2025 6.1800 Design Project:

Enhanced Extraterrestrial Communication with SolarNet

Release 1.1: See also DP FAQ and DP Errata

There are five deliverables for this design project:

1) **DP Prep (DPP):** To help you prepare with your team design effort, this assignment will require some guided analysis of the DP specification below. This assignment will be written by each student individually, will be approximately 750 words, and is due at **11:59pm on February 28, 2025**.
2) **DP Preliminary Report (DPPR):** This preliminary report will lay out your key design decisions, including both a functional system design and a sketch of any data structures, storage management, and/or network protocols required to achieve your design. It will not include any significant evaluation. It will be written by your team as a whole, will be approximately 2,750 words, and is due at **11:59pm on March 21, 2025**.
3) **DP Presentation:** This presentation will address the feedback received on the DPPR, and any corrections or updates to the design project specification. It will also outline evaluation criteria and use cases you will use later for evaluating your design. All team members must participate in the presentation. It will be delivered live and in-person with your recitation instructor. It will occur during the week of **April 14 - 18, 2025**.
4) **DP Report (DPR):** This will be your full report. It will include your final design, all diagrams appropriate for that, your evaluation of your design and a review of how effectively your design addresses the specified use cases. It will be written by your team as whole, will be approximately 6,000 words and is due at **11:59pm on May 5, 2025**.
5) **Peer Review:** In Tutorial your team will have done an early "review," providing informal feedback to another team on their design. For this peer review, you will individually review a few specific sections of that (same) other team's final report and will address some specific questions about that report. It will be approximately 250 words and is due at **5:00pm (NOT 11:59pm) on May 9, 2025**.

Your assignment for each of the five parts above will be distributed in separate "assignment" documents on the dates stated in the course calendar.

As with real-life system designs, the 6.1800 design project is under-specified, and it is your job to complete the specification in a sensible way given the stated requirements of the project. As with designs in practice, the specifications often need some adjustment as the design is fleshed out. Moreover, requirements will likely be added or modified as time goes on. We recommend that you start early so that you can evolve your design over time. A good design is likely to take more than just a few days to develop. A good design will avoid unnecessary complexity and be as modular as possible, enabling it to evolve with changing requirements.

Large systems are never built by a single person. Accordingly, you will be working in teams of three for this project. Part of the project is learning how to work productively on a long-term team effort. **All three people on a team must be in the same tutorial.**

Late submission grading policy: If you submit any deliverable late, we will penalize you one letter grade per 48 hours, starting from the time of the deadline. For example, if you submit the report anywhere from 1 minute to 48 hours late and your report would have otherwise received a grade of A, you will receive a B; if you submitted it 48 hours and 5 minutes late, you will receive a C.

You must complete the three team design project components above to pass 6.1800. For the other two (individual) components of the design project, the contribution to your overall grade will be whatever grade you receive on that component. Thus, if you choose not to do one or the other of them, you will receive an F for that component only as a contribution to your overall grade.

Space network communication:
Enhanced Extraterrestrial Communication with SolarNet

1 Introduction

Your team is taking on a design role for part of the new (slightly fictional) **SolarNet** project, the extraterrestrial network being designed by NASA. SolarNet carries three types of data: management data (to control satellites and other objects), telemetry data, and mission-critical information. To date, all extraterrestrial communication in SolarNet has been directly between a machine on the earth (a *terrestrial* antenna) and a craft in space. We refer to these types of communications as "point-to-point": two endpoints communicate directly with one another, with no additional devices in between.

Extraterrestrial devices can be all sorts of things: sensors and telescopes collecting data, rovers on different planets, antennas, etc. These devices may be as close as the International Space Station, or further to the Moon, Mars, or beyond. However, to extend communications to the farther reaches of space, SolarNet needs to grow beyond point-to-point links.

As a result, NASA has upgraded SolarNet to allow traffic to move through a network of satellites. In some cases, such as between the Moon and Mars, data may not ever reach Earth. This environment poses challenges not typically present in terrestrial communications. For example:

- Extreme variability in roundtrip times[1] between end points
- Extremely long roundtrip times in some cases (e.g., the round trip at the speed of light between Earth and Mars varies between 8 and 40 minutes)
- Intermittent connectivity, both predictable and unpredictable
- The need to take advantage of scheduled and predicted connectivity opportunities[2]
- Traffic errors due to a wide variety of influences, such as noise, not only packet loss as in terrestrial networks:
- Extremely limited resources such as bandwidth[3] and power on occasion.

To address these challenges, the networking community has defined the *Bundle Protocol*. This protocol describes a *store-and-forward* communications protocol between nodes that can handle long and unpredictable delays in communication. The protocol itself requires network storage but does not include a design for such a storage system. Furthermore, although the Bundle Protocol itself is fully specified, such a protocol specification does not define the API for using it. In addition, the operating environment includes a simple approach to routing as is described in more detail below. You will be

[1] Roundtrip time is the amount of time it takes for a message to be sent and a response sent back.
[2] For example, an intermediate node may not be able to forward a bundle directly, but might have knowledge that its nearest neighbor will be able to later, so it may pass traffic to that neighbor for future forwarding.
[3] Bandwidth is the maximum rate at which data can be transferred. In our case, different devices will support different bandwidths.

designing an extension to the Bundle Protocol and its run-time environment to improve performance. To make this useable you will also need to define interfaces (set of calls) to the elements you design.

Your task is to enhance the simple form of the Bundle Protocol, as well as the forwarding decision-making process, to support a network storage system distributed across all the nodes in the network. Your design will need to support the variety of applications that use SolarNet (detailed in Sections 3.1 and 4.2), each of which have different requirements. At times, these applications may have competing requirements – for example, mission-critical data about an impending solar flare may need to be sent at the same time as a large software update. You will need define and justify any trade-offs in your design (e.g., prioritizing one data type over another in a particular situation).

Before we go on, it's worth knowing that NASA is indeed designing and building a less ambitious version of what is proposed here for a communications network.[4] We have simplified our specification below, to make this a manageable project for one semester. Although real-world specifications for components of this project (e.g., the Bundle Protocol) are widely available, you should take what is described here as authoritative for this project, rather than what you may find in other papers, documents, websites, etc.

2 The SolarNet Underpinnings

The existing SolarNet infrastructure consists of the satellites and other hardware, the Bundle Protocol that defines communications between nodes, and the routing protocol that allows nodes to make decisions about where to send data. After the Bundle Protocol and routing protocol discussions, we enumerate a set of calls to provide an interface for you to both protocols.

2.1 Physical Modules: Satellites and other hardware

SolarNet consists of:

- 100 Low Earth Orbit communications satellites (LEOComs). These all operate at a single altitude.
- 10 Geosynchronous Equatorial Orbit communications satellites (GEOComs) stationed at a significantly higher altitude than the LEOComs
- A small number of relay stations (we will call them "relays") near Earth and other solar system bodies (in our initial case the Moon and Mars)
- A set of antennas on the surfaces of the Earth, the Moon, and Mars. On Earth there will be a number of these, and on the Moon and Mars one each.
- Two specialized LEO satellites that collect data and images of the Earth. Such land satellites have many uses including tracking of vegetation, water, etc. In our case, the information of interest is ground movement that will be used as a predictor of land or mud slides.
- Two specialized GEO satellites, directly opposite each other, which collect Solar data. Because of this arrangement, one of these satellites will always be facing toward the Sun. These are research instruments that will collect data about the Sun on an ongoing basis. We are particularly interested in data that is of use in predicting Solar storms, because of the negative impact they can have on our communications systems .

Figure 1 below shows the conceptual design; it is not to scale.

[4] The NASA program is to design and build LunaNet, based on the specification and their implementation of the Bundle Protocol. We point this out, because these are real specifications, designs, and implementations.

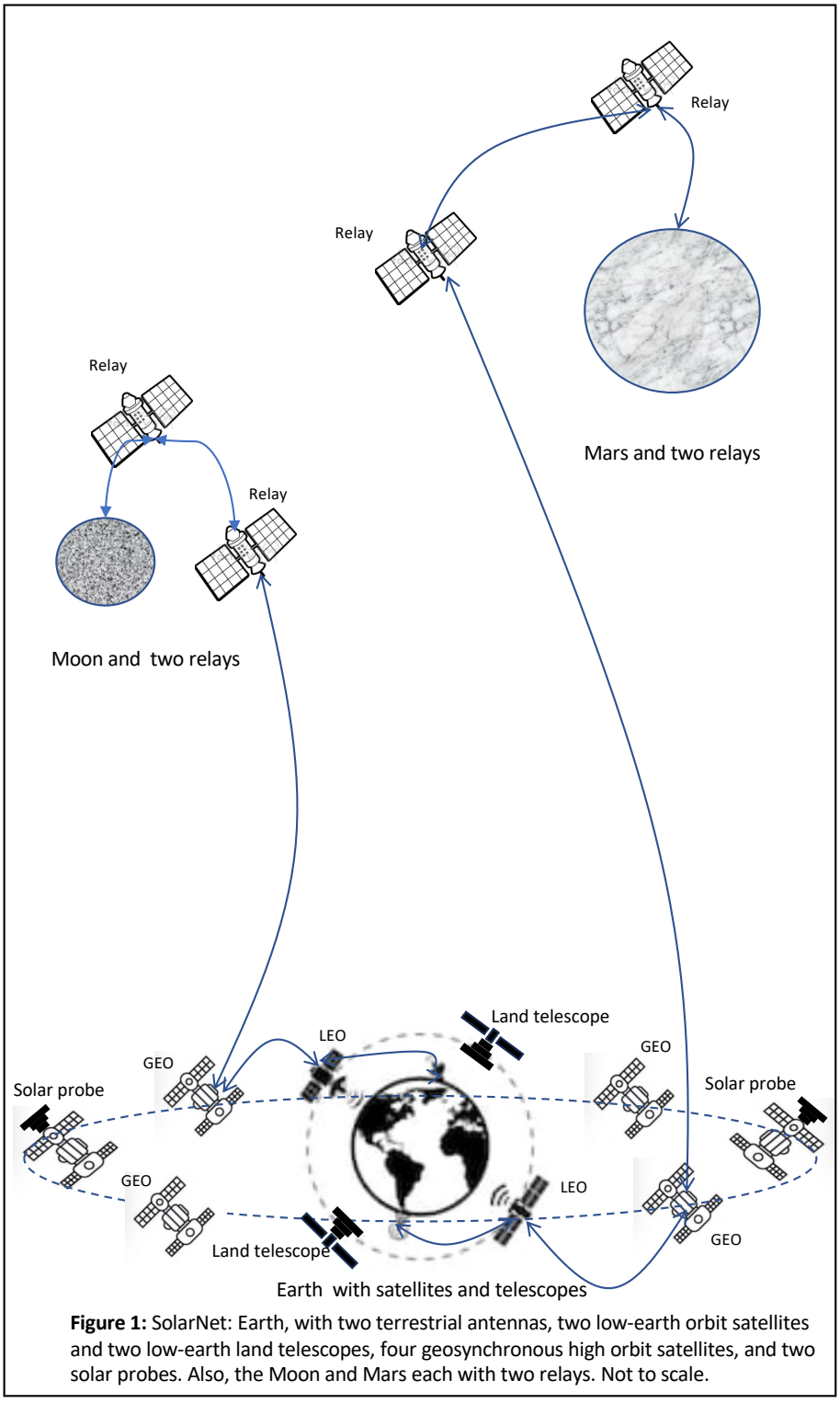


Figure 1: SolarNet: Earth, with two terrestrial antennas, two low-earth orbit satellites and two low-earth land telescopes, four geosynchronous high orbit satellites, and two solar probes. Also, the Moon and Mars each with two relays. Not to scale.

2.1.1 LEOComs and GEOComs

The LEOComs have direct communication with Earth antennas. They can also communicate with their nearest LEOCom neighbors at any given time and with any GEOCom that is in range under a few constraints discussed below. In our design, the LEOComs are in polar orbits, each of which takes 1.5 hours. At this rate, with the Earth rotating underneath them, each LEOCom will return to the same location over the Earth once every 24 hours. These satellites are all at an altitude of 833Km.

The GEOComs are in orbits on the Equator, at about 36,000Km from Earth, but are *geosynchronous*; this means that they orbit once every 24 hours in sync with the Earth. GEOComs can communicate with their GEOCom neighbors, as well as any LEOCom, relay, and antenna that is in range. In practice this is limited to the Moon antenna only. It turns out that the GEOComs are not positioned to be able to reach the Earth antennas. The Moon antenna may be reachable by the GEOComs directly at certain times, but for Mars and more remote locations use of the relays is necessary.

There are two specilialized types of satellites, land telescopes and solar probes. There are two of each type. The land telescopes are LEO satellites at a slightly different altitude than the LEOComs. During normal, non-critical periods, they transmit their data directly to one of two ground receivers, when they are over them, but in an emergency, they can use the SolarNet syste to communicate with either nearby LEOComs or GEOComs. The solar probes are in the same geosynchronous orbit as the GEOComs and use the SolarNet system for all communications. All software updates to these satellites use the SolarNet system. For simplicity, in this work, we assume that the land telescopes receive the same software updates as the LEOComs and the solar probes those of the GEOComs.

2.1.2 Relays

Relays are elements of our network that are not specifically orbiting around Earth. They enable communication from other solar system bodies – the Moon and Mars in our initial design – to the GEOComs. In the case of the Moon, relays are not always needed; for instance, if the Moon-based antenna is in range of a GEOCom. For Mars and more remote communication, relays must be used because these relays will boost the signal to maintain the bandwidth.

2.1.3 Detailed Specs

The system specification includes the following constraints on bandwidth, orbit times, the number of each kind of device in the communications system, and the amount of storage available on each device. Both the Earth and Solar telemetry satellites are included in the list below. It is important to note that they are only origins of data and provide no additional communications capacity. They also have significant storage as shown in Table 1 and each has 64GB of memory. All non-terrestrial nodes will communicate via point-to-point optical links; “point-to-point” means the endpoints communicate directly with one another.

Device Type	**Bandwidth capacity (each direction)**	**Orbit time**	**Number of devices**	**Amount of storage**
Earth Antenna	2GB/s	-	25	10TB
LEOCom	2GB/s	1.5 hrs	100	.5TB
GEOCom	1.2GB/s	24 hours (stationary)	10	.5TB
Relay	622MB/s	None	4 (2 each at Moon/Mars)	.5TB
Moon/Mars Antenna	622MB/s	Revolution time of the body	1 each on Moon and Mars	10TB
LEO Earth telemetry satellite	2GB/s	Precessing, returns to same spot every 12 days	2	10TB
GEO Solar telemetry satellite	1.2GB/s	24 hours (stationary)	2	10TB

Table 1: The physical modules of the system and their specifications. All devices have 64GB of memory.

Note that the bandwidth on a link between two different types of devices is limited by the bandwidth of the lower capacity device. For example, a GEOCom, which itself has a capacity of 1.2GB/s, will only be able to communicate with a relay at 622MB/s, because the relay is more limited.

With this in mind, we identify other communications constraints among the nodes that may be important to you:

- At any given time, there is a 100% probability that at least one LEOCom is communicating with the ground.[5] There is a 50% probability that two are communicating with the ground, a 25% chance that three are, and so forth.
- 90% of the time a LEOCom can reach one of the GEOComs. When the LEOComs are at the poles, their communication may be severely disrupted.
- At each of the Moon and Mars, there are two relays: one that is always in contact with the relevant antenna and the other that is in contact with the GEOComs. There will be times when NASA will need to move the relays with respect to both the Moon or Mars and the Earth satellites to maintain connectivity, leading to service disruptions. Although these movements will be planned, their frequency and disruption are unknown at present. They will be reflected in the routing tables (see Section 2.3). It is important to remember that although communication between the Moon region and Earth only takes a few seconds at the speed of light, between Mars and Earth that time stretches to between about 4 and 20 minutes each way depending on where in their orbits Mars and the Earth are.
- Each device can only be sending to or receiving from one other device at any given moment. Each device has one transmitter and one receiver, so it can send and receive in parallel, but a device cannot be receiving from (or sending to) multiple devices at the same time. However, bundles can be interleaved: Node A can receive a bundle from Node B and then immediately receive from Node C, assuming they are immediate neighbors.

Although these are the numbers at present, NASA hopes to deploy more of each type of device over time. Your design should allow for expansions in the capacity of the system.

2.2 Communications: The Bundle Protocol

Nodes in SolarNet communicate via the *Bundle Protocol*, which we first explain via an example. That is followed by the details of the protocol, the format of a bundle, the unit of transmission, and finally a useful set of system calls available to you.

2.2.1 Introduction to the Bundle Protocol: An example

Suppose a person on Earth wants to use the Bundle Protocol (BP) to send an email containing birthday wishes for an astronaut on the Moon. On Earth, the BP running on the sender's computer will receive the data from the email program, arrange it into a bundle (the set of bits to be sent, plus any additional information required for sending), and forward the bundle to the first bundle node, which we'll call BN0. Let's assume BN0 is a LEOCom. In the simplest case, BN0 will hold the bundle in its storage until a GEOCom (BN1) comes into range; then BN0 will forward the bundle to BN1. BN1 will store the bundle

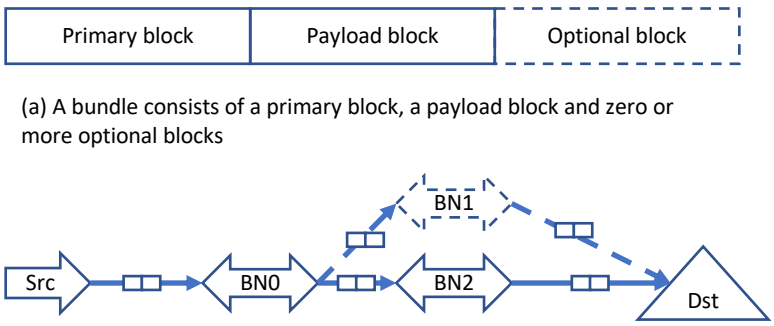
[5] There aren't many ground antennas and the LEOComs are moving, so this is saying that the system guarantees that among the LEOComs it is always the case that at least one of them is in ground communication with an antenna. The other antennas may have no LEOCom connection.

until the Moon-based antenna is available, at which time it will forward the bundle to the Moon-based antenna.

However, it is also possible that BN0 finds that it can reach two GEOComs -- BN1 and BN2 -- because they are both within the direct communication alignment. In that case, BN0 must make a choice: to send to BN1, BN2, or even both (which may increase the odds of the bundle being delivered sooner). The Routing Protocol provides the information about where a node can send next; we discuss this protocol in Section 2.3.

2.2.2 The Basic Bundle Protocol

Every satellite and relay will have long-term storage. Your first challenge is to design the organization of that storage, both on the individual nodes, and collectively among the nodes, to improve communication over the simple model in use currently. The current simplistic Bundle Protocol approach assumes an extremely simple local storage model and no cooperation or collaboration among the nodes' storage systems.



(a) A bundle consists of a primary block, a payload block and zero or more optional blocks

(b) Bundles are stored and then forwarded from the source (Src) through a series of bundle nodes (BN) to the destination (Dst). A bundle may follow more than one path or be fragmented, in which case the fragments may follow different paths. This example has only bundles with no optional blocks.

Figure 2: Example of simple, best-effort delivery of bundles. The Bundle protocol forwards each bundle through a series of bundle nodes, each with network storage, toward the destination. It may try more than one path. It may also fragment the bundle and send the fragments on a variety of paths. Bundle protocol will reassemble the full bundle in either an interior bundle node or at the destination. This example has no responses or acknowledgements.

The Bundle Protocol (BP) is a simple, unreliable, *store-and-forward* protocol: when a bundle arrives at a node, the node forwards it on its path toward the destination and hopes for the best. This has two implications: the node must have enough storage to collect the bits in the bundle until they have all arrived, and once it's ready, the node simply sends those bits along without providing any confirmation that they have arrived at their destination.

The nature of this protocol introduces some challenges:

- **Fragmentation:** A bundle node may move out of range before a bundle has been completely sent to it. To address this problem, the BP can *fragment* a bundle and only send as much as is feasible from one node to the next, allowing for the remainder to be sent along a different path. At each fragmentation episode it only creates two fragments: the first transmissible one and the rest. Fragmentation can be repeated, each time sending the initial fragment (i.e. transmitting more of the bundle) and holding the remainder for further processing. Each fragment will be tagged by the function that does the fragmentation (see the list of system calls below) with the fact that it is a fragment and where it fits into the whole. The bundle protocol has the capability of fragmenting, but you will need a component of the system to decide when and how much to

fragment off and therefore how much is remaining of a bundle, to be handle in whatever way you decide is best.

- **Status Reports:** The original sender never knows whether the bundle arrived or not. To address this issue, the sender has the option to set a flag in the original request asking that bundle delivery status report messages be sent back to the original from each forwarding point along the path; whether a node honors this reporting request or not is a local choice. These responses are sent unreliably. A report will include as many of the following as it knows about at the time of sending the report. See Section 2.2.3.3 for details on the types of reports that might be included.

- Although the BP specification indicates that these responses are handled automatically by a node, for your design, you may want to decide that they will happen explicitly under your specified conditions. Therefore in the list of calls available, we have also included sending administrative reporting bundles. A forwarding node sending such a reply cannot turn on the "confirmation" flag on the report. These status messages can have a significant cost, because a new message may be generated at each hop in the path and will pass through the whole path back to the origin.

- **Custody:** To provide reliability, a subset of the forwarding nodes as designated by NASA (you have no choice in these assignments) is labelled as "custody" nodes. They may be directly connected at times, but more often than not there will be non-custody nodes on the path between them. Custody nodes will take custody of a custody bundle before forwarding them while non-custody nodes will simply forward a custody bundle toward its next custody node. For custody to succeed both the source and destination of the original ADU must themselves be custody nodes. A custody bundle will pass through a series of custody nodes. At each custody node, the routing table will be consulted for the best next custody node on the path toward the final destination. Each custody node will store the bundle reliably (on persistent storage) at least until it has confirmation from a succeeding custody node that the full bundle (all fragments) has been received and stored reliably. The reason for this storage is that the custody node will have a timeout after which if it has not received a custody confirmation, it will retry by asking the routing table for a custody node (hopefully a different one) and sending again. When a custody bundle is being sent to the next custody node, both the current custody node address and the next custody node address must be updated (see the list of fields below) and any forwarding on that subpath from current to next custody node will use the next custody node address to determine next hop forwarding. Only custody nodes will confirm that they have taken full custody of a custody bundle. Although there is a simple, automated process for finding custody nodes and supporting custody, in order to allow you to improve performance of this approach, for your design you will be designing use of the explicit calls available at the end of this section to improve performance.

- **Storage:** It is possible for a node to receive more bundles than it can forward, so its memory may be exceeded. (Remember that each device has 64GB of memory.) In NASA's current implementation, bundles may simply be dropped when a node runs out of memory. Part of your job is to design a distributed storage system that will provide longer-term storage for bundles along their journey in case of either lack of communication options or simply congestion (more

bundles than can be handled) and possibly the ability to offload traffic from a fully-loaded bundle node to a less loaded node.

In our version of the protocol, each bundle can be assigned a priority level: 1 (no priority), 2 (mid-level priority), or 3 (high priority). The priority of a bundle impacts the order in which it is sent: at each bundle node, all priority 3 bundles from a particular source A to a particular source B will be sent first, then the priority 2 bundles from A to B, then the priority 1 bundles. However, the relationship between traffic on the A to B path and another flow of traffic from D to B is undefined. If the A-B and D-B paths have bundles at all three priority levels and their paths converge somewhere, the protocol says nothing about whether all the high priority bundles of both flows will be transmitted before all the medium priority bundles of both flows, and so forth.
To clarify this further consider that there are three FIFO queues between source A and destination B at each forwarding node N, one for each priority level. As bundles arrive at a node they are entered into the appropriate FIFO queue. In addition, for source D and destination B there are also three priority queues accumulating bundles. At node N, all of the A-B traffic at priority 3 will be transferred before any of the A-B at priority 2, and that all before priority 1. But, there is nothing to say that for the D-B traffic at priority 3, that it will be handled before the A-B priority 2 or priority 1. Only that the D-B traffic at priority 3 will be handled before the D-B priority 2 traffic. The handling of these queies is all automated and outside your contorl. Within each queue bundles are simply added to the queue as they are ready to be forwarded. This is important because if N is a custody node and it is considering a custody bundle, that bundle will only be ready to be sent once its fragments have been assembled.

With this definition of the protocol in mind, we can discuss what goes into a bundle.

2.2.3 Bundle Specification

Field name	**Size in bytes**	**Description**
Source address	16	IPv6 address
Destination address	16	IPv6 address
Reply-to address	16	The IPv6 address to which any responses or error messages should be sent
Current custody address (sometimes empty)	16	The IPv6 address of the node that currently has custody of the bundle and to which custody-transfer administrative bundles will be sent
Next custody address (sometimes empty)	16	The IPv6 address of the next custody node to which the custody bundle is being sent
Flags	4	Indicate various information; see Table 3
Time of creation	4	Given in seconds since Jan. 1, 1990.[6]
Expiration time	4	End time of usefulness, also in seconds since Jan. 1, 1990. Set to infinity (all 1's) if the bundle will never be out of date.
Bundle ID	4	
Fragment ID	4	
Fragment start point in ADU	4	

[6] 12 bits represent the week since the epoch began (Jan. 1, 1990) and 20 bits represent the second within a week.

Reserved for other fields	30	
Total	**134**	

Table 2: Fields in the primary block of a bundle

When an application determines that there is data to be sent, it collects that data into an Application Data Unit (ADU) of unconstrained size and calls the Bundle Protocol appropriately. The original bundle node uses that ADU to form a bundle, consisting of a set of blocks, each of which serves a different purpose, the primary block with transmission information, the data block containing the payload and some optional blocks that may have other useful information. The block types are described below.

- A **primary block** with transmission-critical information. This block is of a fixed size and contains the type of information we typically find in packet headers; Table 2 describes this information.

- A **payload block**. If the bundle is a *data* bundle, the payload contains all or part of the data of the ADU from the requesting application. If the bundle is an *administrative* bundle, the payload block contains the nature of the administrative message and any content relevant to it, such as the time and location of the report, etc.

- Zero or more **optional blocks**, which may provide "management" information for handling the bundle, such as the bundle age or a limit on the number of hops the bundle should experience. NASA has specified several optional blocks that can be included in a bundle. The possibly interesting ones from our perspective are:
  - Previous node: this identifies the node from which this bundle was directly forwarded.
  - Maximum hop-count: the maximum number of hops for this bundle. The hop-count is decremented at each node, and when the hop-count reaches zero, it is expected that the bundle will no longer be forwarded.

  Each of these optional blocks contains a tag indicating its type and 64 bits of value. You can decide whether these are useful to you and under what circumstances. You are also permitted to design your own (well justified) options. If that happens, you will need to make the case for them and explain any tradeoffs in each. Adding new options will require NASA to update their implementation of the protocol, so they are likely to be resistant to these changes without strong justification.

2.2.3.1 Flags

Bit #	Flag name	Results
0	Fragment	Bundle is a fragment
1	Administrative record	Bundle is an administrative record
2	No fragment	Bundle cannot be fragmented, it must be sent whole or rejected
3	Custody transfer request	Custody will only be transferred among custody nodes
5	Receive ack requested	Acknowledge receipt of bundle to previous hop node
6	Status time requested	Every status report will try to include the current time
7-8	Priority class of service	These will be one of high, medium, or low.
14	Bundle reception status report requested	Every node will attempt to send a "reporting node received the bundle" bundle for this bundle to the "Report-to" node

15	Request report of custody acceptance	A custody accepted bundle will be sent to the "Report-to" node, if custody is accepted
16	Bundle forwarding status report requested	Every node will attempt to send a "reporting node forwarded the bundle" bundle for this bundle to the "Report-to" node

Table 3: The optional flags available in the primary block of a bundle.

Now that the main aspects of the bundle model have been described, we can return to the set of flags shown in Table 2. NASA has specified a particular set of flags for use in the primary block, shown in Table 3. You may or may not find that you have uses for one any of them. If you need another type of flag to do something NASA did not think of, you may specify that, but you will need strong justification, as the protocol has already been standardized and implemented. The maximum number of additional flags you can add is four, because the other bits are either being used for other things or are being reserved for future extensions.

2.2.3.2 Fragments

As mentioned, a bundle can be fragmented if it cannot be transmitted in its current size. Administrative bundles can never be fragmented, but they also are quite small. If a bundle node needs to fragment a bundle, it will split the bundle data into two parts: two bundles each with a primary block, as well as an indication that each are fragments of a larger bundle. Your design will need to determine the size of the first fragment, but the BP itself will handle the fragmentation itself. The whole original bundle will be reconstituted by the BP bundle handling code when all the fragments rejoin at the same node (perhaps at the destination). If any fragment does not arrive or is corrupted, the bundle is lost. The current approach to simple local storage is used to hold bundle fragments if necessary until their remaining fragments arrive and all can be assembled. Note that since bundles may have expiration times, if a bundle is incomplete by its expiration time, the BP may decide to delete the fragments and not wait any longer.

2.2.3.3 Reporting

A bundle may have one or more flags set requesting reporting. All reporting is sent to the report_to address, an address designated by the originating sender to receive such reports. Choices for reports include:

- `Bundle_received`: this will report whether the whole bundle or a fragment (including the fragment ID) was received.
- `Bundle_forwarded`: this will report whether all or some of the bundle(some fragment(s)) were sent.
- `Bundle_custody` was accepted: T/F

In addition, if the `receive_ack_requested` flag is turned on, any node receiving that bundle should send an acknowledgement of that bundle to the preceding (sending) node.

When a report is formed, as many of these as are applicable at the time are formed into a single ADU to be sent in an administrative bundle. The intention is that these reports will be sent from every node at which they are applicable and have values to send to the same destination. That said, they are optional. If they are sent, they may not have any flags set including custody, so they will not in turn generate

further reports or support fragmentation. In your enhancement of forwarding, you can choose to use any set of these that suits your design.

2.3 Routing and Forwarding

As nodes receive bundles, they need to make a decision about where to send the bundle next; where to *forward* the bundle. The routing protocol implemented by NASA builds the *routing table* on each node. When a node receives a bundle, it looks up the bundle's destination address in its routing table and learns:

- The next possible hops for this destination. There may be more than one.
- The maximum number of bytes that can be guaranteed to be forwarded to that destination before it might become unavailable
- The two closest neighbors of the same type of device
- Custody forwarding information, which contains three facts: the address of the next custody node in a "custody" path toward the destination, the address of the next hop Bundle Node toward that next custody node, and the amount of connection time remaining for communication with the next hop Bundle Node.

NASA's routing protocol keeps this table updated. The updates are in a compressed form, and therefore quite small and incremental. They occur once a minute.

Though NASA is responsible for keeping the routing tables up to date, your design will dictate how each bundle is forwarded. There are a few basic cases to handle:

- In the simplest case of a bundle without custody, the bundle node should send the bundle to at least one of its next hops. Remember that as in the example in Figure 2 above, a bundle (or fragment of a bundle) can be sent along multiple paths. This may allow for faster or more reliable delivery at the cost of additional communications resources.
- If a report is expected, the report should be created and sent towards the report-to node, the designated receiver of reports, via its next hop(s) as specified in the routing table.
- If the bundle requires custody and the current node is not a custody node, the bundle should be forwarded, this time to the next hop towards the next custody node.
- If custody is required and the current node is a custody node, it must collect the whole original bundle (all the fragments) before the custody reply can be sent, so it will not consider the bundle "received" until the whole bundle has been reassembled. You can assume that for a custody bundle, nothing will be returned until all fragments have been assembled. At that point, the whole bundle is returned to the caller. The caller must then reply to the sending custody node before beginning the process of sending the bundle forward to the next custody node.

To remind you, as a consolidation of some ideas from above, at any point, it is possible that communications resources will limit the size of a bundle that can be moved forward through the network. The only options at that point are (1) to delay the bundle until more resources are available, (2) fragment the first part of the bundle if a flag has been set to allow this, send that much and reschedule the remainder, (3) drop the bundle if it has reached its expiration time or it will never be transmissible.

2.4 Storage

Forwarding raises several key questions for your system's design. For instance, if there is no next hop available, how should the bundle be stored? How should storage be organized locally to wait for a time when the bundle can be forwarded? Perhaps more importantly, would it be feasible to pass the bundle to a neighbor to store for perhaps improved forwarding? Would it be reasonable to duplicate the bundle elsewhere in the system to increase the probability of delivery? Part of your design will be a distributed network storage system that addresses these issues. You will need to design the decision-making procedure for deciding when a bundle will be moved or replicated and specify how those moves will happen (the coordination part of your distributed storage system). In the next section we include a set of calls to the existing Bundle Protocol and routing system; to support the enhanced Bundle forwarding system, you will need to define and describe the additional calls to take advantage of your enhancements.

2.5 Bundle Protocol Interface

With all this in mind, we can identify key calls provided by the BP. Notice that calls that involve sending a bundle do not return any values. Since communication between nodes is via a single bundle at a time, a bundle that might reflect a call to the remote node is simply delivered to that node, and that node, in response, hopefully will send a "responding" bundle which can be received through the `receive_bundle` call. In contrast, anything that is local, like creating a bundle or querying the local routing table, will return values.

Function call	**Returns**
`create_bundle(ADU, bundle_type, destination_addr, report_to_addr, expiration_time, fragmentation_flag, custody_flag, report_custody_transfer_flag, status_report_bundle_received_flag, status_report_bundle_forwarded_flag, receive_ack_requested_flag, next_custody_node, priority, hopcount)`	`bundle`

- `ADU` contains application data or any relevant payload for an administrative bundle.
- `bundle_type` must be "administrative" or "data"
- The destination address will be the final destination of the bundle.
- The report_to_addr is the address to which to send reports.
- The expiration time will be in absolute time as specified in Table 2.
- Each of the flags will be T/F.
- If the `custody_flag` is T, the `next_custody_node` will be the IP address of the next custody node. If the `custody_flag` is F, `next_custody_node` will be set to be empty.
- The priority will be set to 1, 2, or 3.
- If `hopcount` is a positive integer, include the optional hop count block. If it is zero, do not. A negative number is an error.

- If the `receive_ack_requested` is true, include the IP address of the current node in an optional `prior_node` block.

`unpack_bundle(bundle)`	`ADU, source`

- Note that this call cannot be completed until the whole original bundle has been received

`update_next_custody_address(bundle, next_custody_address)`	`updated_bundle`

`update_current_sending_node(bundle)`	`updated_bundle`

- If the flag has been turned on to include the prior node address (i.e. the current node that is forwarding to the "next" node) then update the optional block that contains that address.

`bundle_available()`	`True/False`

`send(bundle, next_node)`	

`receive_bundle()`	`bundle`

- For the present this function will simply wait for incoming data and return the data and source of that data. Depending on the type of the received bundle you will need to do one or more of the following. If it is an administrative bundle and is simply reporting, the information is noted or forwarded as needed. If it is an administrative bundle related to custody, if custody transfer has failed, you must try again to send perhaps by a different path, unless it has expired. If it succeeds, you can delete the bundle locally. If the bundle is a data bundle, after figuring out the appropriate next step, arrange for it to be queued for forwarding. For all administrative bundles, the bundle will also be returned to the caller. In all cases, it where it is a data bundle, it will simply be returned to the caller for any further decision-making about forwarding, fragmenting, etc.

`next_hop(destination)`	`next_node, remaining_capacity`

- `next_node` is an IP address and `remaining_capacity` is how many bytes can be guaranteed by the routing system to be send-able.

`next_custody_node(destination_addr)`	`next_custody_node, next_forwarding_node, remaining_capacity`

- `next_custody_node` and `next_node` are IP addresses and `remaining capacity` is how many bytes can be guaranteed by the routing system to be send-able to the next node)

`fragment_bundle (bundle, size_of_initial_fragment)`	`fragment_bundle, remaining_bundle`

- `fragment_bundle` and `remaining_bundle` include the primary block with all fields set correctly to continue transmission

`get_neighbors()`	`list_of_neighbors`

- This uses the node's own address to query the routing system. `list_of_neighbors` includes all types of neighbors available, each is element in the list is a tuple of IP address, type (satellite type, relay, antenna), and remaining capacity. This capacity is the amount in bytes that can be reliably (modulo dropped traffic) to this neighbor before it is out of range. It is based on a variety of kinds of information including relative trajectories and bandwidth available between the two nodes.

`query_neighbor(bundle_size, neighbor)`	

- Sends a message to the neighbor for remote storage available
- The node sending this is requesting at least `bundle_size` amount of storage

`storage_available(storage_available, expiration time for that availability, neighbor):`	

- This is the response of a neighbor to the previous `query_neighbor (bundle_size, neighbor`).
- The node sending this response commits to the specified amount of disk storage being available to the neighbor who requested it (the "neighbor" in this bundle)

3 Your challenge

Your overall challenge is to design an enhanced bundle transport system that allows for improved reliability and decreased delivery times as required by some of the applications using the system, while not incurring significant overheads in bandwidth, storage utilization, and delivery disruptions for normal traffic.

To do this, you will be designing a distributed network storage system, improved forwarding to take advantage of the distributed network storage system, and a set of calls for access to this "enhanced" Bundle Protocol environment. The motivators for these enhancements are a set of use cases to be supported by the system. Each of these is constrained by system limitations and tradeoffs.

It is important to remember that there is no absolute correct answer here. All design decisions are likely to have tradeoffs and repercussions. Part of your task is to recognize, evaluate, and justify the tradeoffs that you select.

3.1 Data types to support

The primary challenge to your design derives from the needs of the applications. These may include urgency of delivery, tolerance to incomplete delivery of all ADUs, and other delivery expectations. Your system must support the following three types of data:

- **Routine management data** is any data used to manage SolarNet and other NASA systems. It may involve specific commands for relocation, directional adjustments to focus on particular data, commands to rovers, etc. This traffic is relatively low volume and, most of the time, non-critical. Management data must be correct and complete (i.e., applications that use this data will not tolerate incorrect or missing data). On rare occasions, management data will become mission critical, as in the case of impending solar flares (see Section 4.2).
- **Telemetry data** is data that is collected from sensors. Currently, the primary telemetry is from sensors collecting data for the National Oceanographic and Atmospheric Administration (NOAA). This data is collected on the GEOComs and includes both ground-focused data – such as visible light and infrared images of the Earth – and solar information, used in part to detect impending solar flares. Some of this data may be tolerant to lapses and delays, while others will have significant urgency to them. The volume of this type of data is extremely high.
- **System maintenance data and updates** consist primarily of software updates. These must be delivered completely, correctly, and to all destinations, and then be guaranteed that the updates were installed correctly. These data sets may be quite large.

In Section 4.2, we describe use cases that generate each type of data.

3.2 The Distributed Network Storage system

The "network storage system" refers to the network storage available on each bundle node and any communications necessary for coordination and cooperation among bundle nodes with respect to operating this shared system. Part of your task is to design both the individual nodes and coordination management among the Bundle Nodes' storage systems, to provide a distributed network storage that will improve performance over a system with no such distributed coordination.

Within a single node, storage will need to be organized and managed. There are two kinds of influences on your design: the size of the storage available on the node and input/output limitations of the individual node, and transport policies (the degree of urgency for the traffic, the tolerance to lost information, etc.). We will discuss each of these separately.[7]

3.2.1 Storage Constraints

The most basic constraint on the network storage system is the size of storage on each node. Each satellite and relay will have only .5TB of storage. Your storage system will need to store and retrieve bundles as efficiently as possible. You will need to make decisions about, e.g., how to handle duplicate bundles, how a node decides which bundle to forward next, what to do if network storage is full and new traffic arrives, etc.

[7] In such systems, power utilization is a critical characteristic. For purposes of this design project we set that aside, because the primary power consumption will derive from driving the lasers and that is outside the scope of the project.

As mentioned above, bundles may have one of three priorities, with higher priority bundles between a single source and destination being delivered before lower priority ones. The bundles are appended to the appropriate queue when they are ready. In addition, the queue management for each source-destination pair is independent of the other pairs, as discussed in the protocol explanation above.

3.2.2 Communications limitations

The second constraint on this system is the combination of the bandwidth between the communicating nodes of interest and the amount of time when they are in direct communication. These will determine how much traffic can arrive and how much can be moved either onwards or to a collaborating neighbor. The closer nodes are to Earth, the higher the bandwidth they support.

The amount of time when a pair of nodes is in communication is available from the routing table. Each node's routing table will reflect the remaining amount of transmission time to each neighbor of a different sort than itself. Notice that their neighbors of the same sort as themselves never timeout; they are always adjacent to the same neighbors of the same sort.

3.3 Enhanced forwarding

Given your distributed storage system, an important feature of your system will be a decision-making process for forwarding bundles or fragments of bundles. It is likely that the amount of time remaining in communicating between a current node and its next hop, along with information about each of its neighbors, can improve either the situation for a particular bundle or the overall performance of your system. In addition, it may allow you to address situations in which a "next hop" is temporarily unavailable. For this you will need to determine what additional information might be needed to make these decisions and how to acquire that information.

3.4 The Interface to Your Enhancements.

NASA wants a uniform approach for applications to invoke the protocol to hand off their ADUs (application data units) to the BP with all the appropriate flags set to provide the particular choice of transport service required by each application at the time at which it is requesting a transfer. Part of your responsibility is to add to that the additional calls to enable taking advantage of your network storage system and enhanced forwarding decision-making.

It is valuable to note that all the things the BP is specified to do, such as determining exactly where and when to forward a bundle, sending reports, custody handoff, and reassembling a bundle from a set of fragments, are handled by the BP itself. The application should never be involved in that sort of activity. The applications only send and receive whole ADUs with the desired attributes on that BP transmission of the bundle containing the ADU, so the calls you define will be used internally to the enhanced BP.

4 Design criteria and evaluation

As you develop your system design, you should consider how you will evaluate it; evaluation is critical in convincing your readers that your design is acceptable. Below, we describe some basic evaluation metrics you should consider, as well as three specific use cases your design should handle.

4.1 Evaluation metrics

There are several metrics you might consider; this is not an exhaustive list.

- Network overhead: how much bandwidth beyond the actual transmission of bundles your design will require. Overhead may be affected by lateral movement among satellites at the same altitude in order to achieve some performance tradeoff, as well as any reporting you may require.
- Times to completion of delivery of bundles
- The tradeoffs between time and space of managing the storage your organization of the local bundle node storage capability
- The costs (bandwidth and time) of your design choice for how to handle the handing off of a bundle to a neighbor satellite.
- Increase in reliability of delivery of bundles using your coordinated network storage facility. This may be in terms of time, bandwidth, or other metrics.

4.2 Use cases

An important aspect of evaluation a system is the consideration of how effectively that system can serve particular use cases, and how those use cases may impact general usage. In Section 3.1 above we discussed four specific types of uses of the system: general communications for humans; land satellite observations, especially for detecting and predicting land and mud slides in remote areas; solar storm predictions; and distribution of software updates to components of the system. These use cases may be both drivers for your system design and, in the end, a means of evaluating the efficacy of your design.

a) Routine communications: Assume there are two people resident on the Moon in different places, as well as two people resident on Mars in different locations. On a regular but not continuous basis, Ground Control wants to check in with them, send them task responsibilities, send and receive email between them and family, and so forth. 5% of the time there will be simultaneous messages between these people and Ground Control. The messages will range in size from 1KB to 10MB (e.g., a video between family members). For email and other personal messages there are no delivery time constraints as long as they arrive in a reasonable time-frame, but for task-related messages, there is a requirement that they arrive within 10 minutes of their expected time. It is important that at the application layer (email, video streaming, wake-up calls, etc.) all messages be received completely. The average rate of the traffic between each person and Ground Control is 1KB per minute, but it may be bursty. There will be very rare occasions (such as the Solar Flare situation in Use Case c)) when some of these small messages will become extremely critical and will require both as fast and as reliable delivery as possible for human safety.
b) Land Satellite telemetry: This data is being collected by satellites similar to Sentinel-1. Our version has a few improvements and simplifications over that. There are two satellites, each in a Solar-synchronous orbit at 693 km above Earth. In a 12-day cycle each satellite returns to about the same spot, Solar-synchronously (which means the path from Earth to Sun is identical and adjusted for the Earth's rotation around the Sun). This is a polar orbit. Since there are two satellites, each on a 12-day cycle, this orbit means that one of them reaches the same spot in the cycle every six days. It also means that the angle of its lighting from the Sun is always the same at that point in the cycle.

The telescope on each satellite has two features. First, it allows for an accurate measure of altitude of the Earth; as the ground becomes increasing likely to slide (landslide or mudslide) its altitude changes. Second, it is a "large swath" telescope with an over 300 km. wide image. The swaths are overlapping, so a particular spot will be in the images from 3 successive revolutions of each satellite. The system also includes two ground stations, one in Northern Scotland and one in eastern Australia. Each satellite has enough storage to hold all its collected data until it comes into contact with one of the two ground stations, but that may take some days.

For our use case we are interested in Kyrgyzstan.[8] The country has enormous regions that are very sparsely occupied and difficult to reach. Satellite imagery is the only means of learning about imminent dangers to the population in the very steep and rocky terrain, so the government is interested in predicting land and mudslides, to notify the population with at least 10 minutes, to move out of the way or to higher ground. There are movements of the land that are used for such predictions. Some evidence can be seen up to 10 days before (for some kinds of earth movement).

Each satellite is collecting many kinds of data, not only the data for predicting dangerous earth movement, so only a small subset of the total data collected is needed for this prediction. The satellites can do limited processing on the data to decide whether there is some probability of such an event, using that to determine when they need to expedite sending the data to Earth for more detailed analysis. These events (predictors of possible slides) occur on average once every 2 cycles for each satellite (so once every 24 days).

In addition to the satellite system's own ground stations, in the case that there is evidence of a possible landslide, the satellites can also communicate with the GEOcoms or LEOComs, to transfer a set of 20 files, each of which is 2GB. This situation is considered exceptional and should not be utilized unless necessary. Once on the ground, the analysis and alarm notification takes 10 min. You have no access to the code that is making the determination about whether it is urgent to send the data. Your responsibility is to support the sending of the data quickly and correctly if it is determined that it should be sent.

c) Solar telemetry: For Solar storms, "solar loops" provide an early predictor a few hours before an actual Solar storm. To collect this data, satellites must be high above the Earth, so they are in geostationary orbits. For our scenario there are two of them. They are collecting many different kinds of data, but the data required for analyzing and predicting Solar storms is in the form of a 200MB file, once every minute. Twenty such datapoints in succession are enough for a high-confidence prediction. The probes do enough onboard processing to determine when these files should be transmitted both expeditiously and accurately. These predictions of a possible solar storm occur on average once a month. The solar loops provide a 5-hour warning window, once analyzed on Earth. Again, the determination of whether a transmission needs high priority is out of your hands. It is your responsibility to respond effectively to such high priority determinations.

   These satellites are in the same geosynchronous orbit as the GEOComs, as discussed in Section 2.1. They can communicate directly with both GEOComs and LEOComs in their neighborhoods.

[8] This is in part based on conversations I've had with researcher in Kyrgyzstan with exactly our concern.

d) Large transmissions requiring guaranteed correct delivery: On a fairly frequent basis, software updates will be needed on all of the different types of devices: LEOComs, GEOComs, relays and the facilities on the Moon and Mars. For each software update, it is critical that every target device receive and, within 30 minutes, initiate the update process. In addition, each device must confirm correct installation of the updates. The maximum sizes and frequencies of these are:

Destination	**Size**	**Frequency**
LEOComs	4GB	1/month
GEOComs	16GB	2/month
Relays	1GB	Every 2 months
Moon and Mars equipment	100GB (in 10 10GB files)	1/month

Table 4: Software update sizes and frequencies