FSW Workshop 2018
A Case for Bundle Protocol in Space

JP Swinski/NASA GSFC

Jonathan Wilmot/NASA GSFC

December 5th, 2018

Abstract

NASA, through the Advanced Exploration Systems (AES) project, is investing in the development and infusion of delay tolerant networking (DTN) protocols for use on future space flight missions. The cornerstone of the DTN suite is the Bundle Protocol which provides network layer addressing and routing of data blocks. In 2017, the Plankton, Aerosol, Cloud, and Ocean Ecosystem (PACE) mission was selected as the first in-house robotic science mission to implement the Bundle Protocol for downlink of housekeeping telemetry. One year into the design and incorporation of the Bundle Protocol on PACE, this presentation makes a case for using the Bundle Protocol for communication with future space assets. Specifically, the use of the Bundle Protocol (1) simplifies relaying data through store and forward routing and custody transfer; (2) simplifies downlink management through delivery guarantees; and (3) simplifies storage services through block level interactions with memory devices.

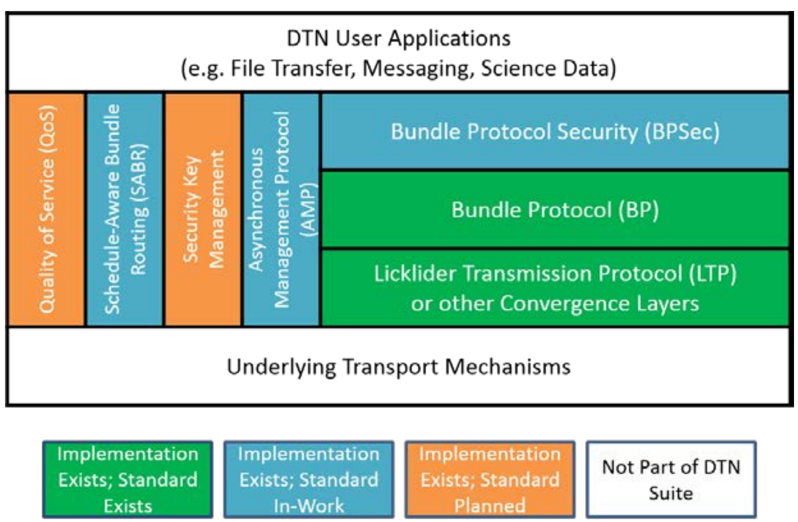
DTN Overview

- DTN is a network layer for space comm.
  - **<u>Standardized</u>** automated multi-hop end-to-end data delivery
  - **<u>Protocols</u>** have been developed to accommodate space link and mission operation conditions (delays, data loss, disconnections, data rate transitions, etc.)

- NASA’s goal for DTN is to provide:
  - Network layer **<u>interoperability</u>** with international and commercial partners
  - Greater service **<u>infrastructure</u>** provided to missions

What is DTN

- DTN is a suite of protocols designed to provide internetworking functionality in conditions that are not optimal for Internet Protocols
- The primary DTN protocol, Bundle Protocol (BP), uses a store and forward approach for network layer end-to-end data delivery
- Operations of a DTN network requires DTN-enabled nodes integrated into data flows with the ability to configure, control, and receive status from the nodes
- The suite of protocols are at various levels of standardization and within the Internet (RFC) and Space (CCSDS) communities



What is Bundle Protocol (BP)

- The bundle protocol is a network service layer standard designed for networks that experience large delays or intermittent connectivity. The protocol provides the following characteristics:
  - Routing information
  - Guaranteed delivery (at least once, or at most once)
  - Ownership (custody) transfer
  - Data lifetimes

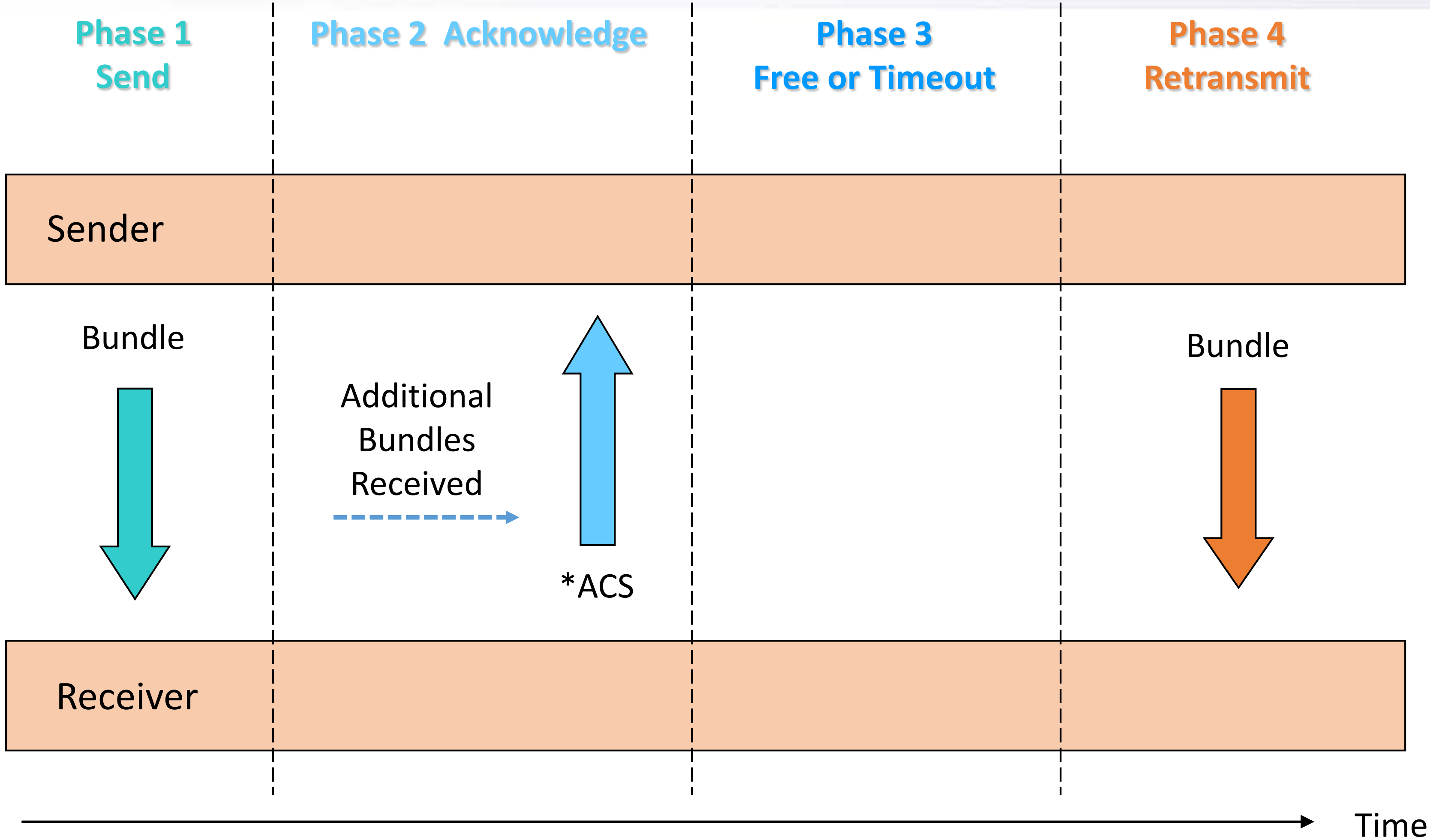
- The bundle protocol can be extended through standard extensions to provide the following additional characteristics:
  - Variable resolution acknowledgment
  - Data integrity

- The bundle protocol *does not* guarantee the following but requires the mission implementation to include a mix of other protocols and conventions to provide:
  - Order of delivery
  - Data streaming (arbitrary slicing and reconstituting of original stream of bytes)



Bundle Protocol Overview

Bundle Protocol Message Passing



*Aggregate Custody Signal is a special bundle that gives a positive indication that a bundle or set of bundles have been received. There are no negative acknowledgments.

What Does a Bundle Look Like

Primary Bundle Block

Version	Bundle Processing Control Flags
Block Length (Bytes) of Rest of Block	
Destination Scheme Offset	Destination SSP Offset
Source Scheme Offset	Source SSP Offset
Report-to Scheme Offset	Report-to SSP Offset
Custodian Scheme Offset	Custodian SSP Offset
Creation Timestamp (seconds since 1/1/2000)	
Creation Timestamp Sequence Number (0 up)	
Lifetime (seconds after creation)	
Dictionary Length (Bytes)	
Dictionary Character Array (Sequence of Null Terminated Strings)	
[Fragment Offset (Original Payload Starting Byte)]	
[Original Payload Total Length (Bytes)]	

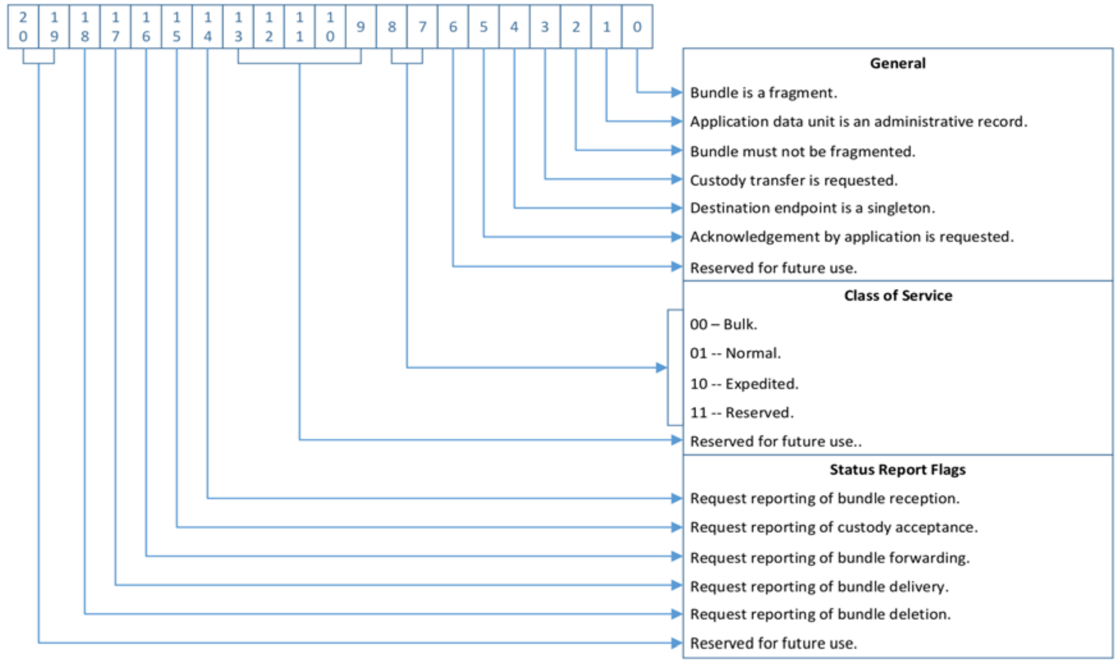
Bundle Payload Block

Block Type (=1)	Block Processing Control Flags
Block Length (Bytes)	
Bundle Payload (variable)	

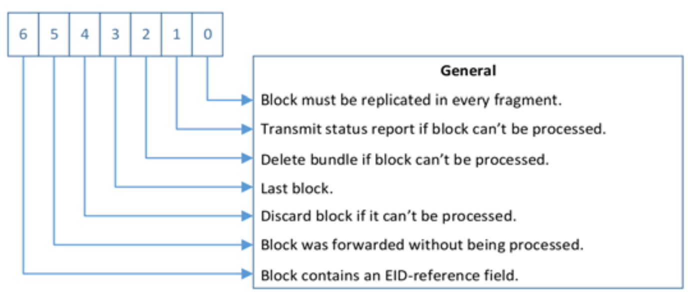
Extension Block Structure

Block Type	Block Processing Control Flags
[EID Reference Count (Example = 2)]	
[EID_1 Scheme Offset]	[EID_1 SSP Offset]
[EID_2 Scheme Offset]	[EID_2 SSP Offset]
Block Length (Bytes)	
Bundle Payload (variable)	

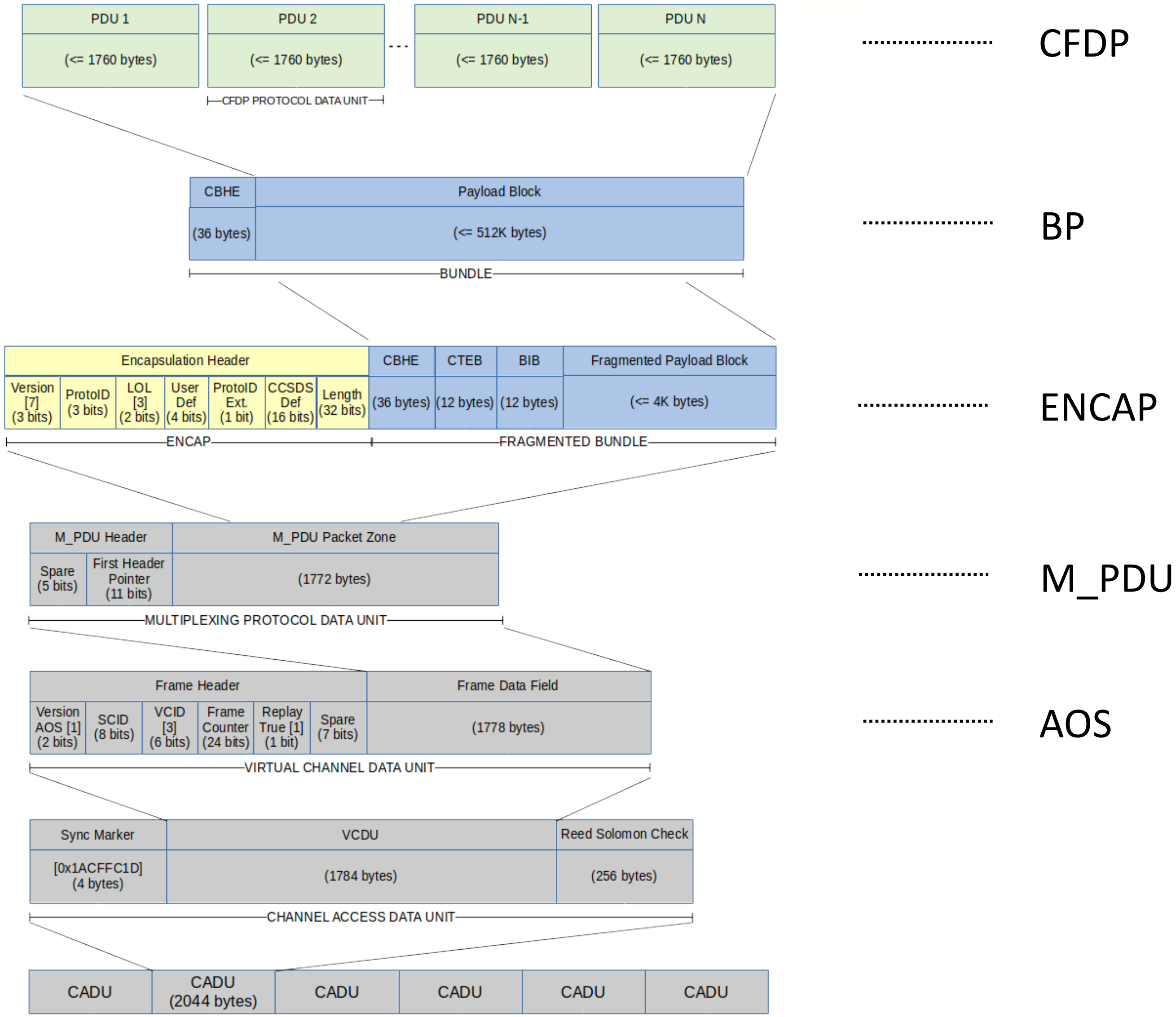
Bundle Processing Control Flags



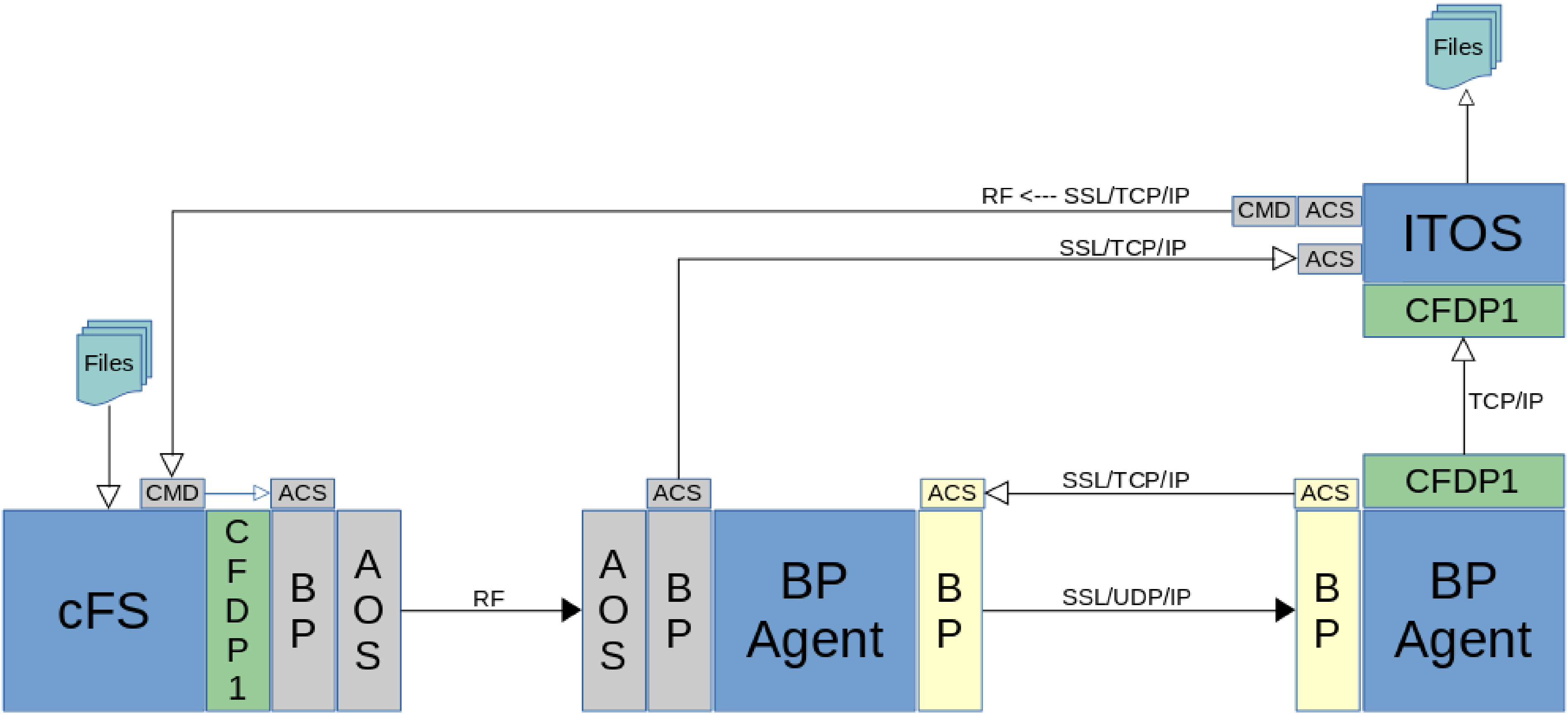
Block Processing Control Flags



Telemetry Downlink Stack



Mission Context Diagram

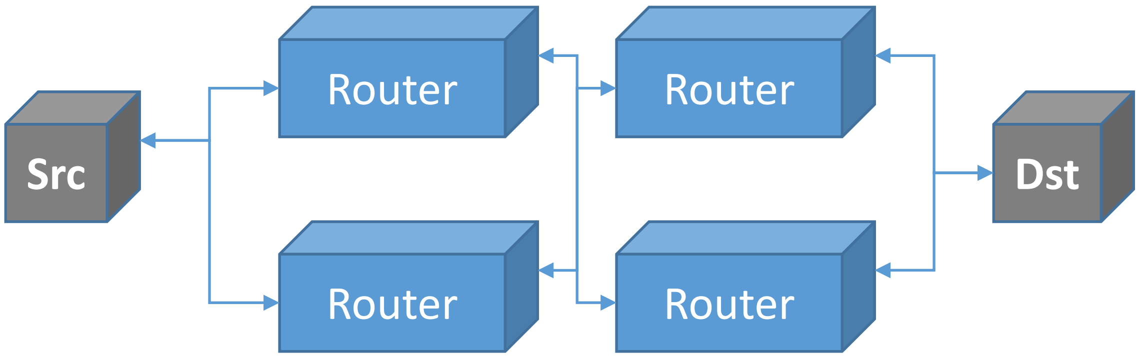


Flight Ground Station MOC

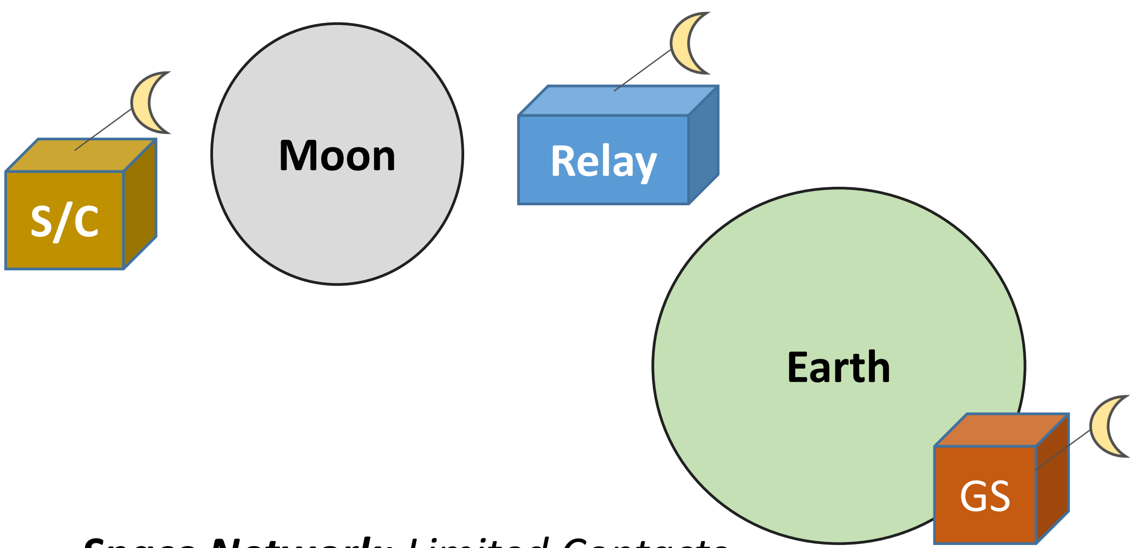


Case 1: Simplifies relaying data through store and forward routing and custody transfer.

Multi-Hop Data Transfers



Terrestrial Network*: Connections Always Available*



Space Network: *Limited Contacts*

- Ground networks have physical links that are **always connected**, yet often governed by bandwidth.
- Space links are governed by **scheduled contacts**, yet bandwidth is typically high.
- Protocols that require the illusion of a continuous connection between the source and destination (i.e. TCP/IP) do not fit the constraints of space networks.
- We want a protocol that takes advantage of short periods of high bandwidth connections that are disjoint from other hops along the path.

Bundle Routing

Bundles are **routed** using a destination address embedded in the bundle header.

The bundle protocol requires bundle agents to be able to **store bundles** that are not able to be sent right away. This applies to data sources as well as relays.

Different flows of bundles can be **prioritized** based on class of service identifiers in the bundle header as well as bundle source and destination addresses. This allows missions to manage data flows in real-time to address failures and operational contingencies.

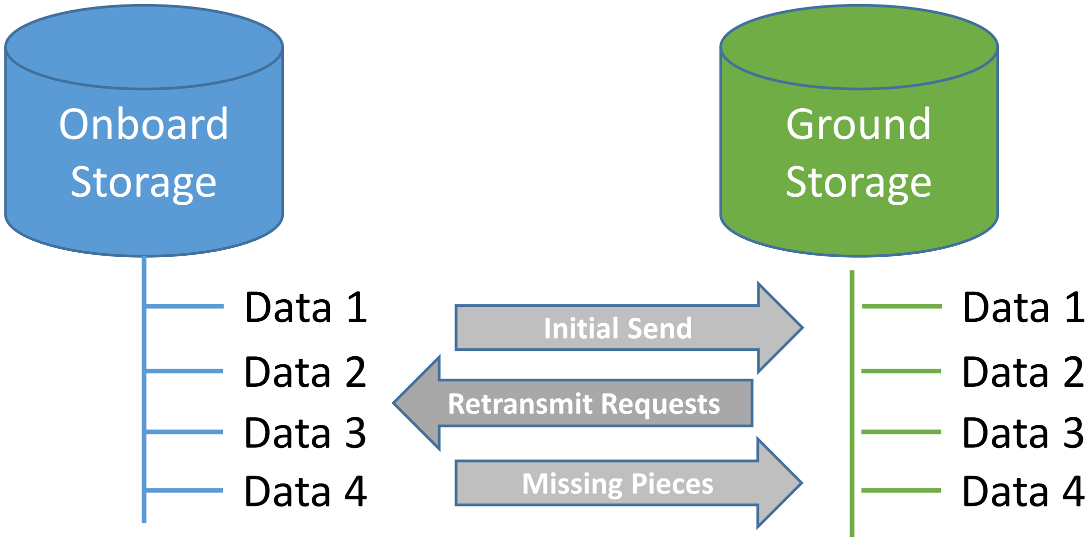
Given that onboard resources are often very limited, a bundle relay can **take custody** of a bundle and send a message to the bundle sender saying that the sender can delete the bundle in their storage system. This fits well in the context of lunar or deep space mission relays as well as near earth orbiting missions where data downlink bandwidth greatly exceeds the ground stations terrestrial network bandwidth.



Case 2: Simplifies downlink management through delivery guarantees.

Fundamental Challenge

Reliably transfer data from source in space to destination on ground



- How do you keep track of what you've sent?
- How do you detect that something is missing? How long do you wait? What about corruption?
- What is an efficient way of only sending the pieces you need?
- What is an efficient way of requesting only the pieces you need?

→ These problems are solved again and again, in slightly different ways, for every project.

Bundle Data Delivery

Bundles are **identified using a single positive integer**. This makes onboard code structures that keep track of bundles simpler and less resource intense.

A bundle agent is an **aggressive sender** – meaning it will keep trying to send a bundle until it receives positive acknowledgement. There are no negative acknowledgments. This behavior is configured with two parameters:

- ***Timeout***: how long to wait for positive acknowledgement before resending
- ***Lifetime***: how long to wait before giving up altogether

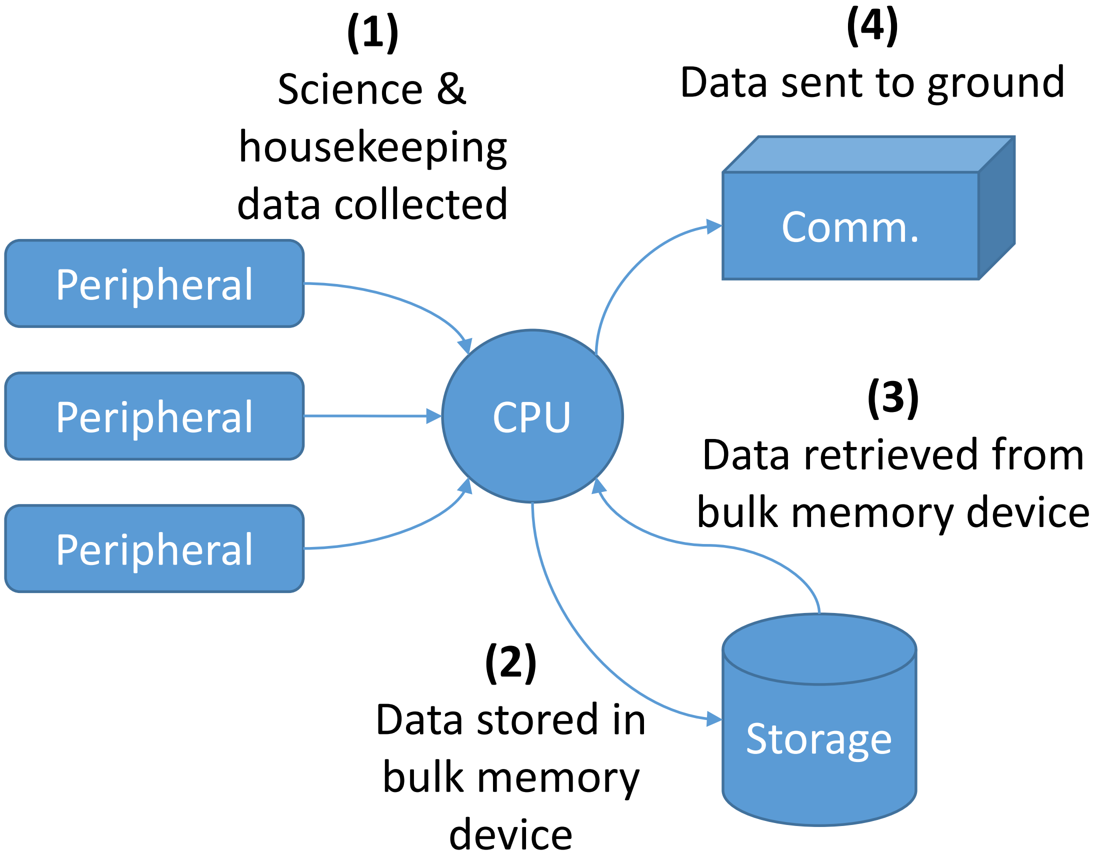
A receiving bundle agent has the option of acknowledging every bundle, or sending an **aggregate acknowledgement** of many bundles at a configurable rate (that matches the timeout and lifetime of the sender).

→ These mechanisms are not new or ground breaking; rather they are simple and standardized.

Case 3: Simplifies storage services through block level interactions with memory devices.

Typical Telemetry Downlink Context



- Data is usually collected continuously, or from a series of observations that occur over some ***extended period of time***.
- Data is stored in bulk memory at a ***lower*** (and easier to manage) ***rate***.
- Data is pulled from bulk memory at a ***high rate*** to support transmission over a limited time window (i.e. a contact)
- Data is sent to the ground encapsulated in protocols that provide the means for ***detecting errors*** and ***requesting retransmission***.

Challenges with Telemetry Downlink

- The mismatch between the lower rate of data storage and the high rate of data transmission creates an unbalanced stress of onboard resources at the time of transmission.
- The data must go from its raw form collected from peripherals, to its fully encapsulated form at the time of downlink.
- Therefore, the more processing that occur earlier in the flow to put the data in its final form, the lower and smoother our peak resource utilization will be.

In other words – we want to pre-build as much of the data's final downlink form as possible in order to reduce how much CPU and memory we need onboard to support data downlink.

Advantages to Bundles

- Since a bundle is a self-contained block of data with a well-defined prepended header, it can be fully* built ***at the time of bundle storage***.

- Therefore, the processing required for the bundle protocol can be spread out across the longer data collection/storage time period, and the processing required at the time of data transmission is minimized.

- Since a bundle is a pre-sized data block identified by a single unsigned integer, its storage location can be managed through the ***use of a lookup table*** (e.g. LUT[id] => block address)

- Therefore, the processing required for retrieving a bundle from memory is minimal which further reduces the amount of processing required at the time of data transmission.

**Note: on PACE we jam a custody ID just before transmission to optimize the generation of aggregated acknowledgments.*

Acronyn List

AES	Advanced Exploration Systems
ACS	Aggregate Custody Signal
AOS	Advanced Orbiting Systems (CCSDS data link standard)
BP	Bundle Protocol
CCSDS	Consultative Committee for Space Data Systems
CFDP	CCSDS File Delivery Protocol
cFS	Core Flight System
ENCAP	Encapsulation (CCSDS standard)
FSW	Flight Software
GS	Ground System
IP	Internet Protocol
ITOS	Integrated Test and Operations System
LUT	Look up table
M_PDU	Multiplexing Protocol Data Unit
MOC	Mission Operations Center
NASA	National Aeronautics and Space Administration
RFC	Request For Comments
S/C	Spacecraft
SSL	Secure Socket Layer
TCP	The Assert Set of Tools for Engineering

Backup

Storage Service: Bundles in Contrast to Files

Things that are nice about **files**:

- Provides hierarchical organization and identification of data.
- Data can be grouped into meaningful clumps.
- Can match the format of the original data source – e.g. images.
- Can simplify and complement ground processing that generates data file sets.

Things to worry about with **files**:

- Text strings as names, and optimized key-value lookup is non-trivial if you are both memory and CPU constrained (like most of our flight computers, e.g. on previous projects we've monitored worse case number of collisions on a file name hash in the FAT)
- Fragmentation; how to best tune block size to balance internal vs external fragmentation
- Managing a full file system is not as simple as a circular buffer
- File APIs assume sequential access (seek to offset, not directly addressable… of course underneath it can be the same); for replaying data, we often want direct block addressability.

Things that are nice about **bundles**:

- Data can be organized via service numbers for source and destination similar to IP ports.
- Bundles are self-contained and are agnostic to the data inside
- Bundles can be tuned to match block sizes and found with simple look up tables

Things to worry about with **bundles**:

- SDNV encoding of header fields consumes a lot of CPU cycles
- Changing a header field can often cause the memory layout of the header to change, breaking DMA transfers

DTN Network Model

DTN 6 Layer Model

Application Layer
application data reconstitution, analysis, presentation

Transport Layer
process to process ***message*** delivery
encoding, encryption, compression, format

Network Layer
host-to-host **bundle** delivery
fragmentation, store and forward routing, custody transfer

Bundle Hop Layer
data transfers between neighboring network DTN elements

Link Layer
data transfers between neighboring network path elements
error detection, flow control, ***frame*** synchronization

Physical Layer
physical medium carrying representation of a ***bit*** stream

DTN Space Stack

Command & Telemetry system	CFDP
Bundle Agent	BP
PTP	AOS/ENCAP
GS	RF

Protocols for data streaming, ordered delivery, etc.

Protocols for hops over existing networks like terrestrial IP

cFS BP Application

- The BP application development is funded by the NASA Advanced Exploration Systems (AES) program for initial use by the PACE mission.

- The application targets the cFE 6.5 release and will be compatible with cFE 6.6. No external dependencies are required, but will integrate with the cFS CFDP application (CF), and a POSIX compliant file system when provided.

- The bundle protocol implementation is a new development optimized for memory constrained embedded environments.

- The implementation conforms to the CCSDS Blue Book recommendation 734.2-B-1 issued September 2015, which is based on RFC 5050.