The Use of the CCSDS File Delivery Protocol on MESSENGER

Christopher J. Krupiarz[1]
Scott C. Burleigh[2]
Constantine M. Frangos[1]
Brian K. Heggestad[1]
Douglas B. Holland[1]
Kevin M. Lyons[1]
William C. Stratton[1]

[1]The Johns Hopkins University Applied Physics Laboratory, Laurel, MD 20723-6099
Email: Christopher.Krupiarz@jhuapl.edu
[2]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109-8099
Email: Scott.Burleigh@jpl.nasa.gov

Abstract

MErcury, Surface, Space ENvironment, GEochemistry, and Ranging (MESSENGER) is a NASA Discovery mission to study the planet Mercury. It will be launched in March 2004 and will include two flybys each of Venus and Mercury, followed by Mercury orbit insertion in April 2009 for a one-year science-gathering mission. Through both phases of the mission, MESSENGER will collect data from seven instruments on key characteristics of the planet to further understand Mercury and the formation of the inner solar system. As part of the system software developed by The Johns Hopkins University Applied Physics Laboratory (JHU/APL), MESSENGER will employ a standard developed by the Consultative Committee for Space Data Systems (CCSDS) called the CCSDS File Delivery Protocol (CFDP). CFDP is used to telemeter instrument and spacecraft housekeeping and science data stored on a spacecraft's onboard file system to the ground system, and to transfer files from the ground to the spacecraft. CFDP was chosen for MESSENGER to allow for the reliable and direct transmission of files from the spacecraft to the ground. CFDP also complements the concept of a spacecraft file system, streamlining the gathering of data from instruments and the spacecraft.

This paper describes how CFDP will be used on the MESSENGER mission. To provide a framework for discussion, an overview of CFDP is given followed by a description of the CFDP software. The flight software is a JHU/APL-designed implementation of CFDP, while the ground software was licensed from the Jet Propulsion Laboratory (JPL). The JPL deliverable, which will also be used on the upcoming Deep Impact mission, includes both the core CFDP functionality and a test harness for verifying operability between the flight and ground. An explanation of both architectures is shown along with a review of the integration of the JPL package into the ground software. Finally, an overview of the flight-to-ground file storage process is described to show how CFDP fits into the larger mission picture.

Introduction

MErcury, Surface, Space ENvironment, GEochemistry, and Ranging (MESSENGER) will orbit Mercury in 2009 following two reconnaissance flybys each of Venus and Mercury. MESSENGER will investigate key scientific questions regarding Mercury's characteristics and environment [1]. Data are provided by an optimized set of miniaturized space instruments [2] and the spacecraft telecommunications system [3].

As part of the collection of this information, MESSENGER will be using an onboard file system. Data from these files will be telemetered following the standard developed by the Consultative Committee for Space Data Systems (CCSDS) called the CCSDS File Delivery Protocol (CFDP) [4]. CFDP was recently developed for file transfer across interplanetary distances. CFDP is an FTP-like file transfer protocol that

is optimized for the challenging communications properties of spacecraft that are operating outside of geosynchronous orbit with long link delays (> 10 s) and asymmetric bandwidth (> 100/1 down/up).

This paper discusses the planned usage of CFDP on MESSENGER. It provides an overview of CFDP, how it fits into the MESSENGER system design, and the modules that constitute the CFDP process on MESSENGER. These processes include the flight software implementation developed by The Johns Hopkins University Applied Physics Laboratory (JHU/APL), the ground implementation delivered by the Jet Propulsion Laboratory (JPL), the integration of the JPL software into JHU/APL's ground software, and the overall flow of files from the spacecraft to the ground.

The CCSDS File Delivery Protocol

With the increase of data storage onboard a spacecraft, it has become desirable to forego the traditional method of downlinking data in large blocks and, instead, adapt a method of using a file system. This not only reduces the complexity of storing the data, but it also clarifies what data are available. Data analysis is also made easier by being able to match the file system on the ground to that on the flight platform.

To promote standardization across missions, CFDP was recently developed for this type of application. It provides a standard set of capabilities for the transfer of files from one filestore to another. It can be used in a variety of configurations, from systems of one spacecraft (or *entity*, in CFDP nomenclature) to configurations of multiple entities, such as a lander and orbiter or a constellation of spacecraft. Ground control centers can also function as CFDP entities. Files can be transferred both to and from an entity and can also be manipulated by the full complement of file system commands (delete, move, copy, etc.). CFDP assumes that an underlying Unitdata Transport (UT) layer exists and, therefore, it is independent of how the data are physically exchanged between the two entities.

CFDP provides a framework for extensive interaction with the onboard file system. The complete protocol will not be discussed here, but in order to gain a general understanding of how the protocol works, it is useful to describe a typical CFDP transaction. A thorough discussion of CFDP is available in the CCSDS File Delivery Protocol Recommendation. Design suggestions are provided in the CFDP Implementers Guide [5].

To begin this example, assume there is a file onboard the spacecraft that is to be downlinked. In response to an on-board command from a user application, a File Data Unit (FDU) is created. This FDU consists of Protocol Data Units (PDUs) that contain information about the file, called Metadata PDUs, as well as the actual pieces of the file, or File Data PDUs. The file is now ready to be downlinked.

Two types of retransmission strategies are provided in CFDP. These are (1) Unacknowledged Mode, where the data are sent but no feedback is provided as to whether the file was received, and (2) Acknowledged Mode, where transmission is guaranteed through a series of positive acknowledgements (ACKs), negative acknowledgements (NAKs), and timers. For the following example, it can be assumed that the Acknowledged Mode has been selected. As the telemetering proceeds, PDUs are sent to the receiving entity. The PDUs can then be processed to determine whether gaps in the data exist. CFDP provides for four different ways to provide a NAK back to sending entity to indicate a retransmission is required. These are:

- *Immediate NAK*--a NAK is sent immediately upon the receiving entity noticing the missing data.
- *Deferred NAK*--NAKs are sent only upon receipt of an indication that the sending side has sent all of its data. This is done upon receipt of an End-of-File (EOF) PDU.
- *Asynchronous NAK*--a NAK is sent when instructed by a source outside of CFDP.

- *Prompted NAK*--the sending entity prompts the receiver to send NAKs.

Assuming that the Deferred NAK mode has been chosen for this example, any missing segments of data are listed for later notification to the flight side that data are missing.

The flight entity continues to send PDUs until the data from the file are exhausted. At this point, an EOF PDU is transmitted to the receiving entity. To ensure that the ground has received the EOF, the flight software starts a timer. This timer depends upon the one-way light time from the spacecraft to the ground and will expire if the ground does not respond to the EOF PDU by sending an EOF ACK.

Upon receipt of the EOF, the ground software responds with a NAK for any data it did not receive. At the sending of a NAK, a timer within the receiving software is started to ensure the flight software receives the NAK. When this timer expires, the NAK is resent if there are still file data segments that have not yet been received.

When the file is determined to be complete through receipt of all File Data PDUs and an EOF PDU, it is delivered to the destination user-application ground software, and a Finished PDU is sent by the ground entity. This indicates to the flight software that the file has been processed by the receiving entity. Again, a timer is started in the ground software, and it is stopped upon receipt of an ACK for the Finished PDU. When the ACK has been received, the transaction is complete.

CFDP on MESSENGER

The MESSENGER system consists of two CFDP entities: (1) the spacecraft, which contains the housekeeping and science data files to be downlinked, and (2) the ground system. The choices made for using CFDP on MESSENGER match the example given in the previous section. MESSENGER will use Acknowledged Mode to ensure receipt of the files and will use Deferred NAKs because of the large one way light time between Mercury and the Earth. MESSENGER's use of CFDP will also be simplified in several ways:

- *File transfer*—files will be transferred only one way, from the spacecraft to the ground.
- *File system operations*—file system operations, such as move and delete, will be performed via native flight software commands as opposed to within CFDP.
- *Initiation of file transfers*—file transfers on the flight side will be initiated automatically or through commanding outside of the CFDP framework. Although CFDP provides the capability to initiate these transactions from the ground through a proxy service, for MESSENGER file transfers from the ground will be initiated via native flight software commands that will instruct the CFDP flight software to downlink files.

These deployment decisions have been made to simplify the flight software, thus limiting CFDP's impact on MESSENGER margins for CPU and memory usage.

As discussed earlier, the UT layer of CFDP is implementation specific. For MESSENGER, this layer will consist of CCSDS telecommand packets on the uplink side and transfer frames for downlinking information from the spacecraft. PDUs that are issued from the ground system will be placed into the data portion of CCSDS command packets intended for the spacecraft. Once the spacecraft receives these commands, the PDUs are extracted from the packets and routed to the CFDP flight software task. The format of the uplinked PDUs is shown in Figure 1.

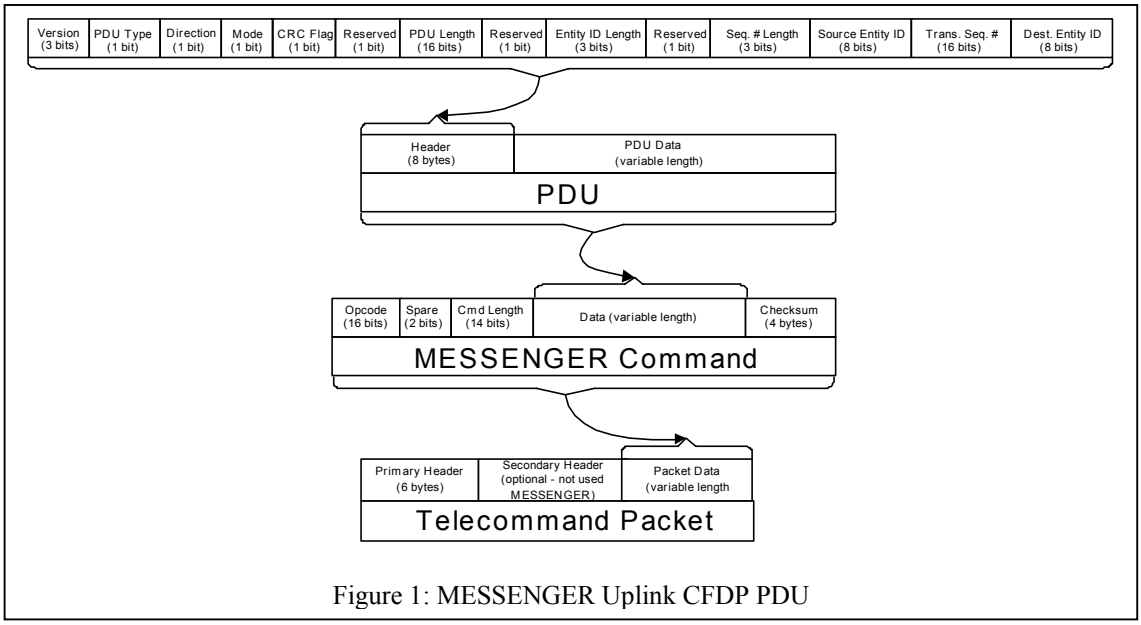

Figure 1: MESSENGER Uplink CFDP PDU

Unlike uplinked PDUs, the downlinked PDUs shown in Figure 2 will not reside within CCSDS packets. For MESSENGER, a decision was made to save downlink bandwidth by placing PDUs directly into downlink transfer frames. This is the same way that MESSENGER's CCSDS packets are placed into real-time transfer frames. The "first header pointer" in the transfer frame header to points to the first PDU header in a transfer frame. The PDU data length field can then be used to determine the location of the next PDU in the transfer frame. PDUs will be in variable lengths (like telemetry packets), so it can't be assumed that a transfer frame will contain an integral number of PDUs.

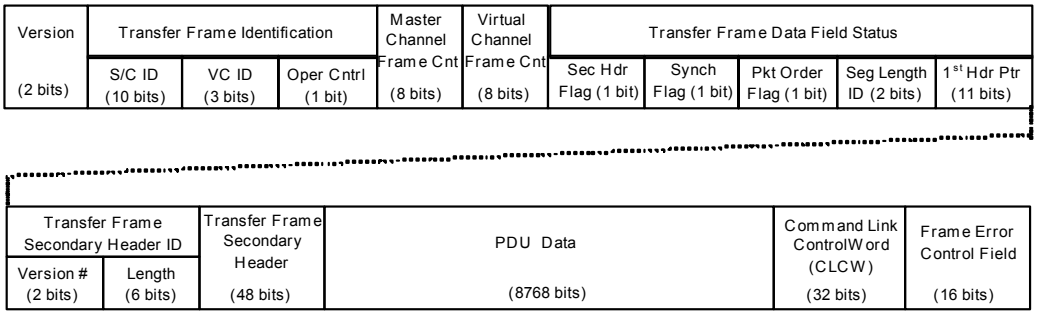

Figure 2: MESSENGER Playback Transfer Frame Format

MESSENGER's CFDP Flight Software

Although CFDP implementations for flight software exist, resource constraints drove the decision for JHU/APL to develop an in-house version of CFDP. This implementation, shown in Figure 3, was developed with the intent to include only that functionality necessary for MESSENGER. For instance, since files will only be downlinked, never uplinked, it was possible to simplify the design because no file

reconstruction will be necessary. Other departures from the full implementation include supporting only Acknowledged Mode and omitting the CFDP file system operations. Due to these limitations, the design is not fully CFDP compliant, but it does follow the CFDP Recommendation regarding the portions of the

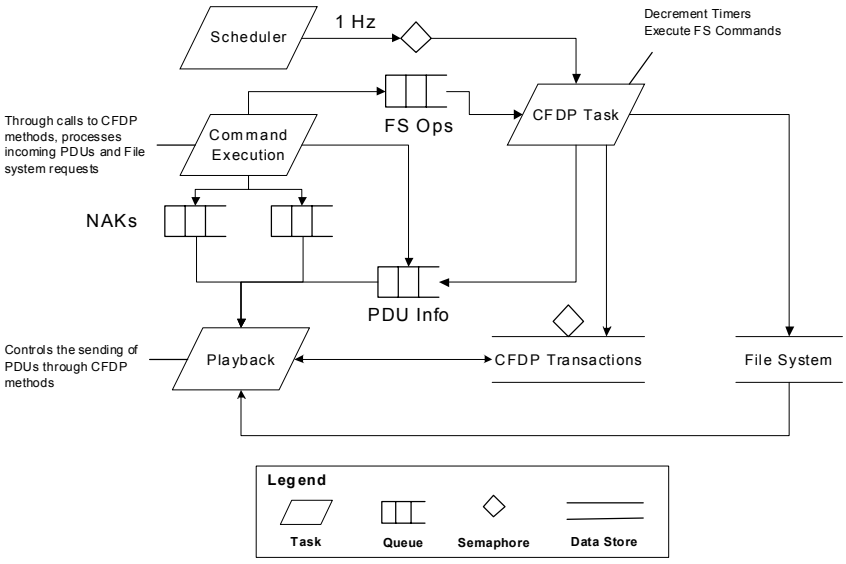

Figure 3: MESSENGER CFDP Flight Software

standard that it implements.

The MESSENGER flight software implementation consists of a primary store of information called the transaction table, and two processing components, a 1-Hz CFDP task and CFDP methods. The 1-Hz task operates independently, while the methods are called by two other tasks within the MESSENGER flight software architecture. Two priority levels are available to ensure that urgent files are downlinked as soon as possible.

The transaction table stores information on all of the current transactions. Information stored includes the file name, the Transaction ID, and an identification of what portion of the file remains to be downlinked. Given the distance to Mercury, it is estimated that at any one time, two hundred transactions may be open. This size is a hard limit set at software startup. These open transactions are defined as those currently being transferred or those awaiting information from the ground in the form of NAKs, EOF ACKs, or Finished PDUs.

The 1-Hz CFDP task is responsible for file system operations and for decrementing the timers for EOF PDUs. Within the MESSENGER flight software architecture, there is a Scheduler Task that starts various processes on the spacecraft at fixed rates. The Scheduler task starts the CFDP task once per second. The CFDP task first decrements the EOF ACK timers. When it determines that a transaction has timed out, it builds another EOF PDU and places it on the outbound PDU queue for future processing. It then checks whether any file system commands are waiting to be executed. If so, it then executes these commands.

The methods or function calls are used by two separate tasks within the flight software: (1) Command Exec and (2) Playback. As described earlier, uplinked PDUs arrive at MESSENGER via telecommands. After verification, these commands are executed by the Command Exec task. This task routes the PDUs to CFDP via an execute method that determines the type of PDU and proceeds accordingly. The

following describes the PDUs, the flight software supports, and the processing that results upon their reception:

- *NAK*-- Because a single NAK can contain multiple retransmission requests, the NAK is divided into potentially several entries in the appropriate NAK priority queue. The information contained in each entry is the Transaction ID and the start and stop offsets for the NAK. These are processed later by the Playback Task through a method call.
- *EOF ACK*--The timer is stopped in the transaction table for the Transaction ID in the ACK.
- *Finished*--A Finished ACK is placed on the outbound PDU queue for this transaction.
- *Suspend*--The transaction is marked as suspended in the transaction table. If this is the transaction currently being processed, a new transaction is selected.
- *Resume*--The transaction is marked as in progress in the transaction table. The transaction may become the current transaction depending upon the priority.
- *Cancel*--The transaction is removed from the transaction table. If this is the transaction currently being processed, a new transaction is selected.

On MESSENGER, the Playback Task is the primary user of CFDP. All Flight CFDP data is provided to the downlink telemetry stream through the Playback Task. Additionally, the Playback Task is generally the source of requests to CFDP to downlink files from the on-board file system. This task uses a predefined algorithm to determine downlink priority of files. Based upon this algorithm, it submits requests to CFDP for the transmission of files each time the previously requested file has been downlinked. CFDP adds the requested file to the transaction table. As space becomes available in the downlink buffers, the Playback Task requests PDUs from CFDP. The CFDP software selects a PDU from one of several sources. EOF and Finished ACKs in the outbound PDU queue are given priority. If none are available, the NAK queue is checked for retransmission requests. If there are no NAKs to process, CFDP will provide the next file segment from the file currently being processed.

MESSENGER's CFDP Ground Software

For the ground system, JPL is providing the core implementation of CFDP. This software, which is also to be used for both flight and ground software on the Deep Impact mission, is fully CFDP Blue Book compliant. The system consists of the following tasks:

Fdpd—transmission and retransmission daemon for a single CFDP entity
Fdpi—receives PDUs from the UT layer
Fpdo—sends PDUs to the UT layer
Fdpq—notifies fdpd when a PDU requiring an ACK has been sent, causing a timer to be started

Also included in the implementation are two libraries, libfdp and libfdput, that can be used to simplify the development of UT layers and user applications. User applications use libfdp functions to submit service requests to, and receive service indications from, fdpd. UT layer implementations use libfdput functions to obtain outbound PDUs from fdpo and deliver inbound PDUs to fdpi. fdpd processes all service requests, data arrivals, and timer expirations.

The JPL implementation of CFDP was designed to serve as a single, stable system that could be used without change in any number of missions, both in ground systems and, where appropriate, in flight software, to help reduce mission life-cycle costs. All code is written in C. Interfaces to file systems, operating system functions, underlying communication functions, and user applications are cleanly defined in small adaptation modules, enabling the core software to be used unmodified in a wide variety of operating systems (currently VxWorks, Solaris, and Linux), communication environments, and mission

configurations. Interaction among the functional elements of the implementation is completely event-driven, minimizing CPU resource consumption. Memory is managed privately, both for improved performance and to assure a hard upper bound on memory consumption. A non-volatile storage management system insulates CFDP from differences among mass storage implementations and enables CFDP to continue operating reliably even across an unplanned power reset.

Another benefit of the JPL delivery is that it provides a test harness for verifying the software. This wraps around the core software and acts as the UT layer through the use of User Datagram Protocol (UDP). It also provides the capability to simulate downlink and uplink rates as well as the dropping of information between the two entities. This enables the testing of the flight and ground software prior to the full commanding and telemetry infrastructure being available.

CFDP will be integrated into the Ground System using existing interfaces that provide CCSDS telemetry, input Telecommand packets, and control and monitoring of Ground Support Equipment. Figure 4 shows these planned interfaces to CFDP and the flow of data files to Mission Operations Center (MOC) and Science Operations Center (SOC) users. For MESSENGER, CFDP is the source of all spacecraft data from the spacecraft's Solid State Recorder. The spacecraft data files are received from the Deep Space Mission System (DSMS) as CCSDS transfer frames containing PDUs. The Transfer Frame Router will split the virtual Channel carrying CFDP Transfer Frames from the telemetry stream destined for the MOC, extract PDU headers and data, and deliver complete PDUs to the CFDP process. To ensure reliable delivery, the IP Operational Network (IONET) Router will buffer PDUs, as necessary, to absorb CFDP processing spikes.

CFDP manages downlink and reconstruction of the files from the PDUs. Completed files are moved to a long-term file repository, where they are copied to the SOC. With the exception of imager files, all spacecraft files consist of CCSDS packets. The ground system opens these files and ingests the packets into a combined archive of packets maintained by the MOC.

A goal of the system is to be able to provide some form of partial data file during the period when the CFDP transaction is not complete. The MESSENGER project plans to have only limited partial file accessibility; access to files that are largely complete but will remain incomplete for many hours until the next spacecraft contact will be allowed. The ground system also has a related requirement to be able to save the state of CFDP to allow for the CFDP process to be paused or stopped, to avoid requiring substantial retransmission. This requirement will enable the supporting system to shut down for routine or emergency reasons.

The CFDP process will be primarily controlled through the Integral Systems Inc., EPOCH 2000 Command and Telemetry system. MOC personnel will enter directives controlling the process through an interface supported by the system for the control of Ground Support Equipment (GSE). As described previously, overall control of the CFDP and the file system will require both commands sent to the

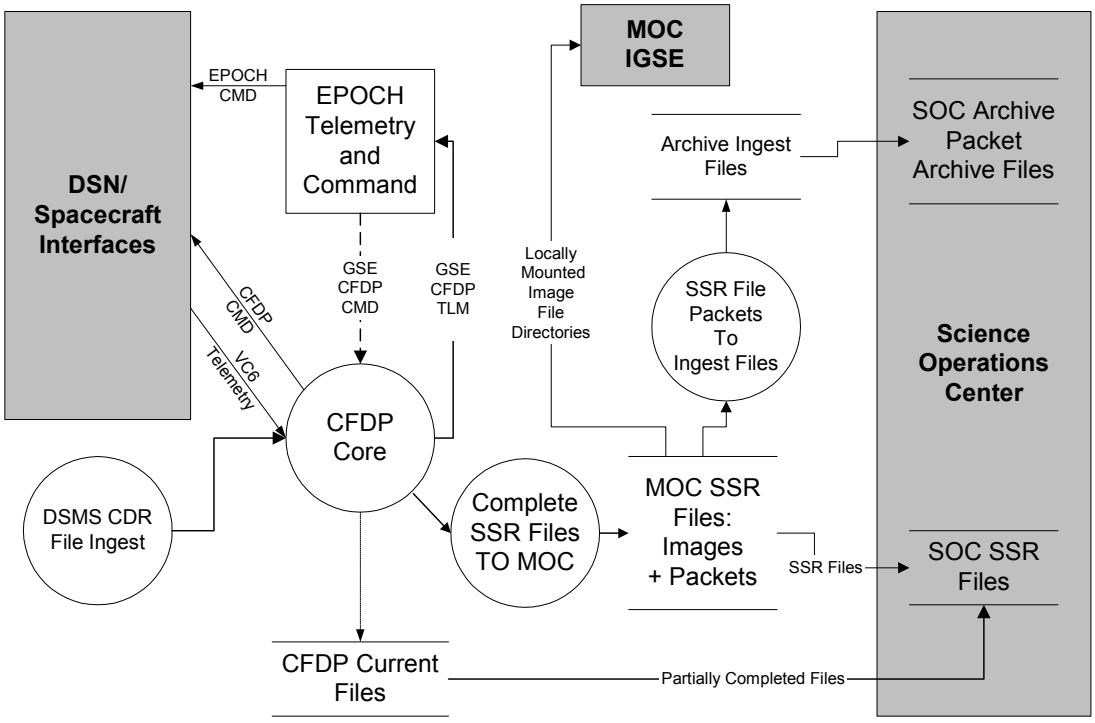

Figure 4: MESSENGER Ground Software Configuration

spacecraft through the normal command interfaces and also directives sent through the GSE interface to the ground CFDP process. CFDP status information required by the operator will be packaged as ground source telemetry packets and routed to operator displays. Because guaranteed delivery by CFDP is planned, transactions will need to be monitored to prevent system errors from producing unnecessary downlink. The CFDP process will maintain detailed diagnostic information in a log file that also contains all of the status information sent to the operators. This combined log will enable diagnostic analysis based on operator descriptions of problems.

The MESSENGER mission will receive a stream of data in near real-time from the DSMS at JPL. Because the real-time data service is not guaranteed to be loss free, the DSMS has a file-based Central Data Recorder delivery service that will be used to recover data in cases where ground loss is suspected. The CFDP process will be able to take input telemetry from the CDR files during periods when no downlinked data are available.

Summary

With its use of an on-board file system, MESSENGER is a prime candidate for the use of CCSDS File Delivery Protocol. CFDP streamlines the process of downlinking of files by providing a standard for communicating between the spacecraft and the ground. This allows the MESSENGER team to both reuse a CFDP implementation for the ground software and independently develop the flight software to fit the constraints of the overall spacecraft computer architecture.

References

[1] Solomon, S. C., R. L. McNutt, Jr., R. E. Gold, M. H. Acuña, D. N. Baker, W. V. Boynton, C. R. Chapman, A. F. Cheng, G. Gloeckle, J. W. Head, III, S. M. Krimigis, W. E. McClintock, S. L. Murchie, S. J. Peale, R. J. Phillips, M. S. Robinson, J. A. Slavin, D. E. Smith, R. G. Strom, J. I. Trombka, and M. T. Zuber, The MESSENGER mission to Mercury: Scientific objectives and implementation, *Planet. Space Sci., 49,* 1445-1465, 2001.

[2] Gold, R. E., S. C. Solomon, R. L. McNutt, Jr., A. G. Santo, J. B. Abshire, M. H. Acuña, R. S. Afzal, B. J. Anderson, G. B. Andrews, P. D. Bedini, J. Cain, A. F. Cheng, L. G. Evans, W. C. Feldman, R. B. Follas, G. Gloeckler, J. O. Goldsten, S. E. Hawkins, III, N. R. Izenberg, S. E. Jaskulek, E. A. Ketchum, M. R. Lankton, D. A. Lohr, B. H. Mauk, W. E. McClintock, S. L. Murchie, C. E. Schlemm, II, D. E. Smith, R. D. Starr, and T. H. Zurbuchen, The MESSENGER mission to Mercury: Scientific payload, *Planet. Space Sci., 49,* 1467-1479, 2001.

[3] Santo, A. G., R. E. Gold, R. L. McNutt, Jr., S. C. Solomon, C. J. Ercol, R. W. Farquhar, T. J. Hartka, J. E. Jenkins, J. V. McAdams, L. E. Mosher, D. F. Persons, D. A. Artis, R. S. Bokulic, R. F. Conde, G. Dakermanji, M. E. Goss, Jr., D. R. Haley, K. J. Heeres, R. H. Maurer, R. C. Moore, E. H. Rodberg, T. G. Stern, S. R. Wiley, B. G. Williams, C. L. Yen, and M. R. Peterson, The MESSENGER mission to Mercury: Spacecraft and mission design, *Planet. Space Sci., 49,* 1481-1500, 2001.

[4] CCSDS File Delivery Protocol (CFDP), CCSDS 727.0-B-1, January 2002.

[5] CCSDS File Delivery Protocol (CFDP) – Part 2 Implementers Guide, CCSDS 702.2-G-1, January 2002.