Lights Out: Evolution of an Automated Ground Segment for Operation of the Aerospace CubeSat Constellation

Christopher Coffman
Darren Rowen
Joseph Gangestad
Brian Hardy

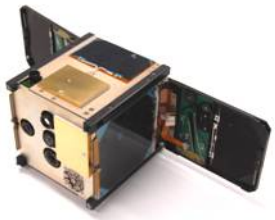
The Aerospace Corporation
Aug 9, 2015

Presentation Outline

- Background
  - *The Constellation and Ground Station Network*
  - *Historical Setup*
  - *Goals/Requirements*
- Initial Evolution
- Automation
- System Architecture
- Automation Examples

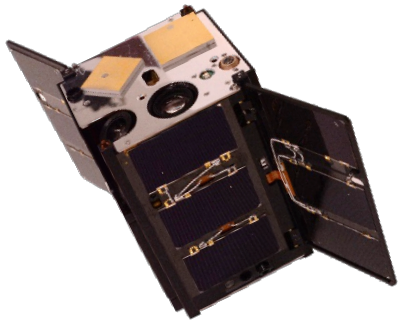
Background

The Constellation

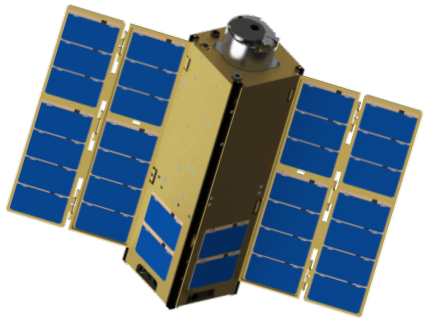


AeroCube-4.0 (1)
AeroCube-4.5 (2)

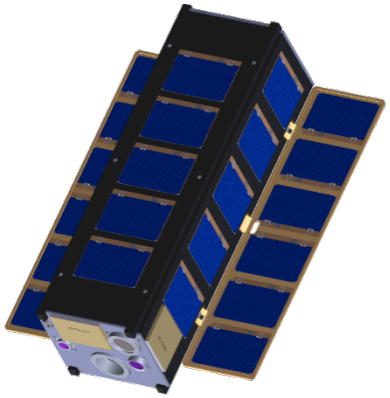
AeroCube-5 (2)



AeroCube-7A (1)

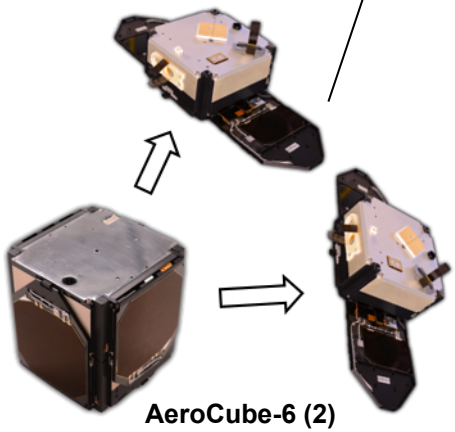


AeroCube-9 (1)

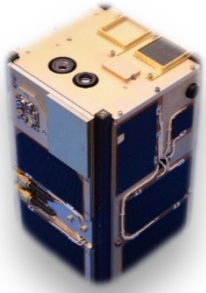


R^3 - TOMSat (2)

2012 2013 2014 2015 2016 2017

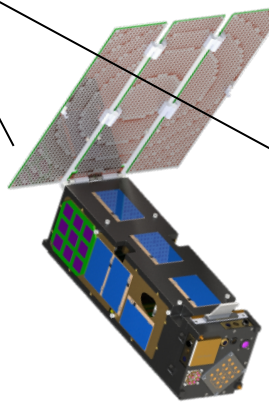


AeroCube-6 (2)



AeroCube-8 (2)

AeroCube-5 (1)

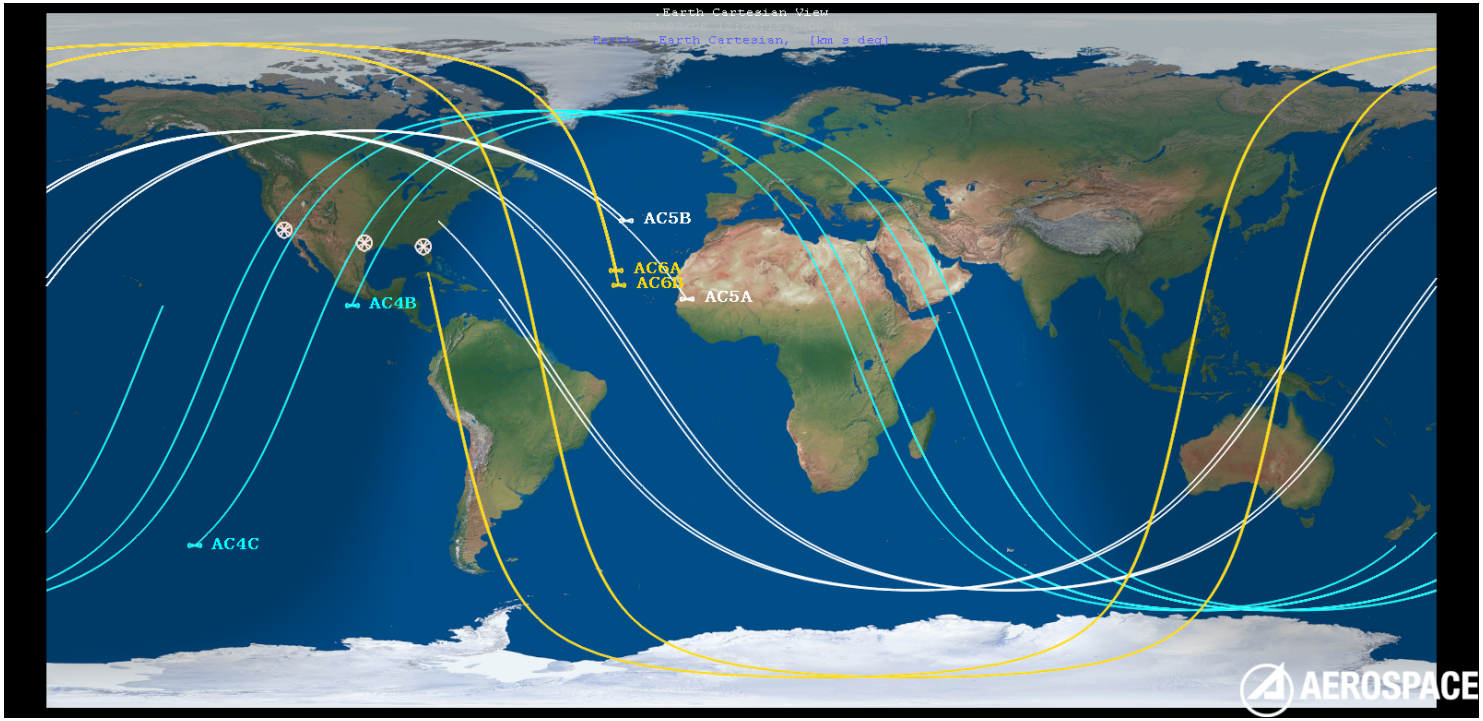


ISARA (1)

AeroCube-7 B&C (2)

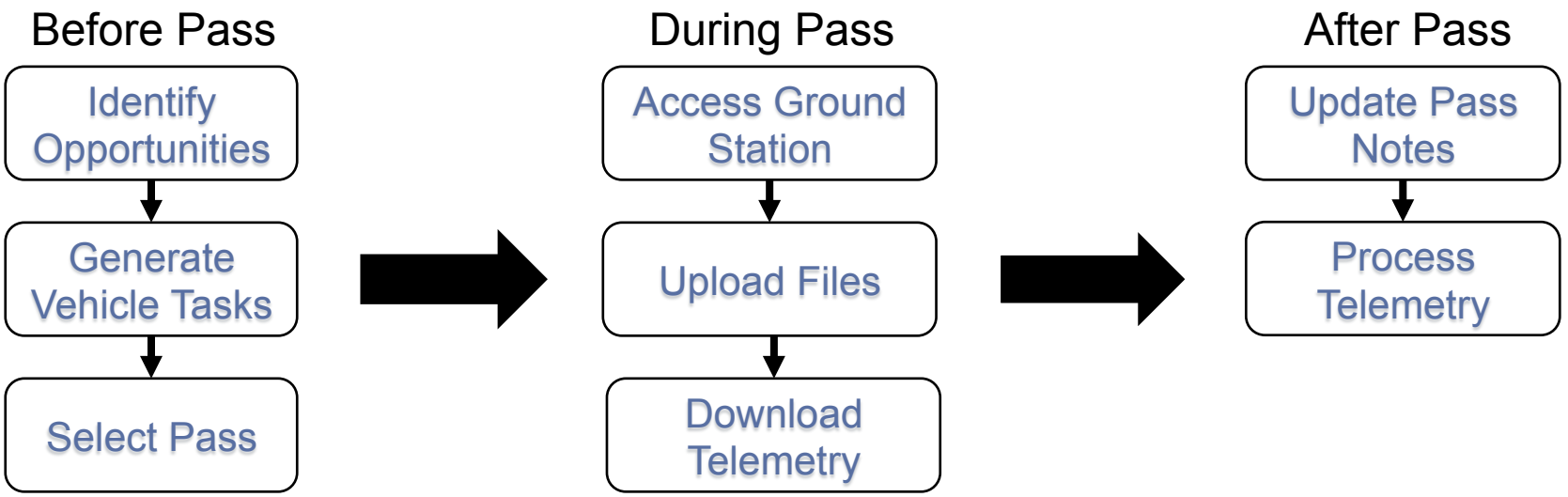
Background
The Ground Network

- Ground Stations
  - *Texas*
  - *Florida*
  - *California*
- Features
  - *Vehicle Independent*



Background

Mission Flow



Background
Historic Operations - Tools

Software Key
- MATLAB
- VB6
- Excel

- Tools
  - *Mission Planning Software*
    - Generates vehicle tasking
  - *Ground Station Control Software*
    - Uploads tasking
    - Allows direct vehicle commanding
  - *Pass Management*
    - Pass data manually logged
  - *Telemetry Handling*
    - Parser Scripts
      - *Time Tagged Files*
    - Data compiled in Excel as needed

- Problems
  - *Labor Intensive*
    - Numerous repetitive tasks
    - Idle vehicles
      - *Still need to collect health telemetry*
  - *Coordination is Difficult*
  - *Data Overload*
    - Telemetry is not readily accessible
    - Hard to find and identify specific events
  - *Multiple Code Bases*
  - ***Constellation is Growing!!!***

System Evolution
The Goal

- Driving Goal
  - ***Reduce The Required Labor to Maintain the Constellation***
- How
  - *Automate Ground Stations*
  - *Simplify Coordination*
  - *Automate Telemetry Collection and Aggregation*
  - *Automate Basic Fault Detection*
- Requirements
  - *Do not interfere with user planned missions*
  - *Interact with existing software*

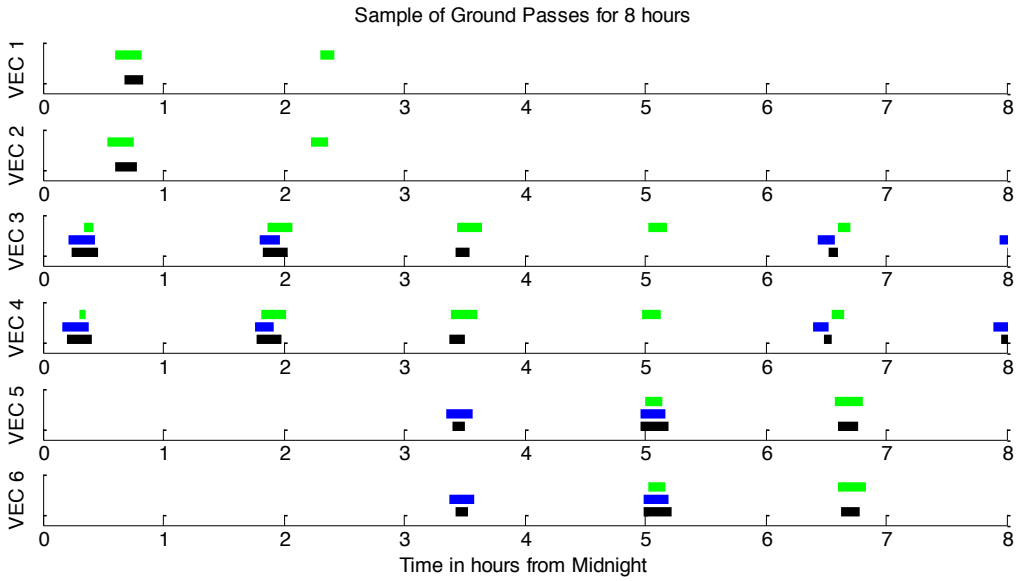
System Evolution
First Steps

- Step 1 – Scripted Ground Stations (2012)
  - *Socket Based Program*
    - Server - Powers client per schedule and sends command file
    - Client - Executes command files and returns telemetry to server
- Step 2 – Pass Management (March 2014)
  - *Database driven website*
    - Provides automatic handling for simultaneous users
    - Database acts as central repository for other programs
- Step 3 – Telemetry Handling (July 2014)
  - *Automated telemetry binary data parsing*
  - *Aggregate in database*

System Evolution

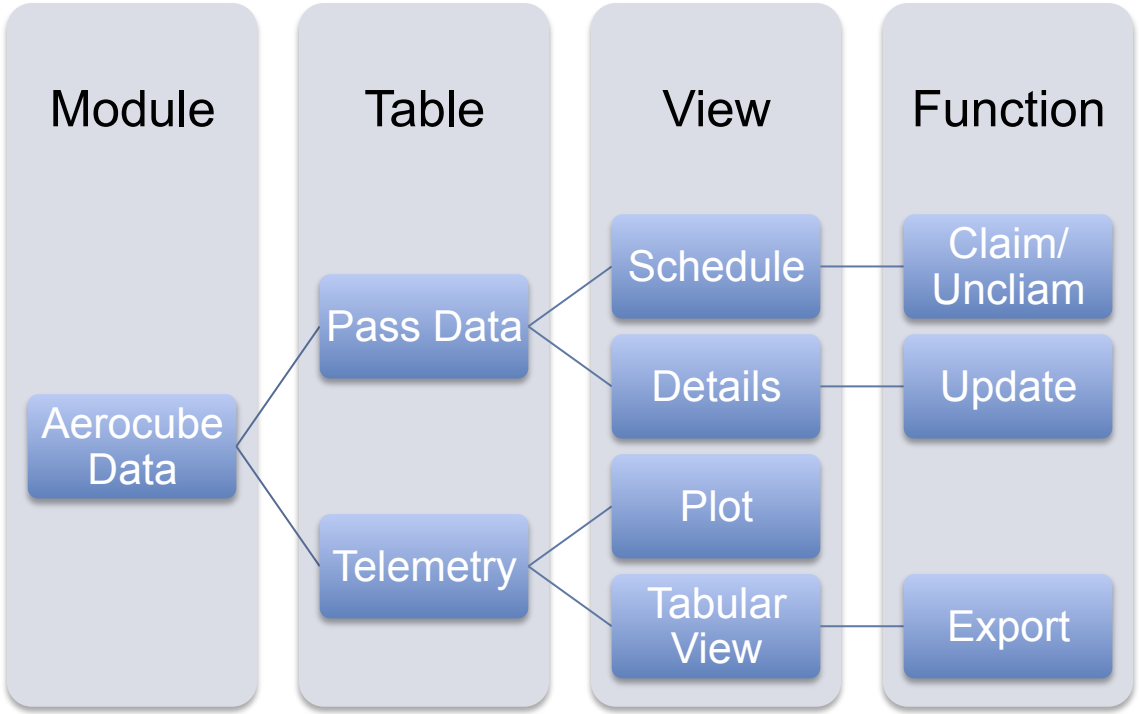
Notes on Pass Management

- Import Predictions to Database
- Identify Conflicts
  - *Considerations*
    - Ground Station Restrictions
    - Vehicle Restrictions
    - Timing!!
- Store Conflict Data
- Management Interface
  - *Claiming pass locks out conflicting passes*
  - *Allows super users to bypass conflict management*



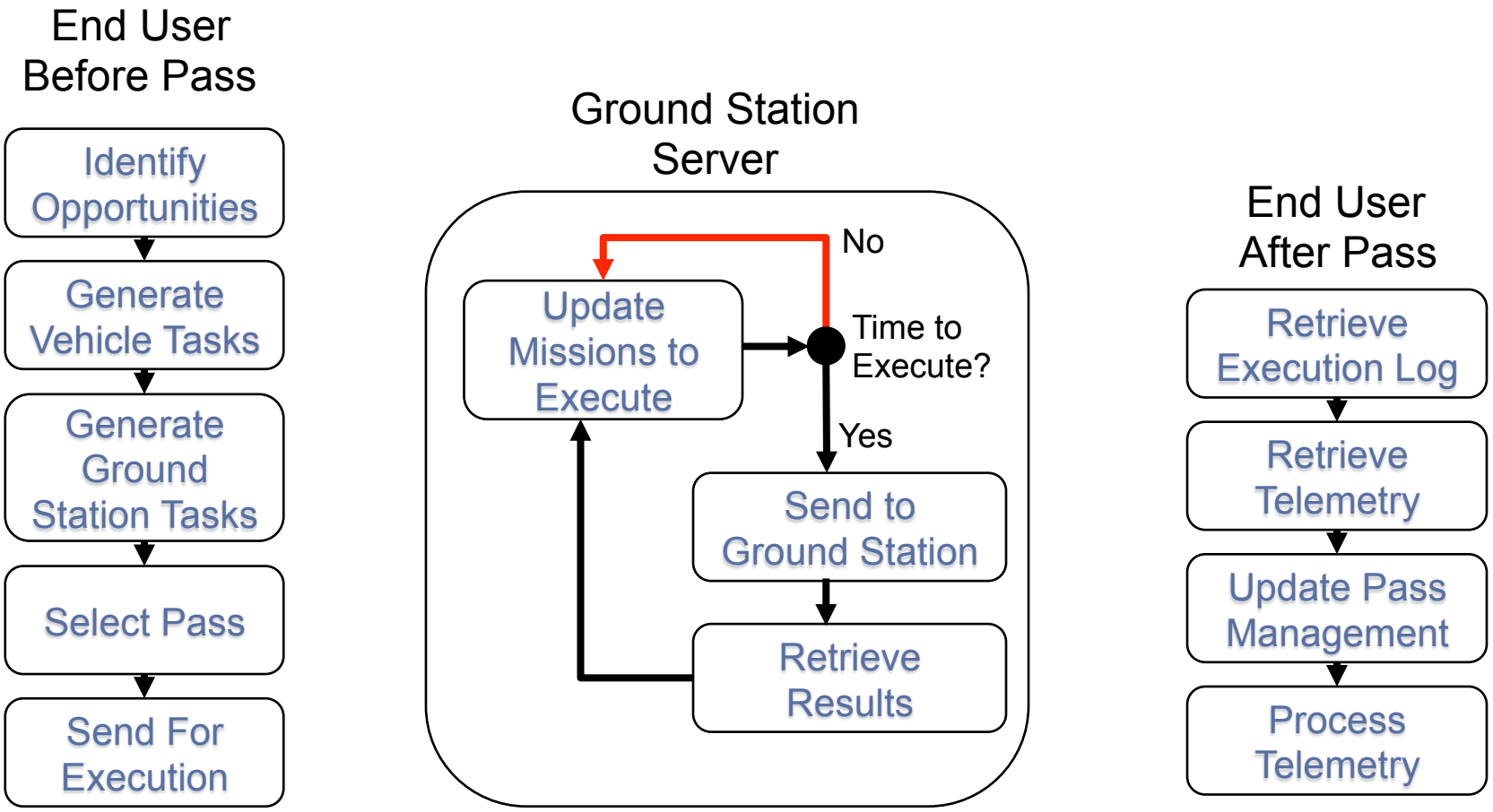
System Evolution

Website Architecture



- Website Advantages
  - *Platform Independent*
  - *Simple Access Control*
  - *Elegant Visualization Tools*
- Django Platform
  - *Python Backend*
  - *HTML, CSS, Javascript Frontend*
  - *Advantages*
    - Python Based
    - Template System
    - Module Based

System Evolution
New Mission Flow



System Evolution
Introducing Automation

- Step 4 – Basic Automation (August-October 2014)
  - *Collect Basic Health Telemetry Automatically*
    - Generate simple vehicle and ground station tasking
    - Generate vehicle priorities
  - *Ground Pass Tracking*
    - Parse results from ground pass server and import to the database
  - *Vehicle State Tracking*
- Step 5 – Contextual Notifications (January 2015)
  - *Email Alerts*
    - Automation Failures
    - Vehicle Health Alerts
- Step 6 – Advanced Automation (May 2015)
  - *Complex Telemetry Collection*
  - *Vehicle Specific Automation*

Constellation Automation

Deciding What to Do

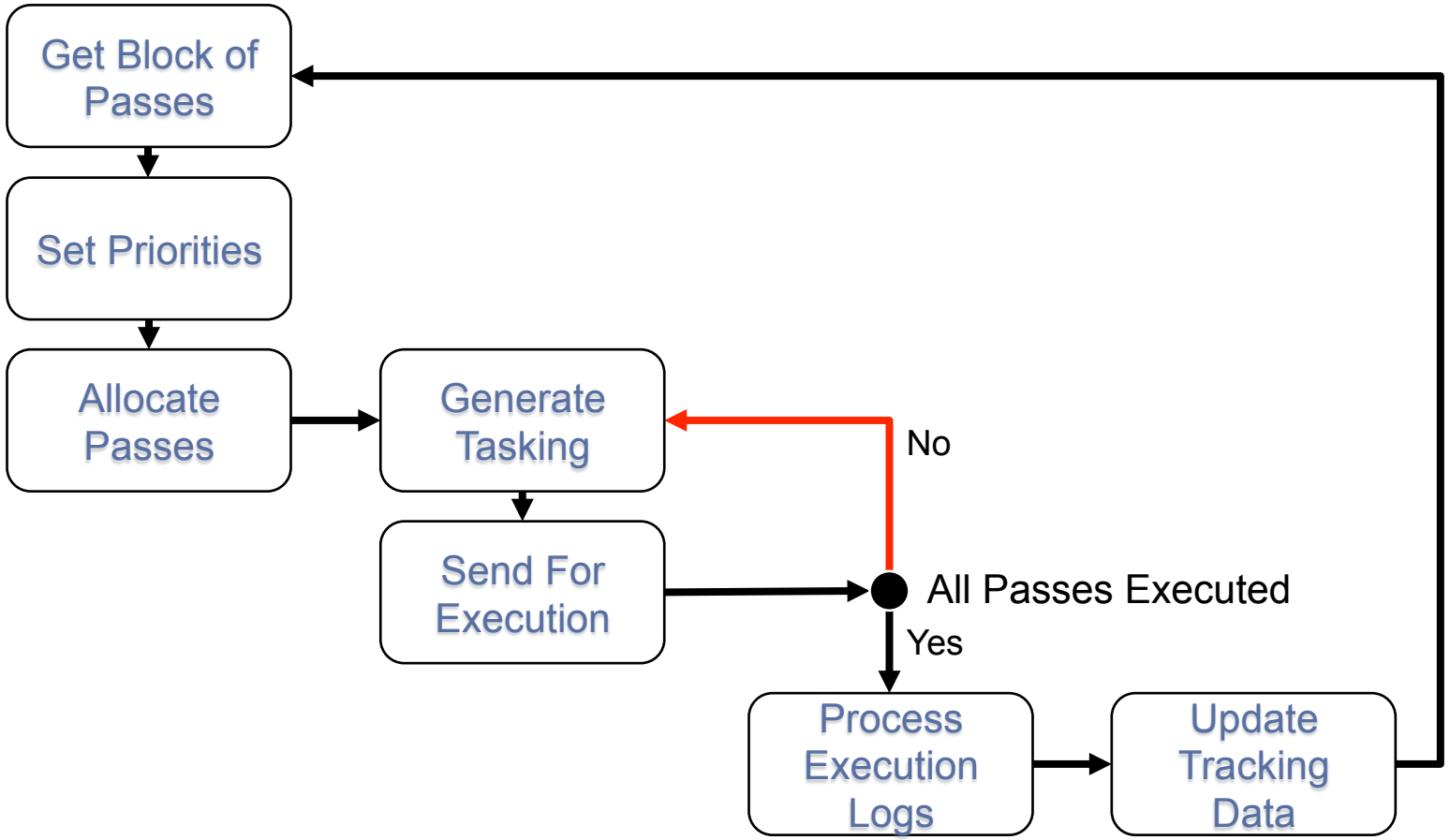
- Considerations
  - User Set Priority
  - Vehicle Restriction
    - Is Telemetry Available
    - When Will Telemetry Buffers Rollover
    - When Does the Current Taking Expire
- Requires a Blending Algorithm!!

- Considerations
  - Conflicts
  - Can Tasks be Combined
  - Timing!!
- Assign Passes based on Priority

- Generate in Execution Order
- Generate Just Prior to Execution
  - Allow End Users First Choice of Passes
- Look for Completed Missions while waiting for next Mission Generation

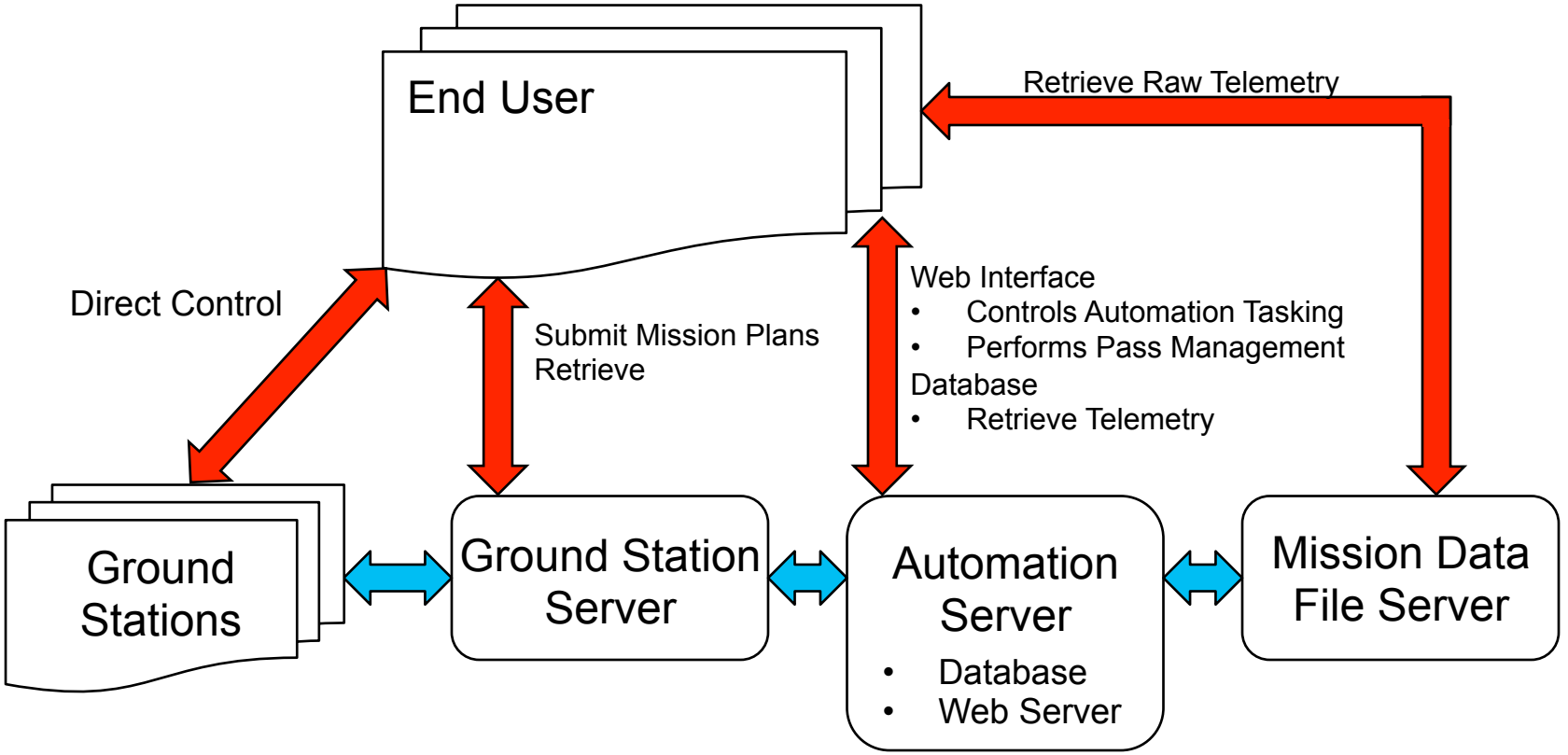
Constellation Automation

Automation Flow



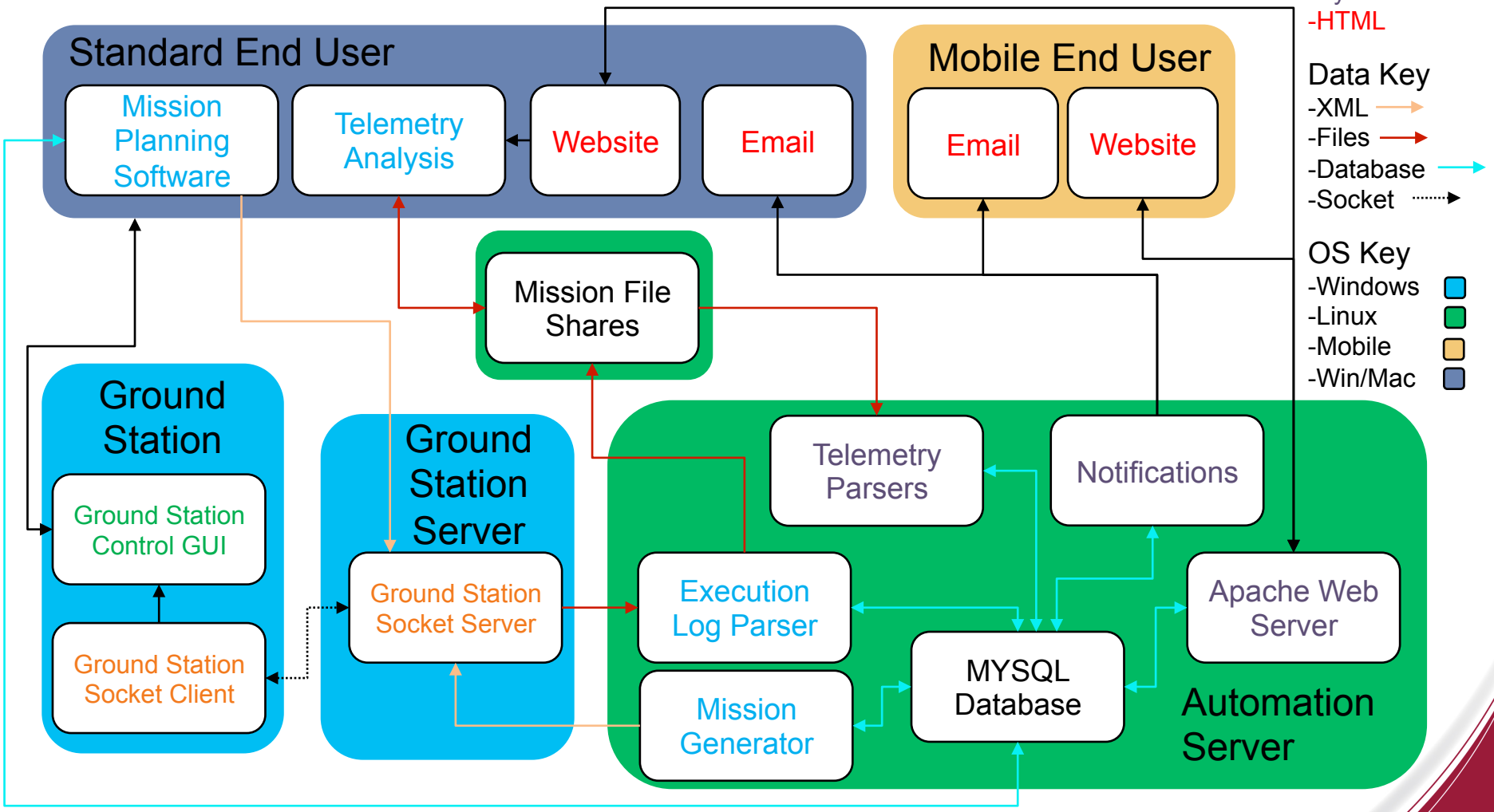
Constellation Automation

Basic Architecture



Constellation Automation

Detailed Architecture



Automation Use Cases

Complex Joint Experiment Planning with CubeSats

- What does it take to execute a complex joint experiment?
  - ***Accurate ephemeris prediction***
  - *Precise attitude control*
  - *Coordinated data exchange between planning groups*
  - *Accurate on-board clock & ability to execute time based stored commands*
- Ephemeris Prediction Options:
  - JSpOC TLEs are released on a (mostly) daily basis
    - Position errors can be too large for some experiments
      - Error is 1–3 km at epoch
      - Error grows at ~10 km per day, making advanced planning difficult
  - GPS Derived Ephemeris
    - Requires on-board GPS receiver
    - Requires potentially large ground network to downlink timely GPS data
    - Requires complex orbit determination software

Automation Use Cases

Complex Joint Experiment Planning with CubeSats

Ephemeris Prediction Position Error as a Function of Planning Lead Time

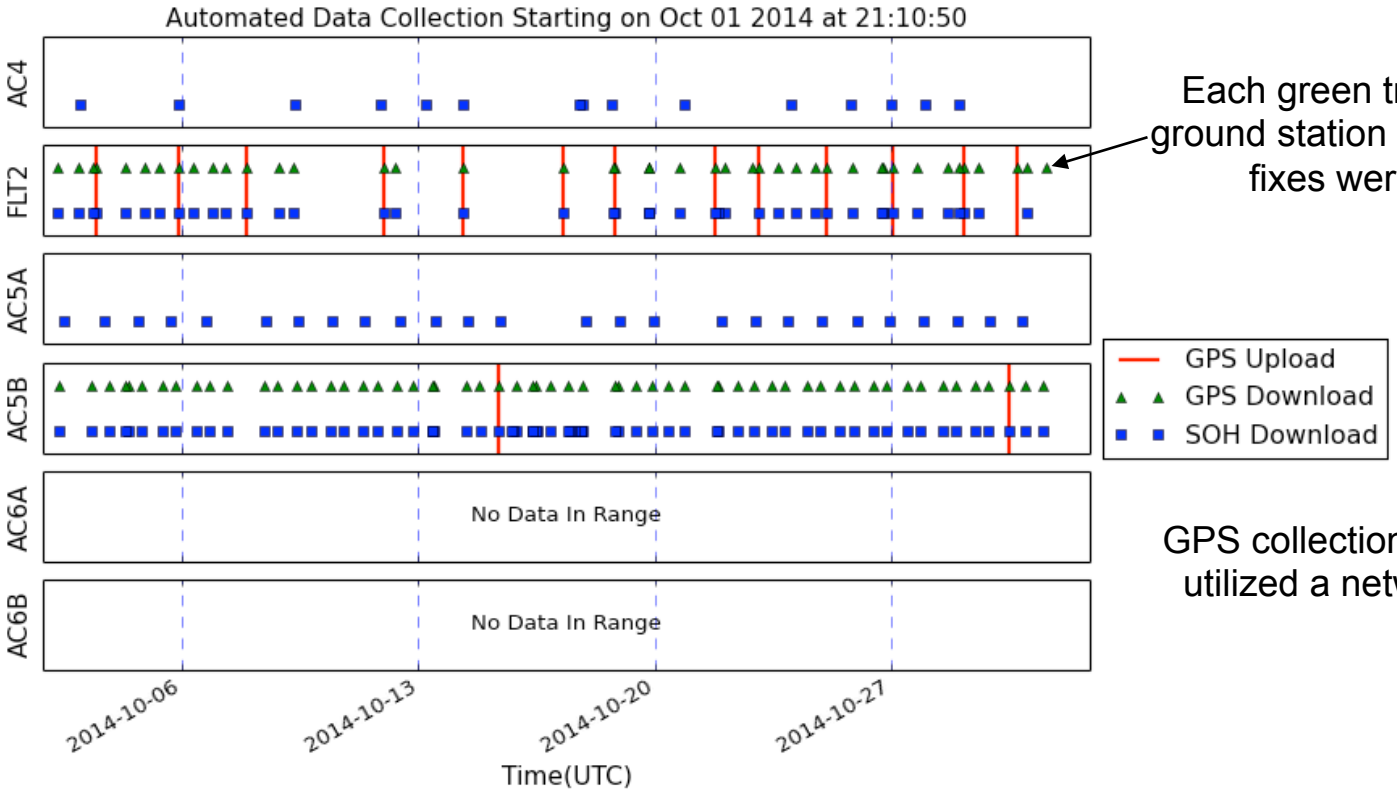
Lead Time	In-Track Position Error	Cross-Track Position Error	Radial Position Error
T-14days	~100 km	~100 m	~2 km
T-3days	2.0 km	12 m	40 m
T-2days	0.4 km	<10 m	<10 m
T-1days	0.3 km	<10 m	<10 m
T-0.5days	0.1 km	<10 m	<10 m

- Note that lead times less than 12 hours could produce better accuracy, however, short lead times are not practical
  - *Limited to ground station access times for GPS data downlink*
  - *Complex experiments require time for planning, data exchange & command uplink*

Automation Use Cases

Complex Joint Experiment Planning with CubeSats

- Regular GPS fix scheduling & downloading is a burden on the ground network
  - *Requires multiple contacts per day to ensure "fresh" GPS data is available for planning*



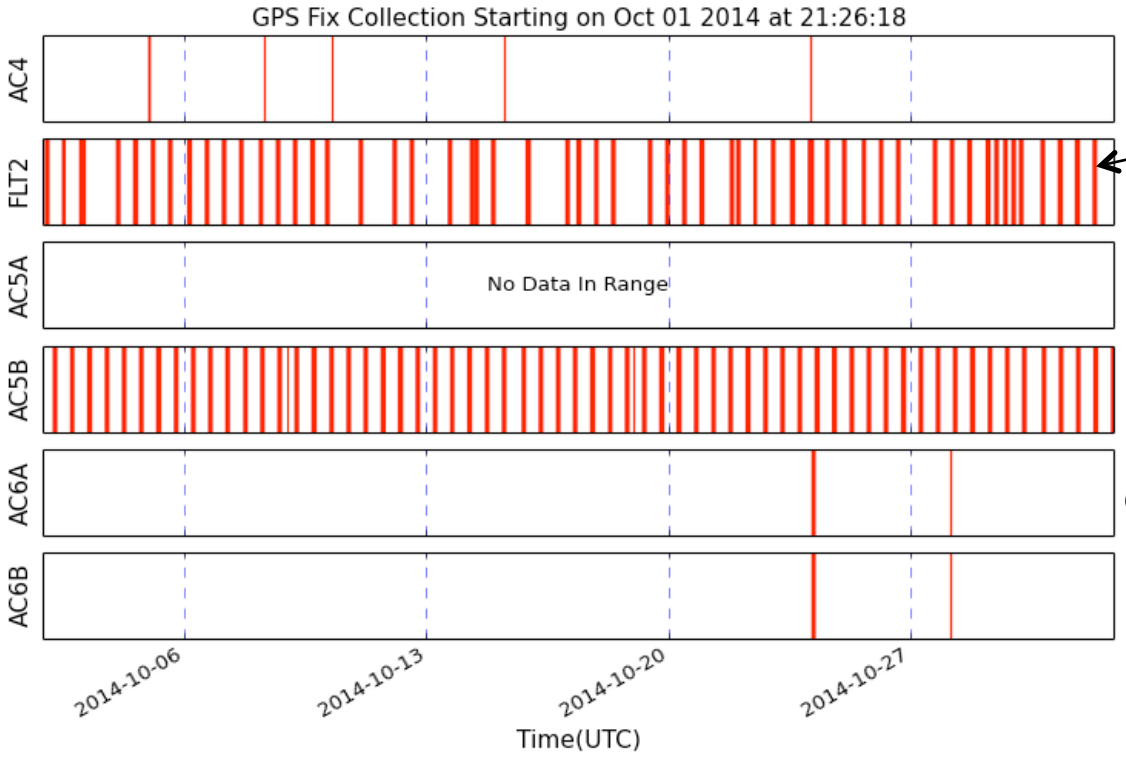
Each green triangle signifies a ground station pass at which GPS fixes were downlinked

GPS collection for 2 vehicles heavily utilized a network of 3 RF stations

Automation Use Cases

Complex Joint Experiment Planning with CubeSats

- GPS data needs to be well distributed throughout the orbit and collected for multiple orbits per day
  - *Each vehicle can collect ~1700 fixes per month to support advanced planning*
  - *Fixes need to be parsed & entered into a database*



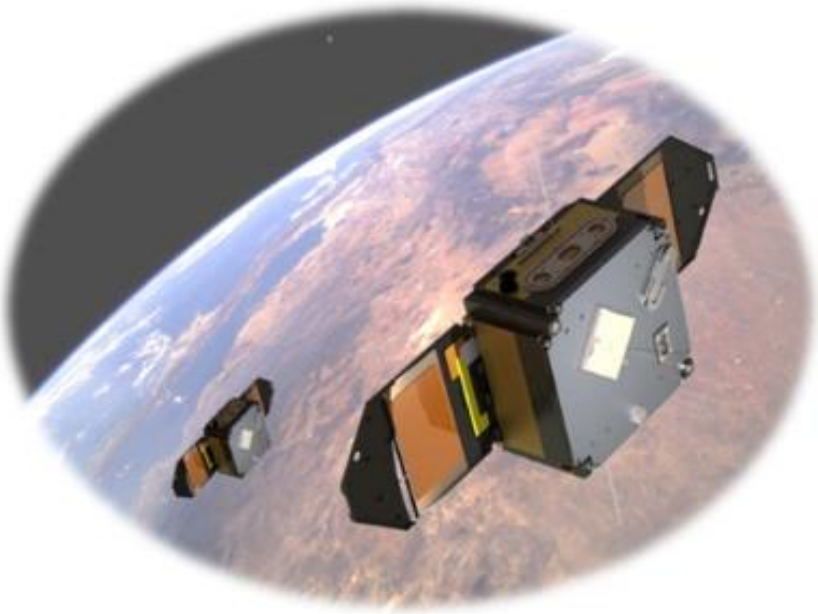
Thicker red bars indicate an entire orbit of GPS data collection, typically 27 individual fixes

Complex Automation Software was developed to manage the GPS collection, downloading and database entry.

Impractical to manage with manual operators

Automation Use Cases
Science Missions

- Two Spin Stabilized 0.5U CubeSats
- Primary Mission
  - *Correlate observations and study small-scale radiation belt structure*



- Unique Requirements
  - *Always on*
    - Gathering dosimeter telemetry
    - Maintaining spin stabilization
  - *Proximity Operations*
    - Need to adjust vehicle separation using differential drag
  - *Regular Contact Intervals*
    - Downlink dosimeter telemetry
    - Downlink pointing telemetry
    - Uplink payload commanding
    - Uplink pointing commands for differential drag

Automation Use Cases

Science Missions

- Regular Cadence to the Mission Operations
  - *Requires 1-1.5 hours to plan daily*

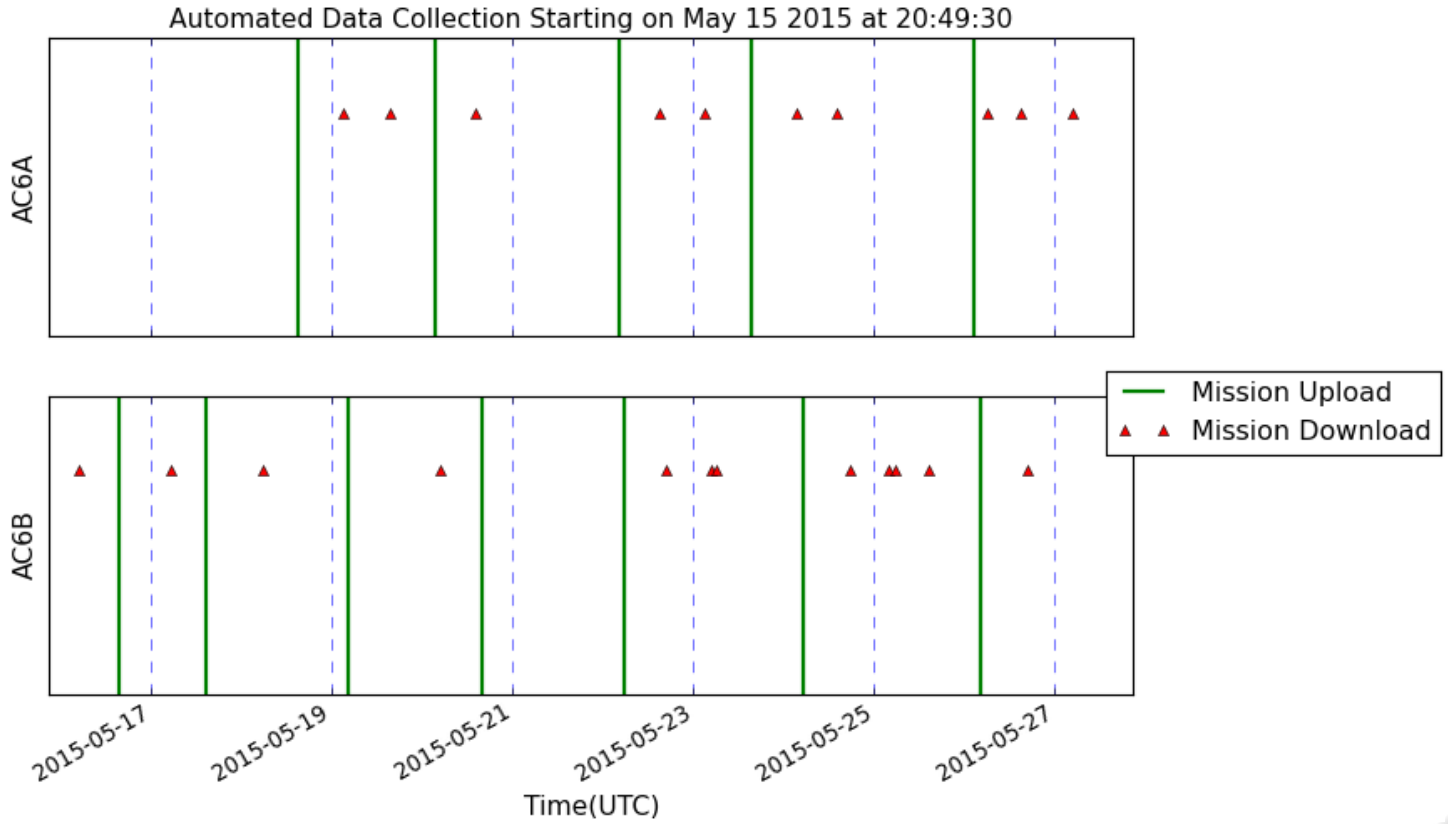
Interval	Task
12 hours	Download Dosimeter Telemetry
1 day	Download Pointing and Health Data
1.5 day	Upload Pointing Commands
2 weeks	Collect GPS Data
2 weeks	Clear Pointing Telemetry Buffer
1 month	Clear Health Telemetry Buffer

Highly Automatable

Automation Use Cases

Science Missions

- Automation Turned on May 15th



Conclusion

- It's possible to blend existing tools into an Automation System
- Taking an incremental approach is a practical method for developing an Automation System
- ***It now take fewer man hours to run 8 satellites than it previously took to run 1***

Questions?