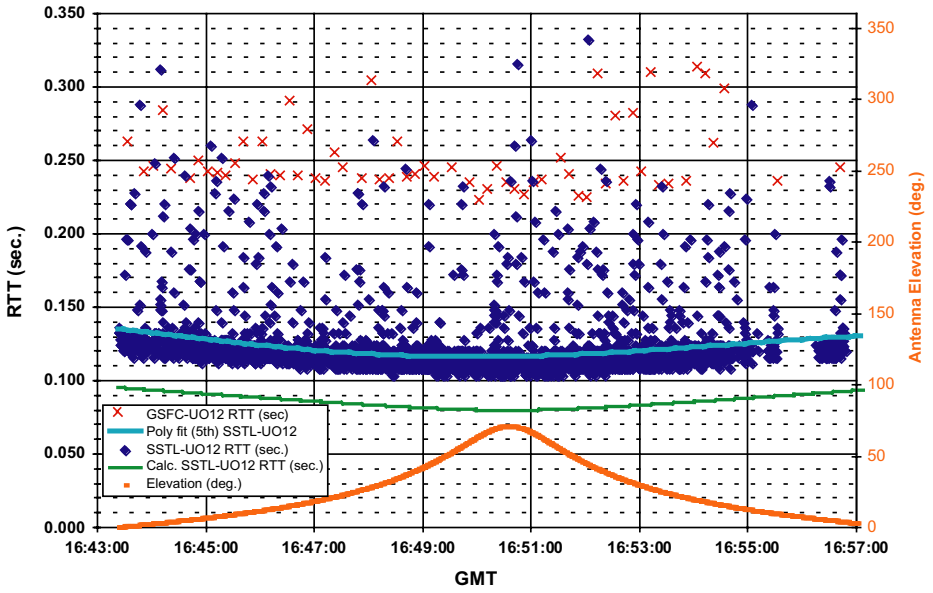

Fig. 13. UoSAT-12 PING test results.

Once the end-to-end connectivity was operational, additional tests were performed to have the spacecraft automatically set its clock using the Network Time Protocol (NTP) [45] by referencing a time server (tick.usno.navy.mil) at the US Naval Observatory (USNO). This represents somewhat of a worst-case test, as the USNO is across the ocean, 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 ground-station to minimize the network latency and variation that NTP as to factor out. Tests were also performed using the standard File Transfer Protocol (FTP) to retrieve files from UoSAT-12 and to send files to it.

Current information on test results and future activities will be posted on the OMNI project Website at http://ipinspace.gsfc.nasa.gov/.

13. UoSAT-12 IP test results

The OMNI project has successfully performed both ground-based and on-orbit validation tests of many of the concepts described in this paper. In particular, on-orbit testing of UDP telemetry delivery, NTP (operating over UDP) and FTP (operating over TCP) were successfully completed with UoSAT-12. Some results are presented here, but for full details, refer to the papers titled "*Demonstrations of Internet Protocols in Space Using TDRSS*" [44] and "*Results of 'Internet in Space' Tests Using UoSat-12*" [46].

13.1. On-orbit clock synchronization with NTP

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 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 ground station 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 tests started out with the onboard NTP server running, but disabled from actually changing the spacecraft's clock. The onboard server periodically negotiated with the USNO timeserver to factor out network delay. If it was successful, the onboard server calculated the offset it thought it had to apply to the spacecraft's clock. This value was sent to the ground in a UDP telemetry stream, where it was logged for later analysis. For purposes of this testing, the NTP negotiation period was set artificially low to 30 s so that a reasonable number of data points could be collected during the 14 min pass. A short time into the test, a command was sent to the spacecraft to enable NTP to actually change the onboard clock. NTP requires two successful offset calculations before it will adjust the clock. Later in the test, a command was sent to the spacecraft to manually set the onboard clock in error by a large amount (2–3 s). After two successful offset calculations, NTP should again reset the clock. If the time is off by more than 1 s, the spacecraft NTP client adjusts to the proper second during one adjustment period, and adjusts for fractional seconds on the next adjustment period.

The results for the test run on April 14, 2000 are shown in Fig. 14. 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 s. 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.

13.2. Error-free downloads with FTP

Current space missions, such as Landsat-7, download their image data "open-loop", without any automatic retransmission. Any data lost due to noise is lost forever. As a result, Landsat-7 employs large ground antennas and strong Reed–Solomon forward-error-correction in order to reduce its nominal bit error rate down to the range

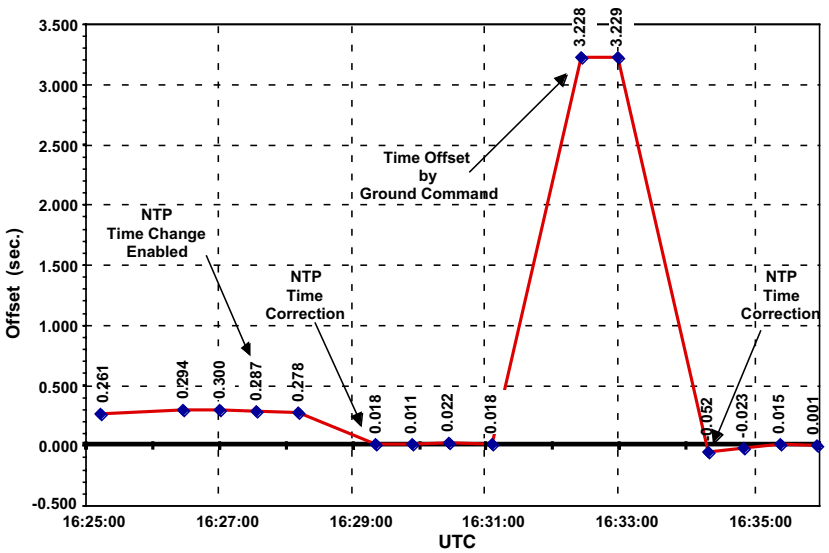

Fig. 14. NTP clock synchronization test results.

of 10^{-7}–10^{-9}. Even so, image data is sometimes still lost and results in ''dropped scan line'' flaws in images.

UoSat-12 does not employ any forward-error-correction, and its ground station employs modest antennas. As a result, UoSat-12's downlink only has a typical bit-error-rate between 10^{-5} and 10^{-6}. Conventional open-loop space mission protocols would provide badly degraded data at best. What is required is an application built on top of a protocol that performs automatic retransmissions, such as FTP over TCP.

For the initial FTP tests, a standard FTP server was ported to the UoSat-12 spacecraft. It was used to provide reliable, error-free transport of UoSat-12 image data to both the ground station and remote user sites using off-the-shelf FTP client applications. For subsequent tests, packet trace software was installed on UoSat-12 and at the ground sites in order to capture and quantify the number and types of lost packets and retransmissions.

Fig. 15 shows a result of tests performed on June 7, 2000. This mosaic consists of four sequential images of Perth, Australia that were downloaded from UoSat-12 via FTP with 100% data integrity, despite numerous packet losses. Note the lack of any data loss artifacts, such as dropped scan lines, shear misalignment, or pixelation.

The cost of this reliability in the face of adverse bit-error-rates is a reduction in performance. Our laboratory tests have shown that FTP/TCP performance is not significantly affected at bit-error-rates below 10^{-7}. Above this, performance falls to around 60% bandwidth utilization at 10^{-5}.

Subsequent FTP testing on July 5, 2000 began to characterize the on-orbit performance. Fig. 16 shows the packet trace for a typical file download. This 227 KB file required 445 packets and experienced nine retransmissions, all due to packet losses on the downlink. These retransmissions are noted by the ''O'' label, signifying receipt of an out-of-order packet. Seven of the nine were single packet losses, which allowed the TCP ''rapid-retransmission'' algorithm, but two were multiple packet losses, which resulted in a retransmission timeout. The overall result of these retransmissions was a reduction in bandwidth utilization to 79.2% compared to a theoretical maximum of 91.6%. These results are preliminary, but are in good agreement with the laboratory testing.

14. Missions using standard IP communication

The primary reason for using Internet Protocols on space-to-ground communications links is to take advantage of the hardware and software available in the commercial network world. However, the space environment does pose some challenges that require selecting the proper

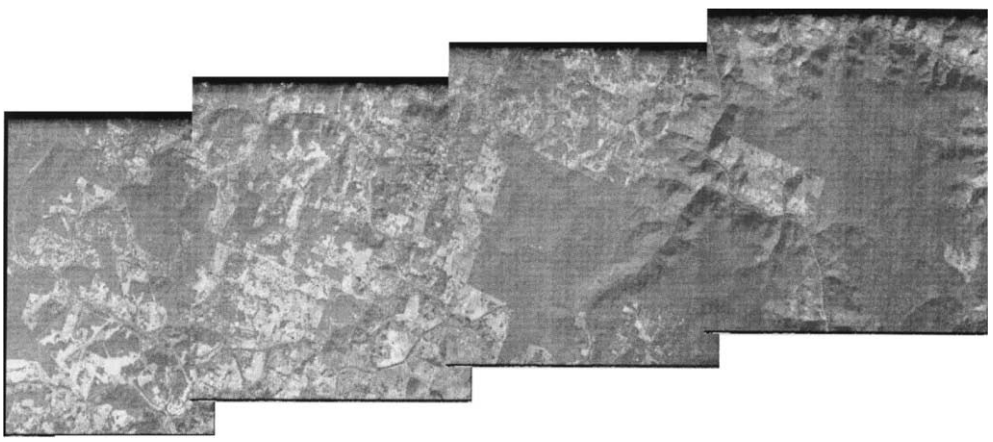

Fig. 15. UoSat-12 images downloaded error-free with FTP.

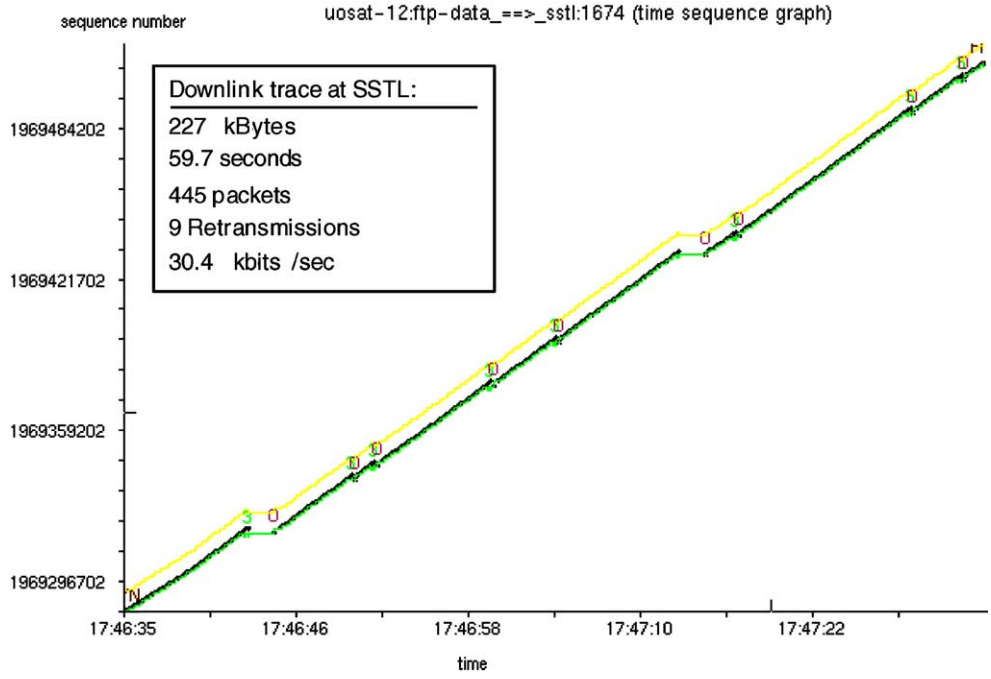

Fig. 16. Packet trace of UoSat-12 download with FTP (Fig. 4).

technologies and protocols for space communication systems. The initial challenge is to identify a data link framing mechanism that supports IP while also performing well over RF links used by space missions.

Some missions have flown, and others are being designed, using a full range of IP technologies in their communication systems. All of these missions use high-level data link control (HDLC) framing, which has been used on over 70 spacecraft during the last 20 years. Most of the spacecraft using HDLC framing were built by the amateur radio community or other low-budget organizations. Many of these missions were developed during the late 1980s and early 1990s when Internet technologies were not as widely deployed. Since many of the missions came from the amateur radio community, they used the amateur radio X.25 (AX.25) protocol over HDLC frames instead of IP.

The NASA community looked at the X.25 protocol in the late 1980s and concluded that it was not suitable for NASA's space communication needs. In the early 1990s, the commercial network world decided that the continual acknowledgement traffic of X.25 was too complex and added too much overhead and potential traffic delay. At this point the Frame Relay protocol was developed as a simple replacement for X.25. As its name suggests, it provides a simple frame forwarding service with no retransmission features and minimal flow control signaling.

The frame forwarding features of the frame relay protocol provide a simple framing mechanism very similar to those traditionally used for space missions (e.g., TDM frames, CCSDS frames). Two major differences from traditional space framing are:

The combination of frame relay over HDLC provides variable length frames. This allows the frame to fit various size user packets and avoids the packet insertion and extraction processing required with fixed length frames.

The frame relay protocol has been widely implemented in millions of commercial routers and frame relay switches worldwide. This pro-

vides low cost equipment and direct interfaces of space data with worldwide common carrier networks.

The IETF has also defined a standard mapping of IP packets over frame relay and this is widely supported by standard routers.

In 1997, NASA/GSFC started using standard Internet Protocols over HDLC on the South Pole TDRSS Relay (SPTR) system. This system uses standard routers at the South Pole and at White Sands to deliver Internet traffic. This connectivity is used for both data and phone service to the South Pole. This service has been and continues to be very successful in providing communication to the South Pole facilities.

In late 1999 NASA/GSFC initiated a search to find a spacecraft that could be used to test the operation of Internet Protocols to an orbiting spacecraft. This search identified the UoSAT-12 spacecraft that had been launched in April 1999 as a possible candidate. The primary qualification was that UoSAT-12 used the AX.25 protocol, which meant that it had HDLC framing support in its onboard hardware. It also had a 386 processor, which provided an easier target to find existing IP software.

15. U0SAT-12 (Surrey Satellite Technology Ltd.)

Surrey Satellite Technologies Ltd. (SSTL) designed and built the UoSAT-12 spacecraft which provided a suitable platform for initial tests of IP communication to an orbiting spacecraft. It supported data rates of 38.4 Kbps down and 9.6 Kbps up, which provided an environment similar to dialup modems. The link did not use any forward-error-correction coding (e.g., convolutional, Reed–Solomon). Consequently the link had to endure some noise and errors but this did not cause any serious problem since the Internet Protocols can operate over noisy links. The spacecraft used a DOS-like operating system called the spacecraft operating system (SCOS). Software was developed to incorporate a Berkeley Software Distribution (BSD) IP stack with the operating system. Applications were also developed to support Network Time Protocol (NTP), File Transfer Protocol (FTP), Hyper Text Transfer Protocol (HTTP), and UDP packet delivery applications.

The first goal of the UoSAT-12 test consisted of simply sending Ping packets to verify the proper operation of the IP/Frame Relay/HDLC space link communication path. Initial Pings were sent from the router in the SSTL control center directly to the spacecraft. This verified the proper operation of the IP stack onboard UoSAT-12, the RF components, and the ground router. Next, Pings were sent from NASA/GSFC to the spacecraft via the Internet, This demonstrated that the spacecraft was able to perform standard IP addressing functions of accepting IP packets from anywhere and using the sender's address to tell it how to send data back to wherever it came from. Later tests included simultaneous Pings from predetermined sites in England, California, Pennsylvania, and Maryland. This demonstrated full addressing capabilities with the spacecraft communicating with multiple ground systems simultaneously.

With IP connectivity established, an NTP application was added to the flight software to test the use of NTP for automatically maintaining the spacecraft clock. NTP was set to communicate over the space link to SSTL and then across the ocean to a NTP server at the United States Naval Observatory (USNO) in Washington, DC. This was not an ideal operational environment but it provided a worst-case scenario. Initially NTP was operated in a shadow mode where it did its time synchronization computations but did not actually change the spacecraft clock. After the first passes demonstrated proper operation of the onboard client, some additional passes were completed where NTP was allowed to check with the Naval Observatory, compute the current time offset, and set the spacecraft clock to the current time. NTP was set to do four time checks every 30 s. If it received four consistent measurements, it readjusted the clock. Otherwise, it did nothing and tried again 30 s later. On one pass it adjusted the clock by about 320 ms at the start of the pass and then made adjustments of 1–5 ms during the pass. At one point near the end of the pass the SSTL controllers intentionally commanded the clock off by over 2 s. NTP proceeded to reset it to the correct time at its next execution. This successfully demonstrated NTP operation in space, but further work is needed to determine the exact

precision achievable and the effects of CPU speed, link data rate, Doppler, and link errors.

The next test was to install an FTP server onboard and use it to pass files to and from the spacecraft. During one test, two different workstations were used at NASA/GSFC to log into UoSAT-12 using FTP, examine the directory, download files, and upload files. The file transfers completed successfully and the files were received intact. This demonstrated the automated retransmission features of FTP using TCP to provide in sequence, reliable delivery of data over a noisy link. Network analyzers were used to verify the automatic retransmission of packets to fill in lost packets. This test demonstrated multiple users simultaneously and seamlessly accessing data on the spacecraft.

Another standard type of spacecraft data is real-time data containing spacecraft housekeeping information and science instrument telemetry. To test this type of data flow, an application was added to the flight software that collected parameters onboard, inserted them into a UDP packet, and sent the UDP packets to a specified network address. These packets were received by a workstation running the Integrated Test and Operations System (ITOS) control center software, where the packets were decoded and the individual telemetry values displayed and plotted. These tests demonstrated the ability to use standard UDP/IP/HDLC as an alternative to traditional real-time housekeeping and telemetry using TDM or CCSDS frames.

After the success of all the other protocols, a simple Web server supporting HTTP was added to the flight software. Final tests were performed using standard Web browsers to access data from UoSAT-12. A simple starting Web page was installed on the spacecraft that pointed to some images and a simple telemetry page. The images were retrieved with standard Web page clicks. Selecting the telemetry page retrieved a page with some formatted telemetry information,which automatically updated every 10 s. These tests proved that standard Web server and Web browser technologies could be used to access spacecraft data. This is not necessarily an approach that would be used for full operational access to spacecraft data but it might be used as a simple mechanism that anyone could use to provide basic access during integration and test or some operational scenarios.

The final tests with UoSAT-12 occurred during the first Space Internet Workshop at NASA/GSFC. About a week before the workshop, the team considered having a live demonstration of IP data from a spacecraft. The contact times of UoSAT-12 over England were examined but it turned out that all contacts occurred in the evening and very early morning when the workshop was not in session. The best spacecraft contact times to support the demonstration occurred over the west coast of the US, so all that was needed was a supporting ground station. Stanford University had extensive experience working with other spacecraft using the same frequencies as UoSAT-12 and they agreed to provide support. A loaner router was shipped to Stanford on Wednesday, five days before the start of the workshop. Stanford engineers connected the router to the output from their receiver in two days and by Saturday the UDP data packets from UoSAT-12 were flowing to GSFC via the Stanford ground station. Stanford did not have the correct uplink equipment, so packets addressed to the UoSAT-12 spacecraft would be routed to SSTL and not to Stanford. This meant that Stanford only supported a oneway downlink; the UDP housekeeping packets received at Stanford were transmitted to GSFC. This activity demonstrated the ease of installing IP capabilities in a ground station and the ability of UDP packets to find their way to their addressed destination no matter where they are received on the ground.

15.1. UoSAT-12 summary

UoSAT-12 was the first known test of using standard Internet Protocols to an orbiting spacecraft using standard off-the-shelf routers and commercial link layer protocols end-to-end. It successfully demonstrated the use of many standard Internet Protocols and applications providing a wide range of data delivery options and automated spacecraft operations.

However, the UoSAT-12 tests did not cover all aspects of a full operational mission. It used a simple addressing mechanism where the spacecraft IP address was selected from the subnet at SSTL, which was the only ground station used. This was a normal routable address on the Internet, which meant that any packets addressed to the spacecraft would be routed to SSTL and then forwarded to the spacecraft. This approach worked fine for these tests but does not scale well to missions using multiple ground stations. Scaling for the general case will involve Mobile IP technology, which will be discussed in subsequent sections.

These tests did not use extensive security measures. The very first Ping tests were performed with the spacecraft accessible to the Internet. The main security was that it was only accessible during a few selected passes over SSTL, which only lasted for 6–8 min each. During following tests, the router at SSTL had access filters configured, which only allowed packets from a few select locations, such as GSFC, to get through.

These tests only used low data rates of 38.4 Kbps down and 9.6 Kbps up. The router was capable of handling Mbps rates but the UoSAT-12 transmitters and receivers were not built for those rates.

Overall these tests were very successful in demonstrating the ease with which Internet Protocols could be used to perform basic spacecraft communication functions. Most of the hardware and software used was standard CPUs, operating systems, network applications, network hardware, and the Internet. These tests demonstrated many of the benefits of using Internet technologies for spacecraft communication.

15.2. AlSAT-1—Surrey Satellite Technology Ltd.

SSTL took the lessons learned from the successful UoSAT-12 IP tests and designed their Disaster Monitoring Constellation (DMC) series of five spacecraft to use IP. The first activity was to clean up the implementation of the IP stack in the SCOS operating system to provide a fully functional, standard application-programming interface for TCP and UDP sockets. The earlier version was not fully standard and required more work to port standard IP applications to the SCOS environment. While the IP stack and applications on UoSAT-12 were added after launch, the AlSAT-1 flight software contained both the traditional SSTL AX.25 support and a standard IP stack as part of the initial design. This was easy because both protocol stacks use the same low-level HDLC framing hardware.

SSTL installed multiple ground stations using standard Cisco routers similar to those used with UoSAT-12. The main difference was to use newer, more powerful routers since the AlSAT-1 downlink runs at 8 Mbps instead of the 38.4 Kbps of UoSAT-12. The network address for the spacecraft was also changed to use private address space and security protocols to get to the router at the ground station. This reflects the shift from the simple test environment used for UoSAT-12 and the full operational environment for AlSAT-1.

15.3. AlSAT-1—Summary

AlSAT-1's usage of Internet Protocols has worked very well. It allowed AlSAT-1 to operate at data rates of 8 Mbps without any special effort to build custom, high-rate front-end communication processing equipment. A very low-cost off-the-shelf router was able to support the data rates and also provide a good range of security protocols to support the necessary security needs of this international mission. Similar IP protocol usage has been designed into the other four DMC spacecraft, with the following three of them launched in September 2003:

- BILSAT for Turkish customer Tubitak-ODTU Bilten.
- NigeriaSat-1 for Nigerian customer National Space Research and Development Agency.
- UK-DMC funded by the UK government/BNSC.

The UK-DMC spacecraft also included an experimental version of a Cisco mobile router onboard the spacecraft. Testing of the mobile router was successfully completed in June 2004 by NASA/GRC.

15.4. CHIPSat—University of California Berkeley/ NASA

The Cosmic Hot Interstellar Plasma Spectrometer Satellite (CHIPSat) is the first NASA mission to utilize the IP/Frame Relay/HDLC protocol stack as its communication system. This is also the first mission known to use IP/Frame Relay/HDLC as its sole communication protocol stack with no alternate communication mechanism. The CHIPSat mission designers closely monitored the tests performed with UoSAT-12 and decided that IP provided the most cost effective solution for CHIPSat. CHIPSat is a NASA University-class mission with a budget cap of $14M for all aspects of the mission. The mission designers were looking for solutions that would provide the maximum capability for the least cost. Using IP allowed them to save significant time and effort in their communication system by simply using existing Internet hardware and software and capabilities built into standard operating system software.

CHIPSat does not use Mobile IP but instead uses private, non-routable address space for the spacecraft address, ground stations, and control center. Virtual Private Network (VPN) tunnels are used to secure connections between the control center and ground stations. CHIPSat can address packets such as UDP housekeeping data directly to the private address at the control center. This data is routed to the control center within the static routes of the CHIPSat VPNs. Many of the other data transfers are performed over the point-to-point link from the current ground station computer to the spacecraft, so no Mobile IP routing is needed.

The spacecraft takes measurements and records them in files along with housekeeping information. During ground contacts, automated scripts use FTP/TCP to retrieve the files from the spacecraft. These transfers occur between the spacecraft and a Linux system at each ground station and the files are later transferred from the ground station computer to the control center at the University of California Berkeley (UCB). Files with stored commands are also uploaded to the spacecraft using FTP.

Using the open Internet as part of the control center-to-spacecraft communication path has presented a few problems. It provides a very low cost wide-area network, but it does not provide the highly reliable communication path used by other NASA spacecraft. Any congestion or circuit outages on the Internet can result in loss of connectivity and the CHIPSat operators just have to wait until other organizations resolve those problems. Using UDP packets for basic housekeeping and telemetry means that some packets have gotten lost between the ground stations and the control center. However, this loss has been within allowed limits. As mentioned earlier, most of the file transfers occur initially between the spacecraft and computer at the ground station. This helps avoid some of the issues with delay and congestion on the open Internet.

CHIPSat uses standard NTP to automatically maintain its spacecraft clock. It does not require high precision (e.g., millisecond) spacecraft timing and NTP provides sufficient precision in a fully automated environment, which helps in reducing operational costs.

15.5. CHIPSat Summary

The CHIPSat use of Internet Protocols as its sole communication mechanism has worked very well. The CHIPSat mission has met all of its objectives and the communication system has performed flawlessly. NTP has provided a simple, automated mechanism to maintain the spacecraft clock. However, CHIPSat does not have any high-precision time resolution requirements and NTP just needs to keep the time to subsecond accuracy to help keep track of file times and simple time stamps.

CHIPSat uses the standard FTP/TCP protocol for its file transfers, which is not highly optimized for space use. However, there is sufficient spacecraft contact time so efficiency is not a key issue.

15.6. CANDOS—NASA GSFC

NASA/GSFC had already performed a wide range of tests using Internet Protocols in space,

but in late 2001 an opportunity arose to perform more advanced tests as part of the Communication And Navigation Demonstration on Shuttle (CANDOS) mission. A primary goal of this experiment was to test the new Low-Power Transceiver (LPT) in the space shuttle payload bay during the STS-107 mission. The LPT provided a transceiver capable of supporting multiple data rates, using both NASA ground network (GN) stations and space network (SN, TDRSS) relay satellites. The data rates used during the mission ranged from 2 Kbps up to 128 Kbps in various combinations of symmetric and asymmetric rates as well as one-way links. The payload included an X86 processor running a version of Linux to control the transceiver, perform GPS computations and run Internet applications. When launch delays provided additional time to incorporate new concepts, the project decided to install additional software to investigate the performance of Internet Protocols and applications to provide more automated operation and increased security.

In Fig. 17, the LPT is the shiny, shoebox-size object in the lower left just under the round 12-in. high-gain transmit antenna. There are three other lower gain but wider angle 3-in. antennas for low-gain transmit and receive and GPS receive. The components were all mounted on a truss installed in the tail of the shuttle bay. Fig. 18 shows the connectivity paths using NASA ground stations and TDRSS.

Some of the primary Internet Protocol and automation experiments added involved:

- Using Mobile IP to automatically set up routing tunnels to send uplink traffic to the correct GN or SN location for uplink.
- Using the Multicast Dissemination Protocol (MDP) for automated, reliable file transfers using UDP.
- Running NTP for multiple days and letting it perform clock setting as well as manage clock drift.
- Using secure shell (ssh) and secure copy (scp) to perform secure access to the payload.

The shuttle launched on schedule January 16, 2003 and approximately 4 hours after launch the CANDOS payload was powered on. The processor booted and the transceiver was configured in its default mode. A housekeeping process was always running to collect status information and send a status packet to the ground every 10 s. These packets were sent as simple tab-delimited

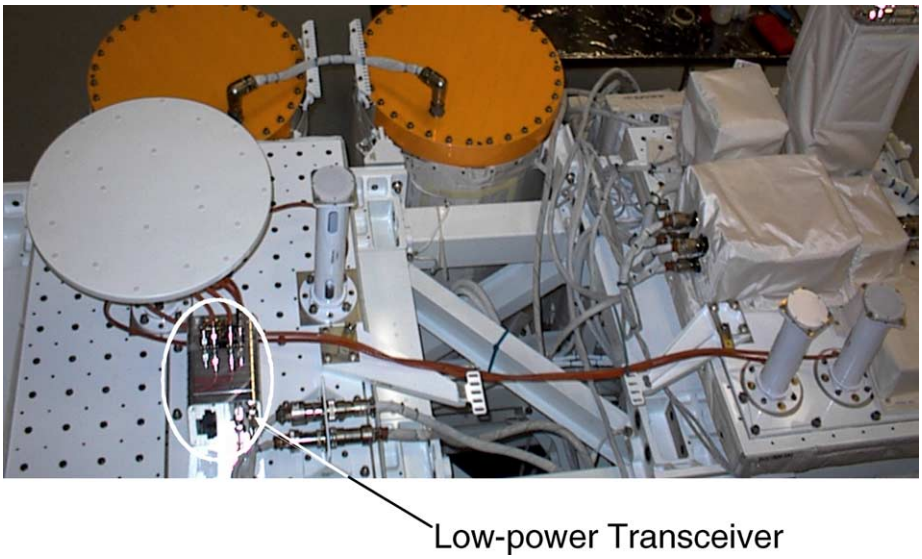

Fig. 17. Photo of CANDOS equipment and LPT.

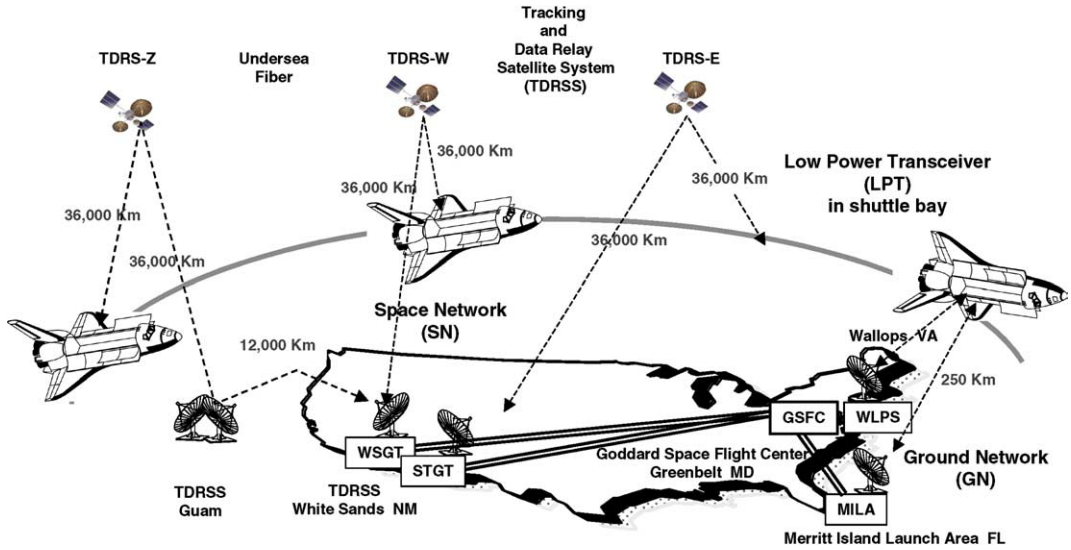

Fig. 18. Mobile IP network connectivity for shuttle to SN/GN stations.

ASCII strings in a UDP packet addressed to a computer in the CANDOS control center. There the packets were logged to a file, displayed on a screen, and forwarded to other systems for additional displays. These UDP packets were the first data received from CANDOS and they began arriving as soon as the ground receiver locked on and a downlink was established. No uplink or Mobile IP functionality was required to receive this data. These UDP housekeeping packets provided very useful status information and proved to be a very reliable indication that the downlink was operational. The control center was often able to verify that the downlink was operational before confirmation was reported by the ground station itself.

Once two-way communication was established on the first communication pass, the Mobile IP daemon detected advertisements from the ground station and automatically set up a routing tunnel. This path was used to initiate an FTP transfer to upload software upgrades. However, due to the slow 2 Kbps uplink, the FTP file transfer only completed 90% of the 300 KB file during the contact. It was only 11 min to the next scheduled contact, so a decision was made to wait and see if the FTP session would continue. When the next contact began, Mobile IP automatically established a new routing tunnel and after 1 min, the FTP session resumed and completed the remainder of the file transfer. This demonstrated the combination of Mobile IP changing a routing path completely transparently to any applications currently active and FTP/TCP automatic retransmissions recovering after a link outage. If the link outage had been much longer than 12–15 min the TCP session would have timed out and dropped the session. However, this time range can be adjusted by changing operating system parameters controlling TCP operation.

After the first day of CANDOS operation, the indication of the Mobile IP tunnel being established became the primary indicator that there was a good two-way link to the payload. Mobile IP requires a two-way link for the advertisements to go up and registration requests to come down. However, Mobile IP only needs to get one packet up and one packet down to set up a routing tunnel. There were times when a tunnel would come up but PINGs were only intermittently successful. This simply indicated that a few packets got through to set up the Mobile IP session, but the link still had enough errors to interfere with many PING request/response sequences. The real benefit of Mobile IP was the way it automatically handled IP routing to the current SN or GN, station and

did it all transparently to the upper layer protocols (TCP, UDP, FTP, NTP, etc.).

However, since Mobile IP does require a two-way link to determine the current link, it will not work in spacecraft communication scenarios with one-way links with only an uplink or downlink. These scenarios were examined during the CANDOS mission. Due to TDRSS resource constraints, it is often easier to schedule a TDRSS downlink only, without requiring an uplink. In this mode, packets simply arrive at a ground router and they are routed according to their destination address. This mode was exercised by scheduling TDRSS return-only service with processes onboard CANDOS scheduled to send only UDP traffic and not require any uplink.

The one-way downlink mode also occurred naturally during the initial minute of GN contacts. During GN contacts the spacecraft transmitter was normally on and housekeeping UDP packets were being sent continually. At a GN site the receiver locked on and these housekeeping packets were immediately passed to the control center. The GN site procedure was to send an unmodulated carrier to the spacecraft and the control center monitored the housekeeping data to verify that LPT receiver had locked onto the ground signal. Then the ground station began modulating the uplink and the two-way connection was complete, and then Mobile IP would set up its tunnel.

The other one-way link scenario of an uplink without any downlink is often referred to as "blind commanding". This may be used operationally to command the spacecraft transmitter to switch on, and it is also used in anomaly situations to attempt to blindly send commands to a spacecraft in the hope that it will respond. Often a spacecraft will keep its power hungry transmitter turned off most of the time but keep its receivers turned on. Then the ground can send a "blind command" turning on the transmitter and establishing a two-way link.

Mobile IP becomes necessary when trying to send data to a spacecraft that is moving among many stations. When the control center tries to send a packet to the IP address of the spacecraft, the ground routers do not know where to route the packet. If an old Mobile IP tunnel was still active and had not timed out yet, the packet would be sent to the last station and the Foreign Agent (FA) router would try to uplink it. However, normally the Mobile IP tunnel would have timed out and the Home Agent (HA) router would not know where to send the packet. This is where the Mobile IP protocol normally steps in with the Foreign Agent advertising, the Mobile Node responding, and the Foreign Agent informing the Home Agent where to route packets. With a one-way uplink, this process will not work and another approach must be available to get commands to the spacecraft. On CANDOS, a manual routing approach was demonstrated in which the ground network operators manually configure the Home Agent and the proper Foreign Agent by setting up the tunnel and two associated static routes. This provided exactly the same IP packet routing function of Mobile IP, but it required human intervention to set up the routing instead of having Mobile IP automatically set up the tunnel and routes.

Security protocols incorporated in the CANDOS mission included the SSH and SCP protocols along with Mobile IP security mechanisms. The Mobile IP protocol requires the use of authentication mechanisms between the Mobile Node (e.g., spacecraft) and the Home Agent (e.g., control center router). This prevents unauthorized systems from using the Mobile IP services advertised by the Foreign Agents at ground stations. Security authentication was also turned on between the FAs and the HA to provide further security. Mobile IP was configured with static authentication keys, which provide basic security, but future missions may want to consider security approaches with more dynamic key management mechanisms.

One major test of the LPT was to use its GPS receivers and the GEODE software to compute the location and speed of the shuttle. This experiment generated many multi-megabyte files, which needed to be retrieved during the mission. Other files were also generated onboard to log basic housekeeping information and NTP performance. Most of these data files were transferred to the ground using the Multicast Dissemination Protocol (MDP). The MDP protocol uses the UDP transport protocol instead of TCP. This makes it especially well suited to space use since it is not

sensitive to propagation delays and can even function across one-way links. The MDP application supported a "hot directory" that allowed files to be collected there between passes and then automatically transferred to the ground when the link came up. It also allowed the control center to prepare a directory of files for automated uplink transmission once Mobile IP established an uplink route.

Since the LPT processor was running Linux, much of the command and management of the transceiver and applications was performed using standard shell scripts and automated operations initiated using the "cron" process. Many commands consisted of an ASCII string containing the name of a shell script and parameters to pass to it. This allowed the addition of many new commands during the mission by simply uploading new shell scripts and then invoking them in future commands. For a long-term operational mission, this process should be made more secure by encrypting the command string before placing it into the UDP command packet.

During the mission some additional scripts were uploaded to demonstrate how IP enables a payload to send selected data to multiple destinations based on onboard decisions. Some data was sent to one destination, other data to a different destination, and some data was sent to both destinations. This demonstrates the full network addressing capabilities of Internet Protocols where the source system or space payload in this case, can determine where it wants various data packets sent to. This is completely different from current spacecraft where the only information provided on packets is the data source. In current TDM and CCSDS mechanisms, the ground data routing must be managed by other mechanisms that normally require extensive scheduling and manual interaction on the ground.

CANDOS also performed tests of the NTP protocol performing long-term clock maintenance. NTP does require a two-way connection so it was not able to update the time during one-way contacts. It did track clock drift and attempted to adjust the onboard clock to adjust for it. It often maintained the clock to within 10–20 ms, but at times it varied beyond that range. More work is needed to determine the limitations of NTP timing accuracy in the space environment.

15.7. CANDOS Summary

All mission objectives were met for the CANDOS mission. The mission successfully demonstrated functions such as HDLC framing, IP packet encapsulation over Frame Relay, and Mobile IP. Standard Internet applications such as SSH, SCP, Telnet, FTP, MDP, and NTP operated properly but their performance was a function of the varying uplink/downlink rates (from 2 Kbps to 128 Kbps) used across the contacts.

The UDP/IP/HDLC housekeeping packets and blind commands (using UDP) provided functionality identical to TDM and CCSDS frames currently used by spacecraft. The main advantage is the support for Internet Protocols and the ease with which software (operating systems support, PERL scripts, and applications) can be developed to use them.

This mission demonstrated Internet Protocols operating in space. However, a long-term operational mission still needs more security solutions for things like dynamic key management, user ID/password management, and more automated onboard file management.

15.8. Additional missions lessons learned

More lessons are being learned as new missions are designed and flown using IP for their communication system. The latest missions launched are the following SSTL missions launched in September 2003:

- BILSAT—for Turkish customer Tubitak-ODTU Bilten.
- NigeriaSat1—for Nigerian customer National Space Research and Development Agency.
- UK-DMC—funded by the UK Government/BNSC.

NASA missions that are contemplating using IP or that have baselined IP:

- Global Precipitation Measurement mission.
- Magnetosphere Multiscale mission.
- Lunar Reconnaissance Observatory.
- International Space Station.

Other missions that are contemplating using IP or that have baselined IP:

- Citizen-Explorer (Univ. of Colorado, Boulder, CO).
- EagleEye (Embry-Riddle Aeronautical University, Daytona Beach, FL).
- LionSat (Penn State University, State College, PA).
- MidStar (US Naval Academy, Annapolis, MD).
- Npsat (Naval Postgraduate School, Monterrey, CA).

16. IP mission lessons learned summary

All of these missions have provided a base of knowledge and experience on the use of Internet Protocols for a range of space communication environments. The basic lesson learned is that through an understanding of a mission's communication needs and end-to-end system engineering and design, Internet Protocols and technologies can be selected to meet those needs. The following lists traverse the protocol stack upwards to summarize lessons learned from space missions that have used Internet Protocols.

16.1. Data link protocols (framing, frame error detection, virtual channels)

- HDLC framing provides a simple framing mechanism that has been used in space communication systems for over 20 years.
- HDLC framing supports variable length frames, which allows simple packet insertion and extraction by sending one packet per frame.
- HDLC framing always uses a CRC-16 error check to identify and discard any frames with bit errors. Any frames received are intact and further data processing is simplified, since data either is good or is not forwarded to the destination.
- HDLC framing is not affected by one-way links or propagation delay, since it has no ACK/NACK mechanism and it operates identically over links at any data rate, distance, or delay.
- HDLC framing does require a clean enough link to receive a frame of data with no errors. If a link has a high error rate, it should be cleaned up with forward-error-correction (FEC) coding such as convolutional coding, Reed–Solomon, Turbo Product Codes, or Low Density Parity Check. The combination of FEC and HDLC provides excellent data recovery for space links.
- Frame relay headers and RFC 2427 (Multi-protocol Encapsulation over Frame Relay) provide a standard serial link format that is supported by numerous COTS network equipment vendors.
- The Frame Relay data link connection identifier (DLCI) field can be used to provide 1024 virtual channels similar to current CCSDS virtual channels.

16.2. Network protocols (packet addressing, packet routing)

- IP addressing provides fully identified packets with both source and destination addresses, which identify both where the data came from and where it should be delivered. This mechanism enables data-driven data delivery, as opposed to the scheduled processes used for current spacecraft. This allows the spacecraft to control where the data is routed whether it be to different facilities or other spacecraft.
- IP addressing is not affected by one-way links or propagation delay, since it has no ACK/NACK mechanism and it will function over any link at any distance.
- The overhead of IP packets is higher than for legacy space protocols but provides more functionality and the overhead is a direct function of packet size. With larger packets of 1000–1500 bytes the overhead difference is insignificant (1–3%).

- Tools for IP communications trouble shooting are off-the-shelf items.
- Mobile IP provides a lightweight mechanism for the spacecraft and ground station to automatically set up a route for sending packets to a spacecraft without knowing which station is scheduled.
- Mobile IP does require a two-way communication link to operate. If a two-way link is not available, the IP routing must be set up manually in the routers, as might be the case for blind commanding.
- Mobile IP only needs to get three packets across the space link in order to set up a route. It can operate over links that have high error rates (even worse than 10^{-5}).
- PING(ICMP echo request/response) provides a simple, standard mechanism for verifying proper end-to-end operation of a two-way IP data path to a spacecraft.

16.3. Transport Protocols (subchannels (ports), unreliable/reliable delivery)

- UDP packets provide an alternative to legacy TDM and CCSDS frame/packet mechanisms for sending packets of data across one-way and two-way data links.
- UDP packets are not affected by propagation delay and will function over any distance.
- UDP packets support 65,535 subchannels (ports) to identify different types of data. This mechanism can be used similar to current CCSDS APIDs.
- The standard UDP socket application programming interface (API) provides a simple, standard mechanism for sending and receiving data using any programming or scripting language.
- TCP provides a mechanism for reliably delivering a byte stream across a two-way communication link, but it requires a two-way link in order to operate.
- TCP performance is a function of data rate, propagation delay, and link error rates.
- TCP can be used without any modifications over space links if the delays are reasonably low (e.g., 1–2 s), data rates are low (tens of Kbps), and error rates are low ($<10^{-6}$).

16.4. Applications

Time measurement

- PING (ICMP echo request/response) provides a simple, standard mechanism for measuring the round-trip propagation time over a network data path.
- PING combined with the IP header timestamp option provides a standard mechanism for measuring the relationship of a spacecraft clock and ground network devices. This provides a mechanism for reading the spacecraft clock and its relationship to the time of the uplink echo request and the receipt of the echo response at the ground station.
- NTP provides an automated mechanism for maintaining a spacecraft clock to within a fraction of a second. The absolute accuracy that NTP can achieve over various space links and satellite computers is a function of many variables and has not yet been determined.

File transfer

- The multicast dissemination protocol (MDP) provides UDP-based reliable file transfer that performs well over space links with minimal performance degradation due to data rates, propagation delays, and link bandwidth asymmetry.
- MDP supports options for doing highly reliable file transfers over a one-way communication link by sending additional Reed–Solomon forward-error-correction packets separate from the data file.
- FTP will work for performing reliable file transfers in space communication environments and is being used operationally by the CHIPSat mission. However, since FTP operates over TCP, it should only be used

in conditions where the impact of data rates, propagation delays, link bandwidth asymmetry, and link errors have been properly analyzed and understood.
- SCP performs reliable file transfers over TCP just like FTP. It does add some additional overhead for setting up a secure connection and for encrypting the data over the link. Otherwise its performance issues are similar to FTP/TCP.
- CFDP performs reliable file transfers over UDP and performs similarly to MDP. It does not support the one-way FEC assisted file transfers or multicast addressing options of MDP.

Remote login

- Telnet provides a low-overhead, remote-login service that will operate over space links. It allows an operator to directly send command and control functions to a spacecraft's operating system with reliable data transfer provided by TCP. However, since Telnet operates over TCP, it should only be used in conditions where the impact of data rates, propagation delays, link bandwidth asymmetry, and link errors have been properly analyzed and understood.
- SSH provides a remote-login capability similar to Telnet, but it first establishes a secure connection and then encrypts all data transferred over the link. There is some additional overhead for the secure connection setup and encryption. Otherwise its performance is similar to Telnet/TCP.

16.5. Test and analysis equipment

- LAN analyzers are readily available to receive, store, and decode IP traffic on Ethernet LANs. These tools range from free, public-domain software (e.g., Ethereal), to commercial software packages, to full hardware systems. These analyzers support full packet decodes for standard IP protocols and some allow users to add decodes for additional protocols.
- WAN analyzers are readily available to receive, store, and decode HDLC frames carrying IP traffic on serial links. These provide insight into the exact data packets traveling across the space link.
- LAN and WAN analyzers provide immediate insight into data packets flowing on communication links without requiring any development of special test equipment.
- LAN and WAN analyzers often identify traffic that is generated automatically by various protocol stacks and that may not be expected by the system designers.
- Decoded packet information such as time-stamps, sequence counts, and packet lengths can be extracted from LAN/WAN analyzer decodes and processed with standard spreadsheet packages to understand protocol performance issues.
- Plots of protocol parameters can provide quick analysis of protocol performance and identify periodic events that might be missed during normal examination of the data.
- Protocol analysis software such as "tcptrace" is freely available to perform more complex analysis and plotting of TCP operation by examining packets captured by LAN and WAN analyzers.

While some Internet Protocols may not be well suited for particular types of space communication environments, the Internet Protocol suite provides a wide enough range of protocols so that with a bit of analysis and understanding, standard protocols can be identified to meet the communication needs of space missions. A significant benefit of being able to select from these standard protocols is their wide support and implementation in operating systems, applications, network equipment, and test equipment. Missions like CHIPSat and the SSTL DMCs (AlSAT-1, NigeriaSat-1, BILSAT, and UK-DMC) have used this to keep costs down while rapidly developing, deploying, and operating new spacecraft.

Current information on test results and future activities are posted on the OMNI project Website at http://ipinspace.gsfc.nasa.gov/.

17. Future work

Providing true end-to-end networking requires changing the sources of data in range and space systems to be full network devices. This includes devices such as command and data handling systems, science instruments, power systems, attitude control systems, and storage systems. Another option is for these data sources to be connected to a computer via direct interfaces and the computer then provides the network access to the instruments. However, in either case, components are needed to complete the network architecture for range and space systems.

The primary reason for the missing pieces in this area is the challenging environment in which the components must operate. This includes issues such as:

- radiation—some radiation hardening in LEO orbits; higher and polar orbits need more hardening,
- power—spacecraft/remote systems with solar or battery power have limited power,
- cooling—devices may require redesign for conduction cooling; relates to low-power,
- thermal—devices must operate over extended thermal ranges and temperature cycling,
- vibration—range/space equipment must survive launch and operational vibration,
- weight—launching or flying more weight costs more money,
- size—larger size results in more weight and associated launch issues,
- reliability—devices must be highly reliable, replacement or repair is often impossible.

These issues have limited the network hardware components currently available. The most common interfaces that have been used for onboard connectivity are MIL-STD-1553 and various serial interfaces. The 1553 technology was developed many years ago and significant investment was made to make it rugged, reliable, and radiation hard. However, 1553 technology is not well suited to network environments and it has limited data rates compared to modern network technologies.

Fig. 19 shows a sample architecture with an onboard LAN with multiple technologies and a WAN link to the ground.

If a remote system does not use an onboard LAN it can still use the WAN components to provide a single network interface to the system. Conversely it could also just have an onboard LAN and not use a network interface over the RF or optical link. However, using network technology both onboard and over the RF or optical link provides the maximum communication flexibility and enables full end-to-end communication.

The following sections discuss the current status of hardware and software network components for space as well as what is needed for future space missions.

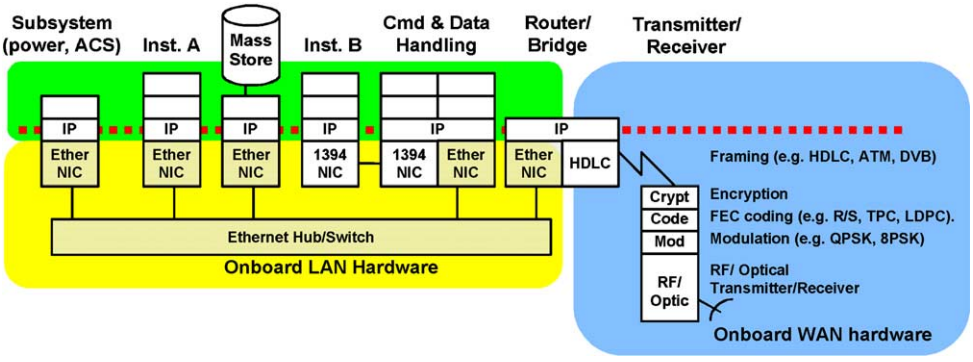

Fig. 19. Example space system architecture.

17.1. Status of onboard LAN hardware

This category covers the hardware components that are needed to provide onboard LAN capabilities. The primary components are the network interface cards (NICs), connectors, hubs, and switches that provide network communication capabilities among onboard components. It may also include standard network accessible mass storage systems and timing systems. These systems are included due to performance issues such as high-rate transfers to mass storage and precision timing needs of time servers.

The primary LAN technologies currently identified are Ethernet, Firewire, and ATM. These technologies all support standard IP networking and are widely available in standard ground COTS products. There has also been some work in developing rad-hard flight qualified components for all of these technologies.

Ethernet is currently the most promising of these LAN technologies due to its universal usage in ground LANs. It supports and ever increasing range of data rates including 10, 100, 1000, and 10,000 Mbps with currently available ground equipment. With increasing data rates and LAN switching technology, Ethernet is moving into environments with high reliability and deterministic response times such as factory floor monitoring and control systems.

Table 2 lists components needed for supporting onboard LAN functions along with notes on current activities developing or using these components. The information on status is not exhaustive but is an attempt to start developing a list of what is happening to fill in missing pieces.

17.2. Status of onboard WAN hardware

This category focuses on the components that move data between the LAN, the WAN link, and the rest of the network. This includes the device between the LAN and WAN as well as the components associated with the transmitter/receiver. Current transmitters/receivers are based on radio frequency (RF) technology but future technology is expected to include optical components at data rates up to 10 Gbps. However, both RF and optical technologies will need similar components for functions such as data link framing, encryption, forward-error-correction, and modulation. Significant challenges arise and hardware development is needed as data rates move to 10 Gbps.

With a large amount of ground network equipment already available with WAN interfaces,

Table 2
Onboard LAN component status

Component	Status
Space qualified Ethernet, Firewire, ATM connectors	Rugged Ethernet connectors for factory floor
Radiation-hard Ethernet 10/100/1000 network interface cards (NICs)	Spectrum Astro working on 10/100 Ethernet using rad-hard and qualified COTS components NASA/GSFC building rad-hard Ethernet FPGA
Rad-hard Firewire (IEEE 1394) NICs	Being developed for NPOESS mission
Rad-hard ATM NICs	Northrop Grumman (TRW) Astrolink design
Rad-hard Ethernet, ATM hubs, switches,	NASA/GSFC working on Ethernet switch Spectrum Astro working on hub Astrolink has onboard ATM switch
Device drivers for NICs in standard OS	Should be similar to standard NICs but may incorporate fault-tolerance features
Fault tolerant LAN equipment and failure recovery strategies	Factor/automation and process control community working on components/concepts
High-speed, network attached random access mass storage for file systems	Possible application of SAN and iSCSI network storage concepts
High stability, radiation-hard, time systems (clocks, network time servers)	Work needed on low-cost stable clocks and fast access mechanisms

developing matching range and space WAN components provides the simplest way to extend network capabilities. Current network spacecraft are using ISO standard, high-level data link control (HDLC) framing which is supported by standard synchronous serial interface chips and cards that are directly supported by standard ground routers. HDLC framing has been successfully used in satellite systems for over 20 years but only in low-earth orbits with limited radiation. The relatively low radiation levels have allowed the use of standard HDLC components but more radiation-hard components will be needed for future missions in more severe radiation environments. Also, HDLC interfaces currently only support rates up to about 100 Mbps on the ground and new interfaces will need to be developed for higher rates or other framing options will need to be identified. Some other candidates for higher rates are ATM, EIA IS-787, DVB, packet over SONET, and 10 Gbps Ethernet.

Moving data between the LAN and WAN can be done using various techniques. A basic approach is to have both a LAN and WAN interface attached to the primary command and data handling processor in the remote system and have software on the processor that routes data between the interfaces. However, the processor may be busy performing other functions and it may be better to perform the LAN/WAN transfers using a separate component. Some missions at GSFC are currently working on a LAN/WAN bridging component to offload this function from the main processor and also provide more fault tolerance by using multiple LANs and bridges. In order to reduce complexity this device will only support minimal bridging capabilities to move packets between the LAN and WAN interfaces and will not perform full IP routing.

Eventually, full function routers will be developed that will move data between the LAN and WAN while also providing additional higher level functions such as mobile routing protocols, security functions, traffic prioritization, and hot standby recovery. This will provide all networking functions in a standardized device and allow the onboard LAN to function identically to any ground LAN. Table 3 summarizes components needed to support the onboard WAN interface functions.

Finally, upper layer functions are addressed in Table 4. Security solutions based on Internet security protocols [47] (IPsec) and virtual private net-

Table 3
Onboard WAN component status

Component	Status
Link level encryption/decryption hardware	Some military components available
Radiation-hard, forward-error-correction hardware (e.g., Reed/Solomon, Low Density Parity Check, Turbo Product Code)	Reed/Solomon encoders available for space Work underway on TPC and LDPC
Radiation-hard framing hardware (e.g., HDLC, ATM, EIA IS-787, DVB, packet over SONET, 10 Gbps Ethernet)	COTS HDLC chips used on LEO spacecraft for over 20 years. Simple to implement in rad-hard FPGAs. Need higher rate rad-hard components for others
High-rate versions of coding, encryption, and framing hardware (up to 10 Gbps)	Possible solutions coming from DoD Transformational Communication project
Rad-hard Ethernet or ATM bridges to transmitter	GPM mission working on Ethernet/serial bridge
Rad-hard routers with Ethernet, Firewire, ATM, serial interfaces	General Dynamics (Motorola) and Cisco working on prototype rad-hard router/transceiver ITT adding routing features to LPT transceiver Spectrum Astro starting on router
Basic mobile IP routing protocols for single IP address mobility	Mobile IP available and flown on CANDOS More mobility solutions coming in IPv6
Mobile Routing to hide mobility details from and entire subnet of onboard systems	Cisco Mobile Routing being prepared for test-flight on SSTL DMC spacecraft in 2003

Table 4
Upper layer protocols/applications status

Component	Status
Security algorithm accelerator chips for supporting network and application level security	Ground based components available but rad-hard ones needed
Application level data encryption and key management	Many standard commercial solutions available but best solutions for range and space use need to be identified
Standard time stamping mechanisms (e.g., time-stamped PING, RTP) for IP packets	Options exist but may be different than legacy mechanisms New concepts built on Internet standards are needed
Time synchronization and clock management mechanisms (e.g., NTP)	NTP being used on so me spacecraft but more work needed on higher timing precision
Standard messaging protocols	GMSEC task at GSFC working with COTS messaging protocols to provide end-to-end software bus
UDP based highly reliable protocols for one-way links (e.g., FEC, MDP, NORM, Digital Fountain)	MDP successfully tested on CANDOS shuttle mission over one-way downlink Digital Fountain is an interesting option for highly reliable file; distribution over one-way links Work needed on memory efficient versions
UDP based reliable protocols for asymmetric, intermittent links (MDP, NORM, CFDP)	MDP being used for various unicast, multicast, and one-way link applications NORM being developed as IETF standard with modular building blocks for FEC CFDP being used on AISAT-1 spacecraft

works [48] (VPNs) will be needed along with the mobile IP environment.

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.

18. Conclusions

The OMNI project at NASA/GSFC has defined a flexible end-to-end spacecraft data communication architecture using Internet Protocols. The tests and demonstrations have 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 20 years. Using the Internet Protocol as a network layer allowed easy integration and testing of 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, causing 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 virtually any communication hardware that has a serial interface.

Using the IETF multiprotocol over frame relay encapsulation 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.

Introducing a network protocol like IP in the space 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 in COTS products.

Full deployment of Internet Protocols for spacecraft will require ground station upgrades and more system engineering to deploy Mobile IP and security solutions. However, these can all be addressed with commercially available products and solutions.

The major missing pieces are hardware components for the spacecraft. Technologies like Ethernet and HDLC are currently in use on some low-earth orbit spacecraft where radiation is not a major issue. More work is needed to develop fully space-qualified components for onboard serial interfaces to the RF equipment and for onboard LANs.

While many of the Internet transport 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 NTP 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.

Acknowledgments

The research described in this paper was carried out by personnel from Computer Sciences Corporation working for NASA's Goddard Space Flight Center's Code 588 under contract GS-35F-4381G S-36130-G, with additional efforts and support contributed by individuals from various GSFC organizations. The work was funded by NASA's Space Operations Management Office (SOMO) Communication Technology Project, the CANDOS mission, the GSFC Standards Program, and the Earth Science Technology Office. The authors would like to thank Dave Israel and GSFC code 450 for their pioneering IP work on the SPTR project and the CANDOS mission, Gary Meyers and James Rash of GSFC for their continual support of the OMNI project, Chris Jackson of Surrey Space Technologies Ltd. and Harold Price of VyTek Wireless for their support on UoSAT-12, and Cisco Systems for the loan of a Cisco 1601 router for use in the SSTL ground station.

References

[1] Internet Protocol, DARPA Internet Program Protocol Specification, Internet Engineering Task Force RFC-791, September 1981.
[2] G. Prescott S. Smith K. Moe, Real-time information technology challenges for NASA's Earth Science Enterprise, in: The 20th IEEE Real-Time Systems Symposium, December 1–3, 1999 Phoenix, AZ, USA.
[3] IP Mobility Support, Internet Engineering Task Force RFC-2002, October 1996.
[4] Internet Protocol, Version 6 (IPv6), Internet Engineering Task Force RFC-2460, December 1998.
[5] Cellular IP-A New Approach to Internet Host Mobility, ACM Computer Communication Review, January 1999.
[6] The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, Internet Engineering Task Force, Internet-Draft Mannet-DSR-03, 22 October 1999.
[7] Security Architecture for the Internet Protocol, Internet Engineering Task Force RFC-2401, September 1998.
[8] Information Technology Telecommunications and Information Exchange Between Systems High-level Data Link Control (HDLC) Procedures, International Organization For Standardization ISO/IEC 13239:2002, (Adopted ISO/IEC 13239:2002, third ed., 2002-07-15).
[9] Hypertext Transfer Protocol-HTTP/1.1. Internet Engineering Task Force RFC-2616, June 1999.
[10] NFS: Network File System Protocol Specification, Internet Engineering Task Force RFC-1094, March 1989.
[11] Telnet Protocol Specification, Internet Engineering Task Force RFC-854, May 1983.
[12] File Transfer Protocol, Internet Engineering Task Force RFC-959, October 1985.
[13] Transmission Control Protocol, Internet Engineering Task Force RFC-793, September 1981.
[14] User Datagram Protocol, Internet Engineering Task Force RFC-768, August 1980.
[15] Telemetry Channel Coding, Consultative Committee for Space Data Systems, CCSDS 101.0-B-4, May 1999.
[16] PPP in Frame Relay, Internet Engineering Task Force RFC-1973, June 1996.
[17] Multiprotocol Interconnect over Frame Relay, Internet Engineering Task Force RFC-2427, September 1998.

[18] PPP over SONET/SDH, Internet Engineering Task Force RFC-2615, June 1999.
[19] Router Information Protocol (RIP) Version 2, Internet Engineering Task Force RFC-2453, November 1998.
[20] Open Shortest Path First (OSPF) Version 2, Internet Engineering Task Force RFC-2328, April 1998.
[21] BGP4/IDRP for IPOSPF Interaction, Internet Engineering Task Force RFC-1745, December 1994.
[22] IP Header Compression, Internet Engineering Task Force RFC-2507, February 1999.
[23] Compressing IP/UDP/RTP Headers for Low-speed Serial Links, Internet Engineering Task Force RFC-2508, February 1999.
[24] Internet Engineering Task Force, RTP: A Transport Protocol for Real-Time Applications, RFC-1889, January 1996.
[25] A. Welch, D. Brooks, D. Beering, D. Hoder, M. Zernic, Experimental results of running TCP/IP over ATM on NASA ACTS HDR, NASA/GRC, 1997. Available from: <http://acts.qrc.nasa.gov/library/docs/qsn/welchpaper.pdf>.
[26] Internet Engineering Task Force, A Proposal to add Explicit Congestion Notification (ECN) to IP, RFC-2481, January 1999.
[27] Internet Engineering Task Force TCP Selective Acknowledgement Options, RFC-2018, April 1996.
[28] G. Morabito, I. Akyildiz, S. Palazzo, TCP Peach: A Transport Layer Protocol for Satellite IP Networks, Second International Workshop on Mobile and Wireless Communication Networks, Paris France, May 2000.
[29] H. Price, J. Ward, Pacsat Broadcast Protocol, ARRL 9th Computer Networking Conference, August 1990, pp. 232–238.
[30] C. Miller, StarBurst MFTP Compared to Today's File Transfer Protocols: A White Paper, StarBurst Communications Corporation, 1996, p. 34.
[31] Consultative Committee for Space Data Systems, CCSDS File Delivery Protocol (CFDP)-Part 1: Introduction and Overview, CCSDS 720.1-G-0.5, July 1999.
[32] Internet Engineering Task Force, NFS: Network File System Protocol Specification, RFC-1094, March 1989.
[33] Internet Engineering Task Force, The TFTP Protocol (Revision 2), RFC-1350, July 1992.
[34] Internet Engineering Task Force, Network Time Protocol(Version 3) Specification, Implementation and Analysis, RFC-1305, March 1992.
[35] Internet Engineering Task Force, File Transfer Protocol, RFC-959, October 1985.
[36] Internet Engineering Task Force, Hypertext Transfer Protocol-HTTP/1.1, RFC-2616, June 1999.
[37] Internet Engineering Task Force, Simple Mail Transfer Protocol, RFC-821, August 1982.
[38] J. Macker, R.B. Adamson, The Multicast Dissemination Protocol (MDP) Toolkit, IEEE, 1999. Available from: <http://mdp.pf.itd.nrl.navy.mil/MdpToolkitOverview.ps.gz>.
[39] Consultative Committee for Space Data Systems, Space Communications Protocol Specification (SCPS)—Rationale, Requirements, and Application Notes, CCSDS 710.0-G-0.3, April 1997.
[40] ITT Reference Data for Radio Engineers, Howard W. Sams & Co. Inc., ISBN 0-672-21218-8, 1975.
[41] Internet Engineering Task Force, IP Header Compression, RFC-2507, February 1999.
[42] Cellular IP—a new approach to Internet host mobility, ACM Computer Communication Review, January 1999.
[43] Internet Engineering Task Force, IP Mobility Support, RFC-2002, October 1996.
[44] J. Rash, K. Hogie, R. Parise, E, Criscuolo, F. Hallihan, T. Le, Demonstrations of Internet Protocols in space using TDRSS, presented at The First Joint Space Internet Workshop, NASA Goddard Space Flight Center, November 2000.
[45] Network Time Protocol (Version 3) Specification, Implementation and Analysis, Internet Engineering Task Force RFC-1305, March 1992.
[46] J. Rash, K. Hogie, R. Parise, E. Criscuolo, J. Langston, C. Jackson, H. Price, Results of 'Internet in space' tests using UoSat-12, presented at The First Joint Space Internet Workshop, NASA Goddard Space Flight Center, November 2000.
[47] Internet Engineering Task Force, Security Architecture for the Internet Protocol, RFC-2401, November 1998.
[48] Internet Engineering Task Force, Virtual Private Networks Identifier, RFC-2685, September 1999, p. 35.

Keith Hogie—Computer Sciences Corporation—has an extensive background in designing and building satellite data processing systems, control centers, and networks at GSFC. He has developed ground data processing systems and control centers for over 14 spacecraft over the last 25 years at NASA/GSFC, and led the development of the NASA Internetworking Laboratory Environment in 1990. He is the technical leader of the OMNI project at GSFC where he is applying his networking and satellite background to develop and demonstrate new communication technologies for future space missions.

Edward Criscuolo Jr. joined Computer Sciences Corp. in 1991 as a Senior Computer Scientist working for the Goddard Space Flight Center. In that time, he has been the task lead for a number of spacecraft ground system projects that span many aspects of Goddard space missions, including Planning and Scheduling systems, spacecraft command management, and level-0 processing of telemetry and science data. In 1999, he joined the OMNI project as a senior project member, where his duties include systems analysis, systems engineering, top-level design, prototype development, and backup technical lead.

Ron Parise—Computer Sciences Corporation—in 1984, was selected as a payload specialist astronaut and was involved in mission planning, simulator development, integration and test activities, flight procedure development, and scientific data analysis. He has logged 615 h in space as a member of the STS-35 and STS-67 crews. In 1996 he assumed a communications engineering support role for Mir, International Space Station (ISS), and the X-38 project. In 1997 he also began working with the OMNI project as a scientific liaison and systems architect.