CCSDS FILE DELIVERY PROTOCOL (CFDP) – WHY IT'S USEFUL AND HOW IT WORKS

CCSDS FILE DELIVERY PROTOCOL (CFDP) – WHY IT'S USEFUL AND HOW IT WORKS

Tim Ray
NASA/Goddard Space Flight Center

ABSTRACT

Reliable delivery of data products is often required across space links. For example, a NASA mission will require reliable delivery of images produced by an on-board detector. Many missions have their own (unique) way of accomplishing this, requiring custom software. Many missions also require manual operations (e.g. the telemetry receiver software keeps track of what data is missing, and a person manually inputs the appropriate commands to request retransmissions).

The Consultative Committee for Space Data Systems (CCSDS) developed the CCSDS File Delivery Protocol (CFDP) specifically for this situation. CFDP is an international standard communication protocol that provides reliable delivery of data products. It is designed for use across space links. It will work well if run over the widely used CCSDS Telemetry and Telecommand protocols. However, it can be run over any protocol, and will work well as long as the underlying protocol delivers a reasonable portion of the data. The CFDP receiver will autonomously determine what data is missing, and request retransmissions as needed. The CFDP sender will autonomously perform the requested transmissions. When the entire data product is delivered, the CFDP receiver will let the CFDP sender know that the transaction has completed successfully. The result is that custom software becomes standard, and manual operations become autonomous.

This paper will consider various ways of achieving reliable file delivery, explain why CFDP is the optimal choice for use over space links, explain how the core protocol works, and give some guidance on how to best utilize CFDP within various mission scenarios. It will also touch on additional features of CFDP, as well as other uses for CFDP (e.g. the loading of on-board memory and tables).

KEYWORDS

CCSDS, CFDP, File Transfer Protocol

INTRODUCTION

Technology improvements have enabled spacecraft to fly with larger and larger amounts of memory, and have made on-board filesystems practical. Most new space missions are ready to reap the benefits of a standard file transfer protocol. The CCSDS international standards group looked at the use of *existing* standard file transfer protocols. After finding that none of the existing protocols were feasible for communication over space links, the CCSDS developed CFDP. CFDP is specifically designed for reliable transfer of files over the existing space link protocols, and can also be used over links based on the Internet protocols. In addition to providing basic file transfer capabilities, CFDP also provides some important features that are tailored to the needs of space missions.

THE PROBLEM – CUSTOMIZED LOADS AND DUMPS

In the twenty years that I have been developing software to communicate with NASA spacecraft, I have noticed certain trends. The software that delivers individual spacecraft commands has been reused from mission to mission to mission (over a 10 year period). This software is based on the CCSDS Telecommand protocols. Software to receive real-time spacecraft telemetry has also been reused from mission to mission. This software is based on the CCSDS Telemetry protocols. However, software to *load* on-board programs and tables, and to *dump* science data has often been customized for the particular mission. While the goal is the same from mission to mission (i.e. reliable delivery of programs, tables, and science data), the method used varies. There is clearly room for improvement of loads and dumps.

THE SOLUTION - CFDP

How can loads and dumps be improved? The programs and tables being uploaded are essentially "files", as is the science data being downloaded (e.g. an image from a detector). Therefore, a reliable file transfer protocol can help. It can provide a standard mechanism for these data transfers, enabling extensive software reuse, as well as automation of manual operations.

Which reliable file transfer protocol should be used? The answer depends on one's perspective. If I were a mission manager, I would want a protocol that works well for my particular mission. On the other hand, if I were in charge of NASA, I would want a protocol that works well for *all* missions (near-Earth, Deep Space, intermittent contacts, full-time contact, etc). I would want the protocol to be standard, and the standard to be *stable*, so that software developed this year would still be valid 5, 10, 15 years down the road. While not absolutely necessary, I would prefer that the chosen protocol be compatible with *both* the existing infrastructure (CCSDS protocols) as well as the Internet transport layer protocols (TCP and UDP).

The candidate protocols are the Internet File Transfer Protocol (FTP), CFDP, Multicast Dissemination Protocol (MDP), and Nack-Oriented Reliable Multicast (NORM). The following table summarizes each protocol versus the NASA-wide requirements.

Table 1 – Comparison of reliable file transfer protocols

Protocol	Works for missions?	Stable?	Existing infrastructure + IP?
FTP	Some	Yes	No
CFDP	All	Yes	Yes
MDP	Some/all	No	No
NORM	?	No	No

From the NASA-wide perspective, the best choice is CFDP. It works well for all missions, including both near-Earth and Deep Space, as well as full-time or intermittent contact. While it is a relatively new protocol, it should be stable (historically, CCSDS standards have been stable). It can be run over the existing infrastructure (CCSDS Telemetry and Telecommand protocols) as well as the Internet transport-layer protocols (UDP and TCP). FTP is not feasible for Deep Space missions, where the data link has an extremely long round-trip lifetime. NORM is still under development, and enhancements to MDP are in-progress. Neither MDP nor NORM can run directly over the exiting infrastructure.

WHAT IS CFDP?

Summary

CFDP is a protocol that can provide reliable delivery of *virtual* files across various links, including space links (see Virtual Filesystem below for a discussion of virtual files). *Reliable* means that the received file is a duplicate of the sent file (nothing missing, nothing added, no errors). In addition to this basic capability, there are features tailored for space missions, such as:

- pause all transactions between contacts
- attach file-manipulation directives to a file transfer (e.g. "if this upload of 'table_xyz' is successful, replace the current version; otherwise, leave the current version intact" – see Filestore Directives below)
- send mission-defined messages (e.g. "the ancillary data for this detector image is contained in the accompanying User Message" - see User Messages below)
- record-oriented files (e.g. a file consisting of a set of compressed detector images, each of which is a separate record, and mostly likely of different lengths)
- open-ended files (i.e. a "file" whose length is not known until after transfer has begun)

Virtual Filesystem

With the name "CCSDS File Delivery Protocol", one might expect to be locked into the use of files. However, that is not true. CFDP delivers a block of data with an accompanying "filename". The CFDP Receiver maps that filename to a *Virtual Filesystem*. Neither a traditional disk nor a traditional filesystem are required. For example, to load on-board memory, the filename "RAM:1000" may be interpreted as "put this block of data into RAM, starting at location 1000" (the Sender and Receiver have to agree on the file-naming convention).

Nodes, Transactions, and Service Classes

There is one CFDP *node* per Virtual Filesystem. Each file transfer is a separate *transaction*. Each CFDP node can have multiple concurrent transactions. A node can play the Sender role in some transactions and the Receiver role in other (concurrent) transactions. The core CFDP protocol is point-to-point – files are transferred directly from one node to another. Extensions to the core protocol provide multi-hop capability – files are transferred from one node to another via intermediate nodes. There are two classes of service. With *Acknowledged Service*, reliable delivery of the file is guaranteed (CFDP will request and supply any necessary retransmissions itself). With *Unacknowledged Service*, there is no feedback loop within CFDP, and therefore, no guarantee of reliable delivery (unless reliable delivery is provided within another protocol layer).

Filestore Directives

Each transaction can optionally include transfer of *Filestore Directives* (e.g. "rename file"). If desired, a transaction can include Filestore Directives without any file transfer. If Filestore Directives are included with a file transfer, they are only executed if the transfer is successful. This feature is especially helpful for loading on-board tables and programs. The transaction can include the transfer of a file named something like "table_xyz.tmp", with the attached Filestore Directive "rename table_xyz.tmp table_xyz". The current table will only be replaced if the file transfer is successful.

User Messages

Each transaction can also optionally include transfer of *User Messages* (messages defined by the User; i.e. by the mission). If desired, a transaction can include User Messages without any file transfer.

How do I implement it?

CFDP can be implemented with 4 main modules. The *protocol engine* module contains the protocol logic. This logic is the same for all implementations, so the protocol engine is reusable. The *Virtual Filesystem* module interfaces between the CFDP concept of a file and the actual implementation of files within the CFDP node. For example, a typical CFDP transaction will deliver a block of data with an accompanying "filename", and the Virtual Filesystem module will write the data to the "file". The *Lower-layer communications* module transfers Protocol Data Units to/from the underlying communications protocol layer (e.g. UDP). Finally, the *User* module makes requests of the protocol engine. Despite the name "User" module, there is no need for a person. For example, the User module might monitor the directory where detector images are stored, and request the transfer of each new image file that appears in that directory.

HOW DOES CFDP WORK? – ROUGH SKETCH

All transactions are started by a *Put Request* from the User. Each transaction requires the transfer of a file and/or optional messages (such as Filestore Directives and User Messages).

With Unacknowledged Service, the Sender transmits all the data once (if an underlying protocol layer provides reliable delivery, all the data will get to the Receiver). There are no retransmissions.

With Acknowledged Service, the Sender transmits all the data once, the Receiver periodically requests retransmission of all missing data, and the Sender fulfills each retransmission request. When all data has been received, the transaction shuts down. Shutdown requires the Receiver to tell the Sender that it is finished, and the Sender to acknowledge receipt of this *Finished* message.

HOW DOES CFDP WORK? – PROTOCOL DATA UNITS (PDUS)

CFDP uses the following Protocol Data Units (this discussion leaves out error-handling):

- Metadata – specifies the source and destination nodes for the transaction, the destination filename (if there is a file transfer), and may contain optional fields such as User Messages and Filestore Directives.
- File-Data – Contains one "chunk" of data from the file being transferred (the file is broken up into chunks that will fit on the underlying link). Each File-Data PDU specifies its starting offset within the file (this makes it easy for the Receiver to reassemble the file if PDUs are received out-of-order, and also makes for easy detection of gaps).
- EOF – specifies that the all data has been sent once. If the transaction includes a file transfer, the file-length and file-checksum are included in the EOF.
- Ack-EOF – acknowledge receipt of an EOF.
- Finished – specifies that all data has been received.
- Ack-Finished – acknowledges receipt of a Finished PDU.

HOW DOES CFDP WORK? – UNACKNOWLEDGED SERVICE

Table 2 shows the sequence of PDUs for file transfer via the Unacknowledged Service.

Table 2 – PDU sequence (Unacknowledged Service)

Sender sends	Receiver sends
Metadata	
File-Data #1 (if file transfer)	
File-Data #2 (if file transfer)	
…	
File-Data #n (if file transfer)	
EOF	

HOW DOES CFDP WORK? – ACKNOWLEDGED SERVICE

Table 3 shows the sequence of PDUs for file transfer via the Acknowledged Service.

Table 3 – PDU sequence (Acknowledged Service)

Sender sends	Receiver sends
Metadata	
File-Data #1 (if file transfer)	
File-Data #2 (if file transfer)	
…	
File-Data #n (if file transfer)	
EOF	
	Ack-EOF
	Nak (if needed)
Metadata and/or File-Data (if Nak)	
	Nak (if needed)
Metadata and/or File-Data (if Nak)	
	Finished
Ack-Finished	

The Sender drives the first stage of the transaction (sending all the data once), and then gives control to the Receiver for completion. This handover occurs when the Receiver receives *EOF*. If the first transmission of *EOF* is not delivered, the Sender will recognize the situation (when no *Ack-EOF* is received), and retransmit the *EOF* periodically until the Receiver responds. When the Receiver takes over, it sends *Naks* periodically, as necessary, until all data is delivered. Once all data is delivered, the transaction can be shut down. Shutdown requires the Receiver to send a *Finished*, and the Sender to respond with an *Ack-Finished.* If the first transmission of *Finished* is not acknowledged, the Receiver will retransmit it until the Sender responds.

Configuring CFDP to match the mission environment (timers)

Among the CFDP configuration parameters are these timers:

- Ack-timer – Used by both the Sender and Receiver, to trigger periodic retransmission of *EOF* and *Finished,* respectively, if the partner does not respond. This timer is typically set to be slightly longer than the communications link round-trip lifetime.
- Nak-timer – Used by the Receiver, to trigger periodic checks to see if all data has been received. If there is missing data, the appropriate Nak(s) are generated; otherwise, shutdown begins. This timer is also typically set slightly longer than the round-trip lifetime.
- Transaction-lifetime-timer – Used by both the Sender and Receiver to get rid of very old transactions (presumably, any transaction alive this long has run into a problem that the protocol cannot otherwise handle). This timer is typically set longer than 12 hours.

HOW DOES CFDP WORK? – MISCELLANEOUS

How does CFDP ensure that the transferred data is valid? A *file checksum* is transmitted with each file, so that the receiver can validate the received file. Optionally, a CRC may be included with *each* Protocol Data Unit (e.g. with each "chunk" of the file that is sent); this enables the Receiver to discard data with errors (and request retransmission of that data).

CFDP transactions are initiated and controlled via these *User Requests*:

- Put – starts a transaction
- Cancel – cancels one transaction
- Suspend/Resume – for pausing *one* transaction
- Freeze/Thaw – for pausing *all* transactions (for example, between contacts)

User Indications are output by CFDP so that its work can be monitored. The implementer is free to output User Indications as desired; e.g. text messages to a console, formatted messages to an on-board program, etc.

CONCLUSION

The use of a reliable file transfer protocol by space missions enables significant improvements in both the loading of on-board programs/tables and dumping of science data. CFDP is the only standard file transfer protocol that is directly compatible with the existing space link protocols as well as the Internet protocols. Along with basic file transfer capabilities, features tailored to the needs of space missions are provided. CFDP is an international standard that is gaining acceptance within the space community. Among the missions that are currently using, or planning to use CFDP:

- AlSat-1 – An Algerian satellite that is currently flying CFDP over IP protocols. This near-Earth mission has achieved reliable file transfers at 8 megabits per second via the CFDP Acknowledged Service Class.
- Deep Impact – A NASA Deep Space mission being built by the Jet Propulsion Lab.
- Mars Reconnaissance Orbiter – a NASA Deep Space mission being built by the Jet Propulsion Lab.
- MESSENGER – A NASA Deep Space mission being built by the Applied Physics Lab.
- James Webb Space Telescope - the NASA follow-on to the Hubble Space Telescope; a NASA/Goddard mission currently being designed. Will operate at the Second Lagrange point (1 million miles from Earth).
- Global Precipitation Measurement - a NASA/Goddard mission (near-Earth); currently being designed; currently planning to use CFDP.

ACKNOWLEDGEMENTS

The members of the CCSDS group that developed CFDP deserve acknowledgement for their cooperative development, implementation, and testing effort. In particular, Scott Burleigh of the Jet Propulsion Laboratory deserves special recognition for patiently guiding the effort.

REFERENCES

[1] CCSDS File Delivery Protocol (CFDP), CCSDS 727.0-B-2, Washington D.C., Consultative Committee for Space Data Systems, October 2002
[2] CCSDS File Delivery Protocol (CFDP) – Part 1: Introduction and Overview, CCSDS 720.1-G-1, Washington D.C., Consultative Committee for Space Data Systems, January 2002
[3] CCSDS File Delivery Protocol (CFDP) – Part 2: Implementers Guide, CCSDS 720.2-G-1, Washington D.C., Consultative Committee for Space Data Systems, January 2002

Note: All CCSDS documents are available at http://www.ccsds.org