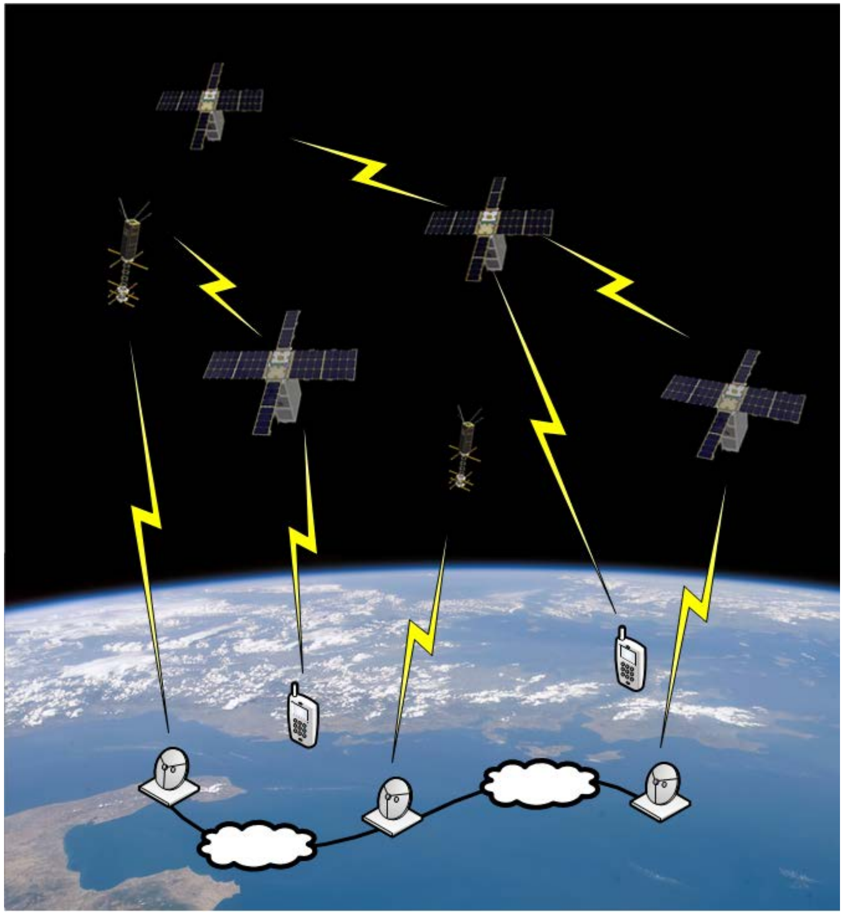
14th Annual CubeSat Developer's Workshop
April 26-28, 2017
San Luis Obispo, CA

A Testbed for Demonstration and Performance Analysis of an Autonomous Scheduling System for Communications Nanosatellites

Peter J. Yoo
Alonzo E. Jenkins
Cherry Y. Wakayama
Mohammad R. Alam

SPAWAR Systems Center Pacific
San Diego, CA

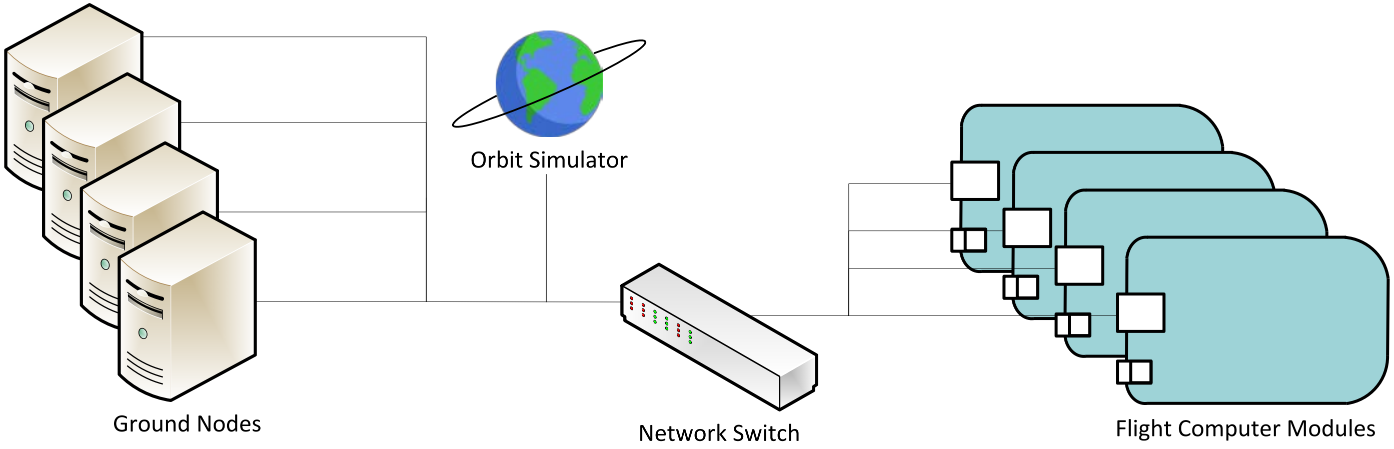
Need: Mission-level Hardware testbed



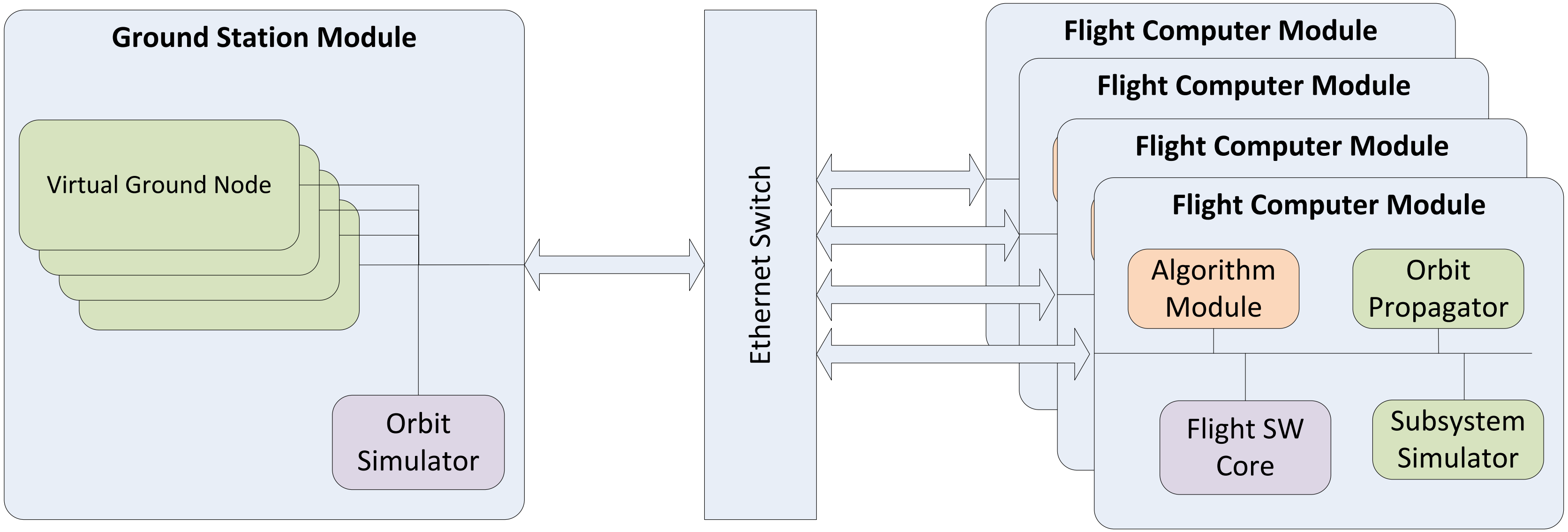
- Software-in-the-loop (SITL) modeling and simulation has limitations
- No low-cost Hardware-in-the-loop (HITL) M&S tool to test/simulate mission scenarios
- Flatsats / ground testbeds can only simulate individual nanosatellites

Proposed Solution

- Create a low-cost, mission-level simulation environment using Commercial-Off-The-Shelf (COTS) hardware
  - Hardware-In-The-Loop (HITL) nanosatellite constellation testbed
  - Open source, industry standards and existing flight software
  - Graphical User Interface (GUI) for input parameters and analysis output
- Demonstrate autonomous algorithms using the testbed



Nanosat Communications Constellation Testbed Architecture

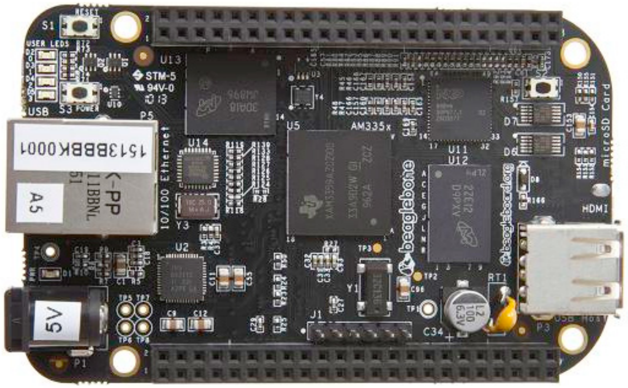


- **Virtual Ground Node** – Simulates ground stations in different geographic areas and hosts GUI and module for automation and optimization algorithms
- **Orbit Simulator** – Provides orbit information for a given mission
- **Flight SW Core** – Provide services for sending commands / receiving telemetry
- **Orbit Propagator** – propagate position and velocity of a satellite
- **Subsystem Simulator** – Simulates battery, solar panel, attitude control, gps receiver
- **Algorithm Module** – User-defined automation and optimization algorithms

Nanosat Communications Constellation Testbed Design

▼ Hardware Implementation

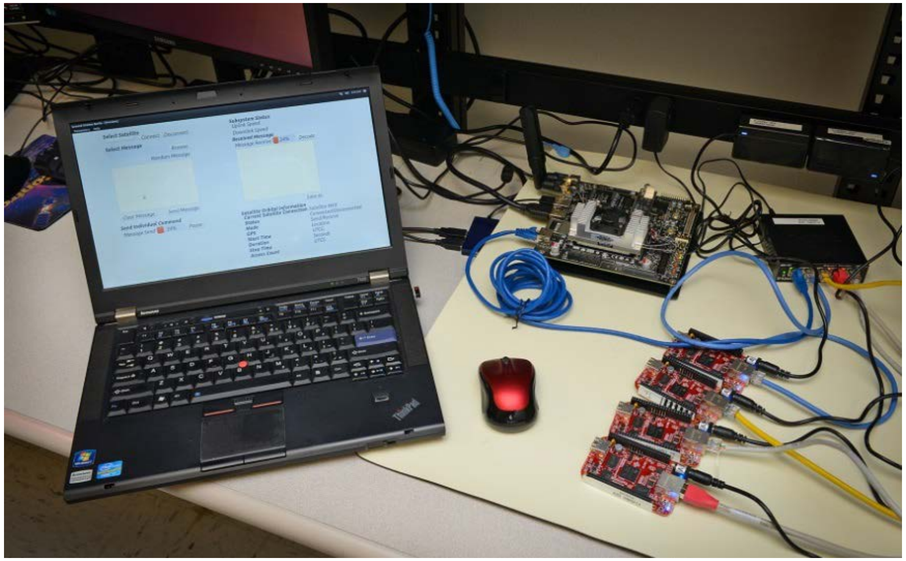
- 1 x Workstation
- 4 x BeagleBone Black (BBB)
- 1 x Ethernet switch (8-port)



*https://beagleboard.org

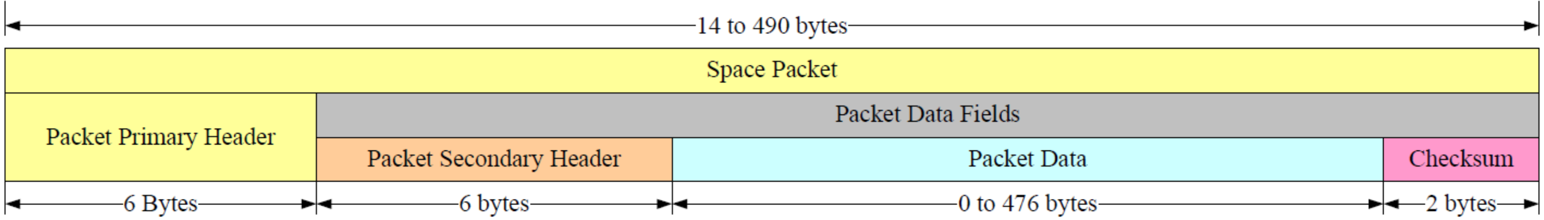
▼ Software Implementation

- Flight Software developed for DoD 6U nanosat bus
- VMware with Ubuntu
- Python IDE



Flight SW Core Description

- Testbed uses actual flight software
- Uses Consultative Committee for Space Data Systems (CCSDS) compliant message structure called “Space Packet” for all telemetry, commands, interface control requests and acknowledgements


14 to 490 bytes

Space Packet

Packet Primary Header

Packet Data Fields

Packet Secondary Header

Packet Data

Checksum

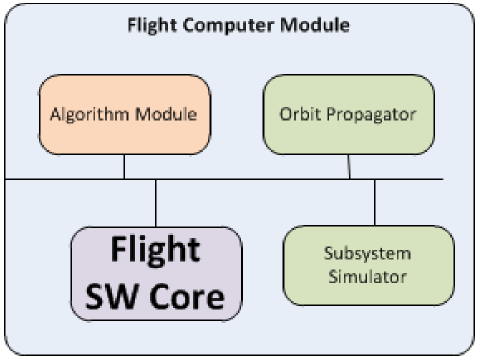
6 Bytes

6 bytes

0 to 476 bytes

2 bytes

- IP-based/Ethernet interface between components including ground stations
- 9 Services* available between bus and payload including
  - Payload Command Forwarding
  - Payload Telemetry
  - Bus Command
  - Bus Telemetry Packet
  - Bus Telemetry Stream
  - Payload Data Storage
  - Payload Data Downlink
  - Payload Data Load
  - Time Service


Flight Computer Module

Algorithm Module

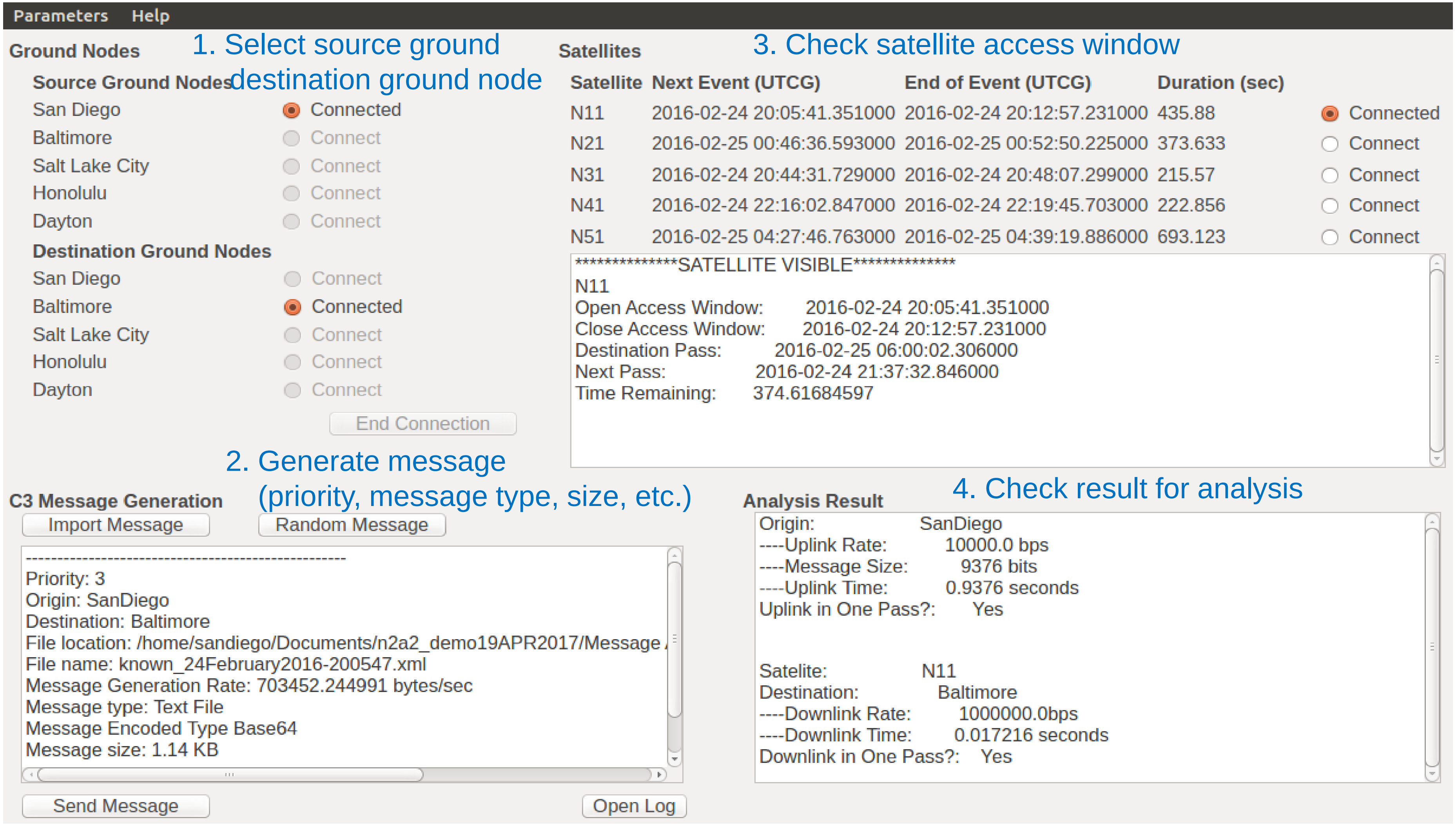
Orbit Propagator

Flight SW Core

Subsystem Simulator

* SN-BPLICD-001 Rev 1.0 by Naval Research Laboratory

Scenario Simulation: Graphical User Interface Design Example



1. Select source ground destination ground node

2. Generate message (priority, message type, size, etc.)

3. Check satellite access window

4. Check result for analysis

Ground Nodes

Source Ground Nodes

San Diego	● Connected
Baltimore	○ Connect
Salt Lake City	○ Connect
Honolulu	○ Connect
Dayton	○ Connect

Destination Ground Nodes

San Diego	○ Connect
Baltimore	● Connected
Salt Lake City	○ Connect
Honolulu	○ Connect
Dayton	○ Connect

Satellites

Satellite	Next Event (UTCG)	End of Event (UTCG)	Duration (sec)	
N11	2016-02-24 20:05:41.351000	2016-02-24 20:12:57.231000	435.88	● Connected
N21	2016-02-25 00:46:36.593000	2016-02-25 00:52:50.225000	373.633	○ Connect
N31	2016-02-24 20:44:31.729000	2016-02-24 20:48:07.299000	215.57	○ Connect
N41	2016-02-24 22:16:02.847000	2016-02-24 22:19:45.703000	222.856	○ Connect
N51	2016-02-25 04:27:46.763000	2016-02-25 04:39:19.886000	693.123	○ Connect

```
*************SATELLITE VISIBLE*************
N11
Open Access Window:      2016-02-24 20:05:41.351000
Close Access Window:     2016-02-24 20:12:57.231000
Destination Pass:        2016-02-25 06:00:02.306000
Next Pass:               2016-02-24 21:37:32.846000
Time Remaining:          374.61684597
```

C3 Message Generation

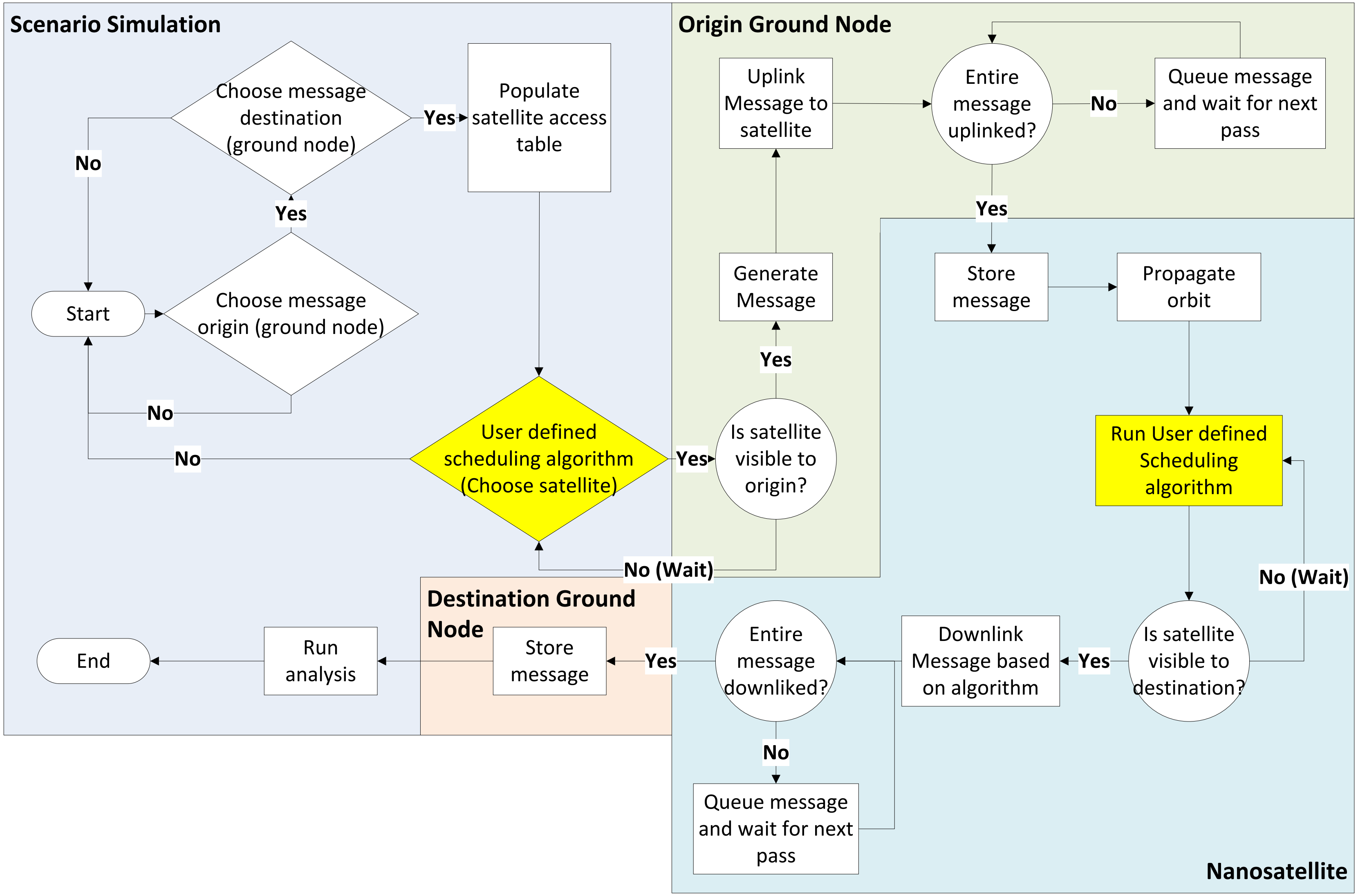
```
--------------------------------------------------
Priority: 3
Origin: SanDiego
Destination: Baltimore
File location: /home/sandiego/Documents/n2a2_demo19APR2017/Message
File name: known_24February2016-200547.xml
Message Generation Rate: 703452.244991 bytes/sec
Message type: Text File
Message Encoded Type Base64
Message size: 1.14 KB
```

Analysis Result

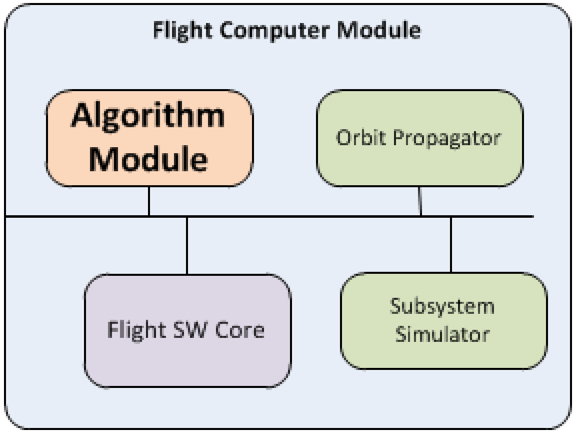
```
Origin:                  SanDiego
----Uplink Rate:            10000.0 bps
----Message Size:             9376 bits
----Uplink Time:            0.9376 seconds
Uplink in One Pass?:          Yes


Satelite:                N11
Destination:                Baltimore
----Downlink Rate:            1000000.0bps
----Downlink Time:          0.017216 seconds
Downlink in One Pass?:      Yes
```

Nanosat Communications Constellation Testbed Message flow


Scenario Simulation
Choose message destination (ground node)
Yes
Populate satellite access table
No
Yes
Start
Choose message origin (ground node)
No
No
User defined scheduling algorithm (Choose satellite)
Origin Ground Node
Uplink Message to satellite
Entire message uplinked?
No
Queue message and wait for next pass
Yes
Generate Message
Yes
Yes
Is satellite visible to origin?
No (Wait)
Store message
Propagate orbit
Run User defined Scheduling algorithm
No (Wait)
Destination Ground Node
End
Run analysis
Store message
Yes
Entire message downliked?
Downlink Message based on algorithm
Yes
Is satellite visible to destination?
No
Queue message and wait for next pass
Nanosatellite

Interface to Algorithm Module



- Software wrapper allows rapid implementation of user-defined scheduling algorithms
  - Provides receive and send capability between ground nodes and nanosatellites
  - Telemetry data is available for algorithm to use during decision making
- Algorithm Module provides pathway to implement optimization in the flight computer
  - Communication scheduling
  - Other event scheduling

Summary and Ongoing Work

Summary

- Developed mission-level nanosatellite communications constellation testbed framework using COTS components
- Defined virtual ground node and flight computer module interfaces
- Developed a GUI design to help modeling and simulation
- Developed message delivery simulation test cases for evaluation

Ongoing Work

- Model a nanosat subsystem simulation module (battery, solar panel and etc.)
- Implement energy-cognizant nanosat message delivery scheduling system
- Implement MC3 ground station network model and simulation
- Demonstrate autonomous message delivery scheduling systems for a store-and-forward nanosat communication constellation using the testbed

THANK YOU