CCSDS Time-Critical Onboard Networking Service

Steve Parkes[1]
Space Technology Center, University of Dundee, Scotland

Rick Schnurr[2], Jane Marquart[3], Greg Menke[4]
NASA/Goddard Space Flight Center, Greenbelt, MD, USA

and

Massimiliano Ciccone[5]
ESTEC, European Space Agency

The Consultative Committee for Space Data Systems (CCSDS) is developing recommendations for communication services onboard spacecraft. Today many different communication buses are used on spacecraft requiring software with the same basic functionality to be rewritten for each type of bus. This impacts on the application software resulting in custom software for almost every new mission. The Spacecraft Onboard Interface Services (SOIS) working group aims to provide a consistent interface to various onboard buses and sub-networks, enabling a common interface to the application software. The eventual goal is reusable software that can be easily ported to new missions and run on a range of onboard buses without substantial modification. The system engineer will then be able to select a bus based on its performance, power, etc and be confident that a particular choice of bus will not place excessive demands on software development. This paper describes the SOIS Intra-Networking Service which is designed to enable data transfer and multiplexing of a variety of internetworking protocols with a range of quality of service support, over underlying heterogeneous data links. The Intra-network service interface provides users with a common Quality of Service interface when transporting data across a variety of underlying data links. Supported Quality of Service (QoS) elements include: Priority, Resource Reservation and Retry/Redundancy. These three QoS elements combine and map into four TCONS services for onboard data communications: Best Effort, Assured, Reserved, and Guaranteed. Data to be transported is passed to the Intra-network service with a requested QoS. The requested QoS includes the type of service, priority and where appropriate, a channel identifier. The data is de-multiplexed, prioritized, and the required resources for transport are allocated. The data is then passed to the appropriate data link for transfer across the bus. The SOIS supported data links may inherently provide the quality of service support requested by the intra-network layer. In the case where the data link does not have the required level of support, the missing functionality is added by SOIS. As a result of this architecture, re-usable software applications can be designed and used across missions thereby promoting common mission operations. In addition, the protocol multiplexing function enables the blending of multiple onboard networks. This paper starts by giving an overview of the SOIS architecture in section II, illustrating where the TCONS services fit into the overall architecture. It then describes the quality of service approach adopted, in section III. The prototyping efforts that have been going on are introduced in section IV. Finally, in section V the current status of the CCSDS recommendations is summarized.

---

[1] Steve Parkes, Director Space Technology Centre, University of Dundee, member
[2] Richard Schnurr, AETD Chief Architect, NASA/GSFC, non-member
[3] Jane Marquart, Senior Software Systems Engineer, NASA/GSFC, Code 582, non-member
[4] Greg Menke, Senior Software Systems Engineer, NASA/GSFC, non-member
[5] Massimiliano Ciccone, Data Handling Software Engineer, ESA/ESTEC, non-member

I. Introduction

Today many different buses are used on Spacecraft requiring software with the same basic functionality to be rewritten for each type of bus. This then impacts on the application software resulting in custom software for almost every new mission. The Spacecraft Onboard Interface Services (SOIS) working group aims to provide a consistent interface to various onboard buses and sub-networks, enabling a common interface to the application software. The eventual goal is reusable software that can be easily ported to new missions and run on a range of onboard buses without substantial modification. The system engineer will then be able to select a bus based on its performance, power, etc and be confident that a particular choice of bus will not cause/place excessive demands on software development.

II. SOIS Architecture

This section presents an overview of the SOIS architecture describing how the Time Critical Onboard Network Services fit into the overall SOIS architecture.

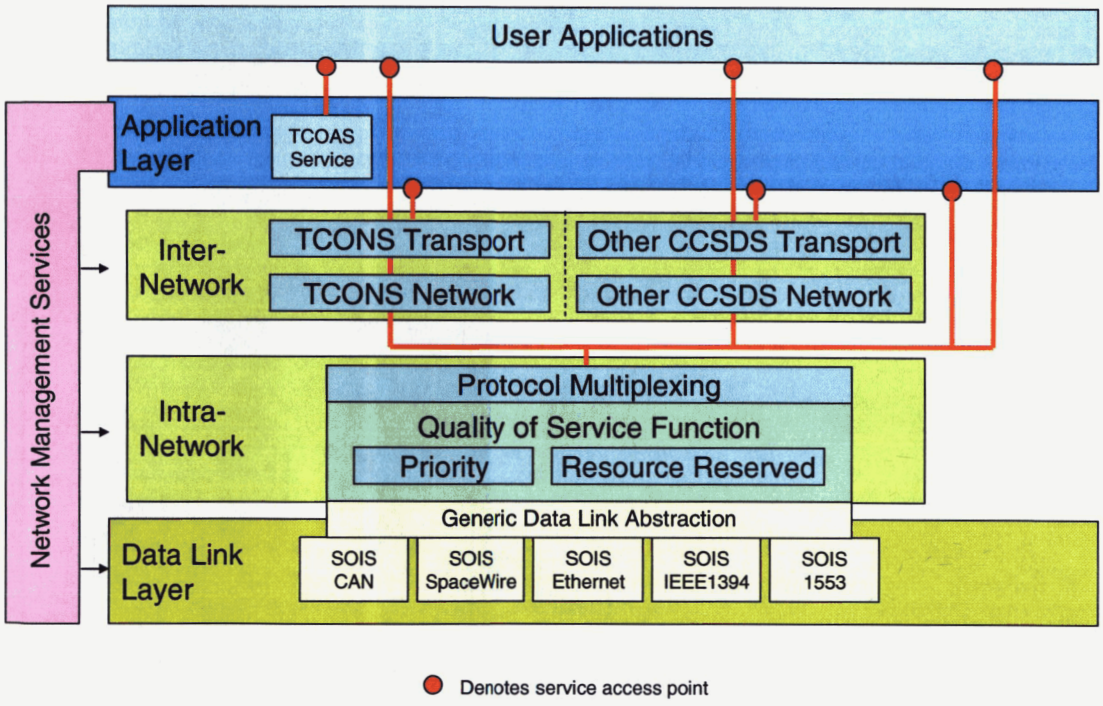

Figure 1. SOIS Architecture

The SOIS architecture, shown in Figure 1, defines a standard set of services for use onboard spacecraft. The services are categorized by a stack of protocol layers where each of the layers provides a specific set of functionality. User Applications use the SOIS services of the underlying layers. TCOAS (Time Critical Onboard Application Services) is a collection of common onboard application layer services that rely on the time-critical services of TCONS. The Inter-network layer provides transport and inter-networking services across multiple underlying sub-networks. Beside the protocols defined by TCONS, other CCSDS transport/network protocols are also permitted including TCP/IP protocols. The Intra-network layer provides protocol multiplexing and quality of service functions for TCOAS and the inter-networking protocols. QoS services provided by the intra-network are supported by functionality in the data link layer, such as redundancy and retry allocation. The Intra-Network layer provides a common interface to the various data link protocols and this will be the focus of this paper. The SOIS-OBL(Onboard LAN) represents the Data Link layer and contains the various data-links supported by SOIS. The Generic Data Link Abstraction is an abstraction of all the functions in the data link. For each data link, a common set of functions, represented by the Generic Data Link Abstraction block, is provided to the Intra-Network layer (see Figure 1). Network Management Services are used to configure, monitor and control the SOIS network.

III. Quality of Service

A key part of the Intra-Network Service functionality is the provision of Quality of Service for the protocols and applications using the service.

Quality of Service (QoS) is the ability of a communication system to provide differentiated services. Users send data by identifying the data to send and requesting a quality of service with which the data should be sent. Quality of Service for a communication service may be characterized in terms of important features relevant to that communications service, for example:

- Reliability
- Transmission rate
- Effective Bandwidth and latency
- Error rate

There are three elements to the TCONS QoS Model:

a) Priority – which corresponds to the priority function of the Intra-Network layer.

b) Resource Reservation (Resource Reserved / Non-Reserved) – which corresponds to the resource reserved function of the Intra-Network layer.

c) Reliability (Try Once / Retry) – which corresponds to the retry and redundancy functions of the OBL layer.

These three independent QoS elements are mapped into four intra-network services:

- Best Effort: Non-Reserved, Try Once
- Assured: Non-Reserved, Retry
- Reserved: Resource Reserved, Try Once
- Guaranteed: Resource Reserved, Retry

Each intra-network service has associated QoS parameters. All four types of service may be implemented in any combination over both asynchronous and scheduled systems. These four types of intra-network service will now be described in turn.

A. Best Effort (Non-Reserved, Best Effort)

The Best Effort QoS (Non-Reserved, Try Once) makes no promises about the time of delivery, the network bandwidth available or the error rate of the traffic. The error rate is the likelihood of there being an error in the traffic received at the destination (missing traffic). Traffic delivered containing errors will be detected and discarded. Only error free packets will be passed to the application receiving from TCONS. Traffic received is not acknowledged so the source does not know if the traffic arrived safely or not. There is no retry for traffic that failed to be delivered safely. The order of data sent is preserved with the exception of missing data. Data delivery is in-sequence, incomplete and without errors. "Incomplete" means that some data may be missing and "without error" means that only data that is received without error is passed up to the application, any data that arrives corrupted in any way is discarded.

Several priority levels are provided for Best Effort (and Assured) traffic. Traffic with a higher priority level is treated preferentially compared to traffic with a lower priority level. Best Effort QoS has the same set of priority levels as Assured QoS. This means that if Best Effort and Assured Service Data Units (SDU) or Protocol Data Units (PDU) are competing for a communication resource, the one with the highest priority will be chosen regardless of whether it is Best Effort or Assured.

Best Effort traffic classes use any remaining bandwidth left over from the resource reserved traffic classes.

B. Assured (Non-Reserved, Retry)

The Assured QoS (Non-Reserved, Retry) tries to ensure that the traffic arrives at the intended destination. If the data does not arrive safely at the destination then it is discarded and resent. To support this, the destination acknowledges the receipt of Assured traffic. The order of data sent is preserved. Data delivery is in-sequence, complete and without errors. The user will be told when completeness cannot be achieved.

The Best Effort and Assured services use the same set of priorities.

The QoS provided is statistical, ensuring that data will get through (wherever possible), and delivers higher priority traffic before lower priority traffic.

Assured traffic classes use any remaining bandwidth left over from the resource reserved traffic classes.

C. Reserved (Resource Reserved, Best Effort)

The Reserved QoS (Resource Reserved, try-once) reserves network resources for traffic. It is able to ensure the timeliness of delivery and the network bandwidth available, but makes no promises about the error rate of the traffic. Traffic may be lost but if it does arrive at the destination it will do so in a timely manner. Any data that arrives containing errors will be discarded. The order of data sent is preserved with the exception of missing data. The service is in-sequence and without errors.

Resources must be reserved, by setting up a channel, before communication can take place between a particular source and destination. Reserving network resources ensures communication bandwidth and timeliness. Channels must be predefined using the TCONS network management service. They are not set up through the intra-network service interface.

Several priority levels are provided for Reserved traffic. Reserved traffic with a higher priority level is treated preferentially compared to traffic with a lower priority level. Since Reserved traffic has reserved bandwidth (or transmission slot) there is no need for it to compete with the other types of traffic. This means that priority for Reserved traffic is used only when queuing traffic for a particular reserved resource. I.e. if two SDUs or PDUs are to be serviced, both with the same Reserved channel (using the same bandwidth or transmission slot), the one with the highest priority will be serviced first.

D. Guaranteed (Resource Reserved, Retry)

The Guaranteed QoS (Resource Reserved, Retry) reserves network resources for traffic and provides for repeated communication attempts in the event of an error occurring. It is able to ensure the time of delivery, the network bandwidth available, and ensures error-free data The order of data sent is preserved. The service is in-sequence, complete and without errors. The user will be told when completeness cannot be achieved.

Resources must be reserved before communication can take place between a particular source and destination. Reserving network resources ensures communication bandwidth and timeliness.

Several priority levels are provided for Guaranteed traffic. Since Guaranteed traffic is using reserved resources its priority only has meaning within that class of traffic, as for the Reserved traffic.

E. Resource reservation

TCONS QoS provides a consistent approach to resource reservation independent of whether it is implemented, for example, using time division multiplexing in a scheduled system or bandwidth reservation in an asynchronous system.

Resource reservation is defined using a cyclic time period known as an epoch which is sub-divided into a number of smaller time slots. In a scheduled system a time-slot is the slot in which data is sent. In a bandwidth reserved system the time-slot is the interval of time over which bandwidth usage is measured and controlled i.e. a time-slot is the sampling interval at which bandwidth utilization is re-evaluated and adjusted.

Each reservation of resources is given an identifier referred to as a channel number. The available resources in each slot are allocated to one or more channels, with each channel allocated a portion of the available resources. No more than 100% of the available resources must be allocated in any one slot.

Best Effort and Assured SDUs are sent in bandwidth which is either not reserved or which has been specifically reserved for these two types of traffic.

On some sub-networks there may be multiple, independent, non-conflicting sets of resource or paths through the sub-network. In this case the percentage utilization of each of these independent sets of resources must not be more

than 100%. The aggregate resource utilization of all of the independent paths may then be more than 100%. No single resource in a sub-network can be utilized more than 100%.

IV. Prototyping Efforts

The focus of the prototyping efforts was to show that the architecture could indeed support multiple underlying physical busses without incurring large rewrites of software applications. Critical to this work was the development of an appropriate Application Programming Interface (API) which captured the functionality, or at least the interface to the functionality, of the TCONS services. Once this API had been defined and agreed in principal between the TCONS working group members, it was possible to start on the development of various prototypes. TCONS Intra-Network Service prototypes are being developed by NASA GSFC and University of Dundee. The NASA prototype is described below.

F. GSFC Prototype, TCONS over Ethernet

The first TCONS prototype was developed in 2004 by GSFC in conjunction with a risk mitigation effort for an onboard IP network over Ethernet. At this time, the TCONS architecture was in its infancy and some trade studies were done to prove out the most basic assumptions. The first question was whether or not an existing protocol would satisfy the onboard timing requirements of packet passing between distributed subsystems on a single onboard network. Since this was an IP network, TCP and UDP were examined. While TCP is a reliable transport, the time variation in acknowledgment is not deterministic. In addition, if a TCP port goes down, the restart time may be too long. UDP is a better fit, but requires reliability/redundancy be implemented by the user.

The most significant trade, however, was determining where to implement reliability, either at the data link layer or at the application layer. The simplest and most straightforward method of implementing reliability is at the application layer, so a custom software implementation known as the Fault Tolerance Messaging Service (FT-MSG) was written as an application. This service added a conventional ack/retry algorithm (Assured service) on top of UDP. While the application provided the needed services, and was highly portable using standard socket api, other TCP/IP traffic, and the IP stack itself, added unpredictable delay to any given message. Measuring rx & tx packet latency through the stack showed the IP stack's timing overhead per packet, regardless of direction, is highly variable and non-deterministic out to several milliseconds on a MCP750 @ 233 MHz. This was unacceptable, thus reliability had to be implemented at the data link layer. As a result, an ack/retry algorithm was implemented within the NIC driver. The driver accepts all data units bound for the NIC, from the IP stack or others, and manages acks/retries invisibly to traffic. Latency & delay are well defined since control is in place right at the hardware. Reliability has been implemented using the IEEE802.3 Logical Link Control standard, allowing interoperation over conventional Ethernet. Note, however, that this protocol is only useable between conformant drivers.

Prototyping continued on the flight testbed for timing and latency analysis using best-effort and assured services. No issues were found. The final set of tests was simple characterization to ensure that LLC and isochrony (scheduled service) were being invoked correctly.

The test configuration consisted of two MCP750 PowerPC chasses running Linux, each with one Flight NIC. Since the LLC and Scheduled service tests were performed using UDP, each NIC was given its own IP address. The two NICs were connected to an otherwise idle Cisco Ethernet switch, which facilitated the use of conventional ethernet monitoring tools.

The test software was configured to send a predefined number of fixed size packets (each approximately 100 bytes), parameterized such that the NIC drivers in both test systems would direct the packet flows through the logic under test. While sending, the test software monitored the output transmit queues in the driver, keeping each one being used approximately half full. This kept the transmit bandwidth of the NIC fully occupied but prevented transmit contention.

Test results were observed by comparing packet counters exposed by the NIC hardware and driver with the packet counts transmitted by the test software. If the counts were equal, all packets transmitted were received. For tests which involved programmed packet loss, the test packet counts were confirmed by ensuring the correct numbers of packets were dropped by the correct portions of the driver.

- Unidirectional LLC

  For each run, 20000 packets were sent from host A to host B, with only LLC acks returning. This test was run several times; without packet loss, with loss of transmit packet, with loss of ack packet and with no connectivity to the receiver host. The runs that included packet loss exercised all states of the LLC state machines in both the sender and receiver. This test demonstrated the nominal operation of LLC between two hosts.

- Bi-directional LLC
  For each run, 20000 packets were sent from host A to host B. While the test from A to B was in progress, the same test routine was started on host B, sending to host A. Under this test, LLC acks return in both directions for their respective packet flows. These tests were run with the same breakdown of programmed packet loss. This test demonstrated that multiple LLC packet flows between two hosts operate independently.

- Scheduled Service (Isochronous traffic)
  A schedule table was configured for both hosts, as follows. The cycle length was 500ms with 10ms quiescent intervals at the start and end of each cycle. The cycle was divided into 4 isochronous slots of the respective durations; 10ms, 15ms, 20ms, 25ms- each slot was also configured to transmit no more than 1 packet per cycle. All remaining time in the cycle was reserved for asynchronous traffic, which was not transmitted during the test. The varying slot duration allowed the performance monitoring subsystem to measure the timing unambiguously. In a flight scenario, all slots would very likely be of equal duration to ease the scheduling analysis.

The data unit size was increased to 500 bytes, matching previous performance monitoring tests, and programmed packet loss was discontinued. The test software activated the schedule table and began transmitting packets at different rates for each of the four slots, yielding a periodically varying mix of busy and idle slots in each network cycle. When the performance monitoring subsystem sample buffer filled, the traffic generation and sample logging was stopped. The performance data was then retrieved by FTP, the packet flows for which were directed through the asynchronous slot of the schedule. The reduction of bandwidth consumption in unit time during this transfer proved the driver limited the traffic flow according to the schedule table. The sample data was then post-processed and plotted. A random cycle was selected and the actual timing and event sequences checked against the test configuration.

G. GSFC TCONS over SpaceWire

The second prototyping effort done at GSFC was TCONS over SpaceWire. At this point, the TCONS architecture was more mature so updates to the Ethernet prototype software were made to reflect the changes, however, the basic structure of the TCONS architecture remained intact. The Ethernet driver was replaced with a SpaceWire driver. LLC was retained as the supporting reliability protocol. Basic testing of the architecture was accomplished, however, complete testing of the SpaceWire network has not yet been achieved due to hardware and funding issues. Enough has been tested, though, to verify the goal that reusable software can be easily ported to new missions and run on a range of onboard buses without substantial modification. Testing should be complete sometime in 2006.

H. University of Dundee TCONS over SpaceWire

The University of Dundee has implemented TCP/IP running over SpaceWire as an early prototype of potential TCONS protocols while the TCONS concepts were being developed [REF]. This work showed that transport and network layer protocols can be carried efficiently over SpaceWire. The University of Dundee will start work on a prototype of the proposed SOIS Intra Networking services during summer 2006. This work aims to prove the key principals of the proposed standard and will be completed by summer 2007. SpaceWire will be used as the target onboard network with the intention of proving the mapping of the TCONS services to SpaceWire.

I. ESA-ESTEC TCONS over 1553 and CAN

The object of this prototyping activity is to develop an efficient implementation of the CCSDS SOIS communications stack for use in the Virtual Spacecraft Reference Facility (VSRF).

Through this work a communication stack that implements the CCSDS recommendations for the SOIS communication services is developed and tested. The implementation is performed on a Linux platform and, after stand-alone testing; the stack is integrated with the VSRF in ESTEC and with Hardware-in-the-Loop, namely an ERC32-based SVF with representative flight software running under RTEMS.

While the target platform for this implementation is a PC running Linux, the chosen design results in an extremely efficient implementation that obeys the coding rules normally applied to flight software developers, and meets typical onboard flight processor and memory constraints.

The software has been designed so that it can be readily ported onto other platforms, particularly onto an ERC-32 flight processor running the RTEMS operating system. All sections of the code where environment specific modifications are necessary have been isolated into separate compilation units and are clearly indicated.

V. Conclusion

The TCONS working group has provided an architecture encompassing the Intra-Network service, Inter-Network service, data-link layer and network management. Central to this is the quality or service approach which has been defined and documented. A draft Intra-Network service interface document has been produced and a document describing the API has been written. Prototyping work has been done by NASA GSFC, ESA-ESTEC and University of Dundee. Currently effort is concentrated on producing a mapping of the Intra-Network service for SpaceWire, Ethernet, CAN and Mil-Std 1553.

The SOIS work aims to provide a consistent set of onboard communication services that can be used over various buses and networks. The goal is to provide software independence from the underlying communication hardware. The initial results are promising, showing that this can be done without the excessive processing power or memory costs that might be expected.

Acknowledgments

The authors thank the members of the TCONS working group for their contributions to the work on the SOIS standards. S.M Parkes thanks European Space Agency (ESA) and British National Space Centre (BNSC) for financial support for the work done by University of Dundee within the TCONS working group.

References

Mills, S, and Parkes, S M, "TCP/IP over SpaceWire" Data Systems In Aerospace 2003,

Parkes, S M, "SpaceWire: Links, Nodes, Routers and Networks", European Cooperation for Space Standardization standard number ECSS-E50-12A, January 2003.