[SSC23-XII-04]

Enabling Mission Success: A Student's Perspective on Developing a Ground Station

Sean Svihla[1], Adrian Bryant[2], Lucia Witikko[3]

Laboratory for Atmospheric and Space Physics, University of Colorado Boulder, 1234 Innovation Drive, Boulder, CO, 80303

Abstract

In recent years, The University of Colorado Boulder's Laboratory for Atmospheric and Space Physics (LASP) has grown its SmallSat Operations (SMOPS) program into a large team of students and professionals managing the operations of multiple SmallSat missions. The team responsible for the on-site ground station (GSOPS) includes a small team of students alongside professionals who help manage the on-site UHF and S-band antennas, making use of open-source, commercial, and in-house software. In this process, the student ground station team has adapted the limited time and resources of a single ground station to accommodate the needs and science objectives of multiple missions. These accommodations include the development of best practices for scheduling around pass overlap, the design of automation capable of handling multiple missions without manual reconfiguration, and the implementation of robust error handling and intuitive paging to reduce the need for human intervention. This paper describes in more detail the students' perspective on the challenges and creative solutions of maintaining and improving the ground station.

[1]Command Controller, LASP. Graduate Student, Applied Mathematics, University of Colorado Boulder
[2]Command Controller, LASP. Undergraduate Student, Aerospace Engineering, University of Colorado Boulder
[3]Command Controller, LASP. Undergraduate Student, Aerospace Engineering, University of Colorado Boulder

Contents

1 Introduction

As Laboratory for Atmospheric and Space Physics's (LASP) SmallSat Operations (SMOPS) program has grown, so have the needs of its ground station, incentivizing the inclusion of students in its development and maintenance. The team responsible for the on-site ground station (GSOPS) now includes a small team of students who help manage the on-site UHF and S-band antennas, making use of open-source, commercial, and in-house software. GSOPS is, among other things, responsible for generating weekly schedules consistent with the science objectives of in-flight missions, assisting with (primarily) end-to-end testing of in-development missions, and responding to anomalies related to the ground station.

The SMOPS team now provides mission operations for three in-flight missions: Colorado Ultraviolet Transit Experiment (CUTE), Compact Total Irradiance Monitor (CTIM), and Colorado Inner Radiation Belt Experiment (CIRBE). Since its creation, the student ground station team has adapted the limited time and resource budgets of a single ground station to accommodate the needs and science objectives of multiple missions. The goal of this paper is to share our, the student GSOPS team's, perspective on this development and on the challenges of operating a ground station under the constraints of multiple missions.

In the following sections, we outline our goals for addressing the challenge of accommodating multiple missions with limited hardware, the constraints we operated under, our solutions, and our lessons learned from the process.

2 Description of Hardware and Software

Fig.1 displays the command and control interfaces of LASP's in-flight missions. The ops machine exchanges commands and telemetry with the Ultra High Frequency (UHF) band ground station machine during real-time commanding, and the S-Band infrastructure acts as a channel for the high-rate downlink of science data.

On both the UHF and S-Band Software Defined Radio (SDR) machines, software interprets radio signals to and from the spacecraft. Our ground station uses GNURadio, a free and open source SDR. Due to different needs, each spacecraft has a slightly different radio configuration.

In order to account for the Doppler Effect on the frequency from a satellite moving nearly 7.5 km/s, we apply a Doppler Correction on the ground. This functionality is facilitated by a free and open source software called Gpredict. Gpredict ingests the latest two-line element (TLE) measurements from NORAD and uses them to track a CubeSat's orbit. This functionality also allows Gpredict to act as our UHF rotator control, commanding the physical antenna to track the satellite across the sky. As the UHF SDR interprets telemetry information, it forwards it to our command and command and control software for an operator to interact with. The S-Band GS system, conversely, is self contained, operating independently. It uses an SDR configured for a much higher S-Band frequency. The data it detects and decodes is distributed to the CubeSat science teams. The S-Band dish is a more complex system than the UHF rotator and requires a different software, TrackGUI, to supply pointing commands. Since the tracking and radio is split across two computers, the S-Band SDR also requires Gpredict to apply a Doppler Correction.

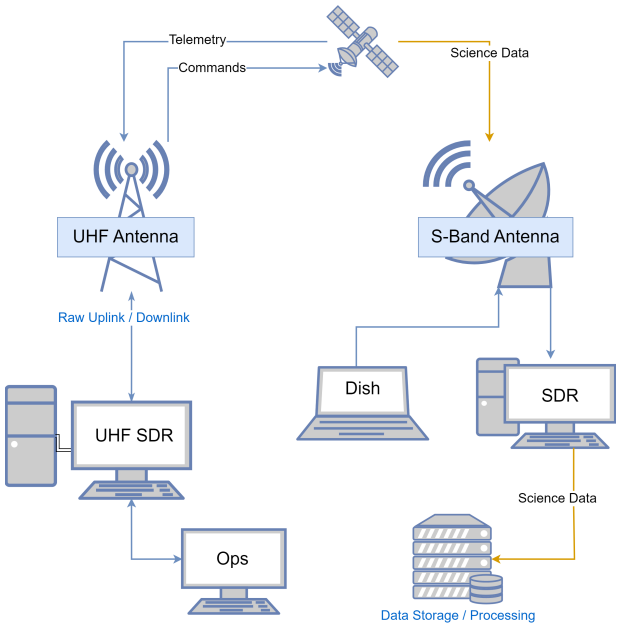


Figure 1: GSOPS Data Concept

3 Objectives and Constraints

Because the ground station consists of only a single UHF and single S-band antenna, it provides an inherent scheduling challenge, and the unique constraints of each mission demand careful coordination and consideration of individual missions' needs and science

objectives. As overlap is unavoidable and has the potential to interrupt the delivery of data to the science teams, it is the primary constraint around which we have tried to organize operation of the ground station.

Our ultimate goal is to manage the ground station so as to provide timely delivery of data to the science teams; however, this task requires more than just time for downlinks. Throughout each planning cycle, the planning and science teams coordinate with one another on when science data will be collected, when science data will be downlinked, and, most importantly, when the spacecraft will be commanded to do any of this. The latter two tasks, as well, require the coordination of the ground station.

A ground station which serves a single missions can be run with little automation: An instance of Gpredict may be started as needed or left configured for the mission indefinitely to allow for automated commanding or scheduled overnight downlinks. A second mission requires the operations team to either invest a great deal of time in reconfiguring the ground station between passes so that both missions can be serviced or sacrifice otherwise good passes during periods when the ground station is configured for the wrong mission. Both options prevent the ground station from participating in an efficient cycle of planning and downlinking. Add a third mission, and this task becomes almost impossible without around-the-clock hands-on operating. The "off-console" work required to coordinate the process is just as time consuming.

Our desire for the ground station is that it can (1) coordinate the schedules of multiple missions, (2) operate with as much autonomy as is reasonable (including reconfiguring itself between passes), and (3) be adapted to new missions with novel scheduling needs in the future. In the following sections, we describe the more concrete objectives we set in order to meet this desire as well as the factors we identified as limiting our ability to do so.

3.1 Objectives

The final implementation of the ground station should be **fault tolerant**. A highly automated ground station is only useful inasmuch as it can respond to faults. One that requires hours of attention responding to faults is not much better than one that requires hours of attention to operate. A *usefully* autonomous ground station is able to diagnose and resolve its own faults and elevate the issue to a human operator only when absolutely necessary.

Of course, the ideal system does not have faults at all; however, acknowledging that faults in general are unavoidable, our goal is to create a system which is robust to faults. Necessary for being robust to faults is identification of common and easy-to-resolve faults as well as a helpful paging system—that is, one that does not "over-page" and thus get ignored. Wherever reasonable, we would like the ground station to resolve faults quietly and return quickly to normal operation.

Some degree of human input cannot be avoided (and might even be desired), but where this is necessary the ground station should be **easy to use**. Even a fault-tolerant system might be cumbersome if anomaly resolution and routine human interactions are unintuitive and needlessly complicated. Again, a *usefully* autonomous ground station is able to *guide* the user through its routine uses and meaningfully point to unidentified or unresolved faults. This reduces the intensity with which future team member need to be trained as well as makes the ground station more accessible to non-members of the team.

The ground station is unlikely to ever be "finished": There will continue to be new bugs patched and new behaviors implemented; therefore, the system should be **easy to change**. The software should be modular to facilitate future work. Because we want to minimize necessary downtime, anticipated changes such as the addition of a new mission should be possible through a user-interface.

3.2 Constraints

Development of a new system is constrained in that we **should not expand existing hardware**. The inclusion of other ground stations in the LASP network might be a sensible solution; however, relying on expansion of hardware alone comes at a steep cost considering that a single set of hardware has ample time to accommodate multiple missions, if only it can be managed properly. Our improvements should focus on development of software that optimizes the use of existing hardware rather than expansion of hardware. If the system makes efficient use of limited software first, then whenever hardware is expanded, we can be more confident in the necessity of the expense and effort.

A further constraint of this development is that **current operations cannot be halted** even as the need for improvement grows. We can neither halt operations entirely during development nor leave the old system in place until it is complete. This constrains both the manner and rhythm of development as we are required to release features of the final design periodically as standalone updates. The result

is that design decisions might be made prematurely or under more constraints than might otherwise be present.

The operations team is responsible for more tasks than planning and scheduling the ground station—telemetry monitoring and delivery of science data, to name a few–and many of these tasks require interfacing with the spacecrafts' schedule. As such, the new system must **operate within existing workflows**. These points of interface are the most immediate constraints on development, as the new system should be outwardly identical to the old in order to not require rewriting of an even larger set of software.

4 Approach and Perspective

4.1 Pass Automation

In reference to the data concept illustrated in Fig.1, instances of ground station automation are hosted on the SDR machines, which are the interface between the physical antennas and the rest of the operations system. We identified the need for reconfiguration between passes as the primary source of unnecessary human effort (and the primary challenge for automation to address).

Although every mission shares an interface to the antennas, each requires a unique software-defined radio (SDR), tracking configuration, and radio configuration, which contain such information as location, uplink/downlink frequencies, modulation, and encoding—without any of which even routine operations become infeasible. GNURadio makes it simple to setup and execute mission-specific SDRs from the command-line; Gpredict, however, makes tracking and radio configuration inaccessible from the command-line by default. We identified two behaviors we wished to be free of human interaction: the abilities to (1) track and engage radio and rotator controllers and (2) configure for a given satellite and transponder.

With some effort, we implemented in our fork[4] of Gpredict the option to track and engage the first available radio and rotator on startup as well as an expansion of the available configuration options. These changes allow automation to swap predefined mission-specific configuration files in and out of Gpredict's configuration directory and enable it to begin tracking a satellite on startup automatically. Our automation software now needs only to ingest the ground station's schedule and configure and execute these programs as passes approach.

This newfound autonomy satisfies our desire for the new system to be easy to use. Whereas the previous paradigm forced us to decide between wasted effort or wasted passes, the new system can handle complex schedules without human input—in fact, in some cases, it can tear down and set up between passes faster than a human operator, increasing the total number of passes we can run in a day. Moreover, because our design philosophy emphasized configuration files, the system is easy to change to accommodate new missions.

4.2 Paging and Fault Tolerance

Fault tolerance consists of two halves: fault detection and fault response. The ground station should be able to detect its own errors either resolve them or escalate the issue to a human operator. To do this, we needed to identify the most common detectable errors, their urgency, and whether or not they can be resolved procedurally.

Our paging design differentiates between "internal" and "external" fault detection. Pass automation can detect three internal faults: a failure of its own sub-processes, the absence has a schedule file, and invalid configuration files. If a fault interrupts the automation process, however, we lose all internal fault detection. As automation might run indefinitely without human input, such catastrophic failures might go unnoticed without a secondary paging system.

This possibility necessitates an external watchdog process. The watchdog references the pass schedule to compare the running process with what it expects at certain points in a pass. While in a pass, it expects the SDR, tracking software (Gpredict), and pass automation to be running, and outside a pass it expects only pass automation to be running. At regular intervals throughout the day, the cron daemon executes a program to assess and report on the status of the entire ground station. This approach also allowed the watchdog program to run to completion every time, minimizing the inherent risk found in internal fault detection.

With reliable fault detection in place, the system becomes capable of several fault *responses*—for example, the SDR might fail on startup, but internal fault detection can detect this failure and restart the program. The issue is escalated to a human operator only when necessary, a practice which helps to prevent "page blindness". A overactive notification system with unimportant warnings can cause impor-

[4] A "fork" is a term used to describe a copy made of an open-source piece of code. Changes made to this copy affect the internal implementation of the program without affecting the publicly available code.

tant warnings to be ignored.

If fault tolerance determines an issue is important enough for intervention, then the tools required to fix a problem should be built into the system. Pages include the detected error(s), the common response to each error, and a diagnostic screenshot of the alerting computer. The person in the loop can identify if it is a common error and then follow the Standard Operating Procedure (SOP) for that error.

4.3 Scheduling

As much effort is required to operate the ground station according to a complex multi-mission schedule, at least as much is required to design such a schedule. The scheduling process must balance the objectives and constraints of several missions, and doing so requires the synthesis of information from a wide range of tools and software, with more interfacing than any other part of operations.

The primary factor complicating the scheduling process is pass overlap, an increasingly present challenge with the addition of more missions. A human scheduler must keep and delete weeks-worth of passes according to mission-specific constraints on minimum number of downlinks, maximum time between downlink, and maximum elevation. This process occurs in collaboration with science planning teams, and when the science planning and ground station schedules disagree, the scheduler is responsible for *re*scheduling. As the number of missions increased, this juggling act quickly became unmanageable and error-prone for a single person, even with the minimal automation which existed.

Despite its complexity, large parts of the scheduling process are formulaic and lend themselves to automation. We compiled the many tools used by the scheduler into a single piece of software able to generate a tenable schedule and, more importantly, identify where it fails to meet known constraints. The task of the scheduler, then, becomes to review the schedule and decide if and how failures to meet constraints can be mitigated.

This more autonomous process satisfies our desire for the new system to be easy to use by removing human input except where it is desired—that is, in reviewing the final schedule and making arbitrary conflict resolutions. Moreover, this ease of use makes the system easy to change as well, as a modified schedule can be generated quickly in response to immediate operational needs.[5]

5 Results

5.1 Lessons Learned

The task of improving a functioning ground station upon which operations of missions depends forced us to compromise on the scope and speed of our development. Working on such a dynamic system without the option of downtime forced us to greatly compartmentalize our changes and design them in such a way that they can be easily reverted from without disrupting the normal flow of operations. This fact further constrained us to making improvements which fit inside of the current framework. The current state of the ground station has room for improvement, but we have placed it in a state that allows us to manage the complex coordination required to operate it—and the bandwidth the further improve upon it.

Any system, no matter how well-designed, will encounter failures. Recognizing this, we designed a system that is focused on robustness to faults rather than removal of faults. Of course, when they are identified, bugs are pursued and solutions are attempted; however, it is often the case that bugs are found in operations rather than testing. In such cases, the ground station should be able to operate until a patch can be made. This ability is all the more important in the development environment described, in which more immediate tasks need to be dealt with in the interest of continuous operations.

The improvements made to pass scheduling taught us to evaluate the degree of manual control necessary for an effective tool. Early on, we imagined a completely autonomous scheduling process. Throughout development, however, we found that individual mission needs could change on a day-by-day basis and impacted one another during periods of pass overlap. We changed our philosophy from creating a tool to automate the entire process to a tool that automated the most difficult steps. A computer can better cross-check ground pass times against spacecraft downlinks, while a person can decide to prioritize a mission so its operations team can resolve an anomaly. This lesson emphasized the importance of good user interface design and accessibility of tools.

5.2 Accomplishments

These improvements culminated in a demonstrably improved Ground System. The focus on ease-of-use and adaptability meant that the addition of a third

[5]For a more detailed description of the changes between the previous and current versions of the scheduling process see Appendix A: Scheduling Processes. For a visual representation of the current scheduling tool in action, see Appendix B: Terminal Scheduling Tool.

mission, CIRBE, was seamless. CIRBE's rapid integration into the UHF and S-Band systems meant it was fully commissioned in five days, a milestone projected by conservative estimates to be measured in weeks.

Pass automation's ability to recover from common issues prevented an occasional SDR bug from causing missed passes. Over the most active 12 day period, the LASP Ground Station was able to successfully complete 12% more UHF passes and 9% more S-Band passes than it could have otherwise. These figures do not include the increased pass time due to automatic configuration between passes.

The most drastic upgrades to the Ground System were within pass scheduling. New scheduling methods included the use of a more accurate pass propagation software. Since pass schedules are created at a weekly cadence, pass times should be accurate to at least 6 days after generation. 6 days after creation, the new pass propagation software was 16x as accurate. Additionally, the previous method of weekly pass scheduling was time and labor intensive. The new method of pass scheduling reduced a 2 hour process to a 15 minute process, a speedup of 8x.

6 Final Remarks

Our development of the ground station has been driven by the need to more efficiently make use of limited time and human resources to accommodate multiple CubeSat missions. We identified scheduling and pass configuration as the primary contributors to wasted effort and created procedures to carry out routine tasks within a fault-tolerant framework that usefully guides human operators through tasks where their input is needed. Critical to the ground station's success is its ability to adapt to new missions with unique scheduling and configurational needs in the future. Under this framework, operators are able to focus their effort where it is most useful and depend on a reliable system of automation. The current state of the ground station is an efficient and easily maintainable system of automation capable of adapting to unique needs in the future.

7 Acknowledgements

The LASP student ground station team would like to thank the LASP Mission Operations team and the LASP Smallsat Operations team for their support.

Special thanks to the following people:

Sierra Flynn -
LASP Mission Operations Manager for SmallSats, LASP Flight Director
Gabe Bershenyi -
LASP Flight Controller
Dr. Adalyn Fyrhie -
LASP Systems Engineer
Scott Genari -
LASP Ground Station Enginner
Nicholas DeCicco -
LASP Professional Research Assistant

Appendix A: Scheduling Processes

Previous Scheduling Process

1. On each mission's development computer, find all scheduled ATS downlinks for that mission
2. On the UHF SDR computer, generate the pass schedule for the next two weeks using an IDL script.
3. Manually edit pass schedule file:
   (a) Open the schedule file in a csv editor.
   (b) Manually verify that all overlaps were mitigated correctly.
   (c) Manually verify that all passes with ATS downlinks are kept.
4. Convert the schedule file into .sav file using an IDL script.
5. On both the UHF SDR and S-band SDR computers, manually verify that automation is looking for correct schedule and restart automation.
6. On the S-Band Dish computer, manually add every kept pass into TrackGUI.
7. On the UHF SDR computer, manually transfer all files to correct destinations.
8. On each mission's development computer, generate and send the dayproc[6] for that mission.
9. On each mission's operations computer, restart OASIS-CC with the new dayproc for the mission.
10. Notify the ground station and smallsat mission operations teams that the scheduling process has been completed.

Current Scheduling Process

1. On the UHF SDR computer, call the schedule generation python script for the next two weeks.
2. Use the manual editing functionality in this script to easily change the schedule (no manual checks required, all are done automatically).
3. On both the UHF SDR and S-band SDR computers, manually verify that automation is looking for correct schedule and restart automation.
4. On the S-Band Dish computer, mark all kept passes as track in TrackGUI (they will have been added automatically).
5. On each mission's operations computer, restart OASIS-CC[7] with the new dayproc for the mission (they will have been automatically generated and sent from the development computers).

Notable Changes

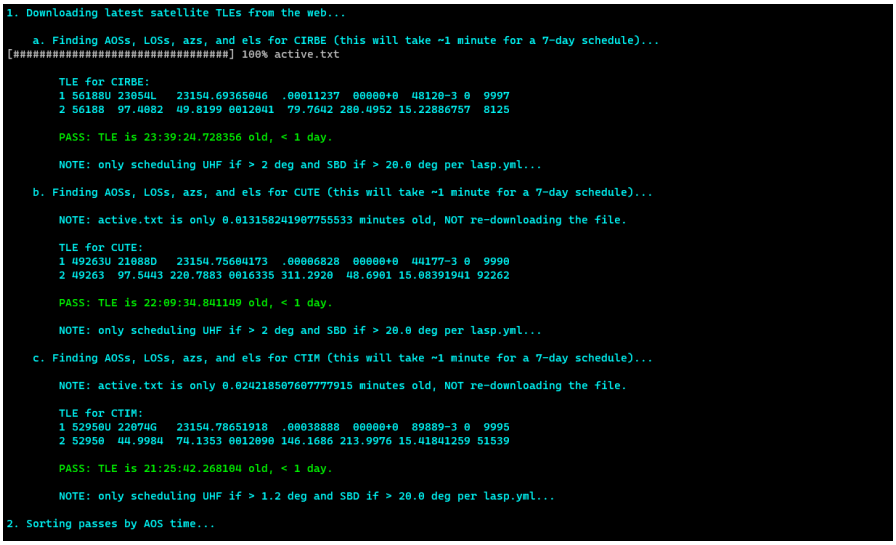
Many steps in the previous version of the scheduling process were automated when moving to the current version. These step include: determining the scheduled ATS[8] downlink times for each mission, checking that all pass overlaps were correctly mitigated, checking that all passes with ATS downlinks are kept, converting the schedule into desired file types, adding all kept passes to TrackGUI, transferring all files to correct destinations, generating and sending the dayproc for each mission, and notifying all correct channels of the completion of the scheduling process. With the automation of all these steps and the simplification of the manual editing process, the total amount of time required to complete the weekly scheduling process dropped from roughly two hours to under fifteen minutes.

[6] A "dayproc" is automation that allows the command and control computer to automatically connect to the ground station at the correct pass time.

[7] OASIS-CC is LASP's internal Command and Control software used to view spacecraft telemetry and send commands.

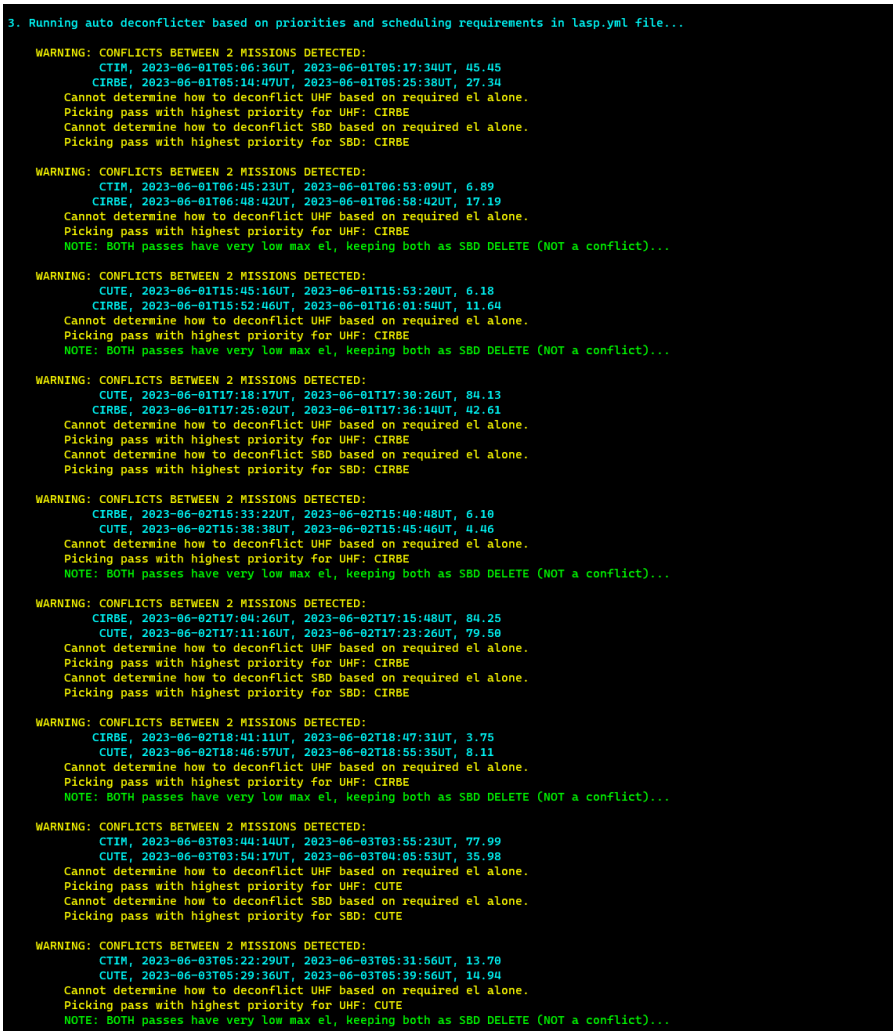
[8] "An "ATS" (Absolute Time Sequence) refers to commands stored onboard a spacecraft that execute at predetermined times. The spacecraft is loaded with future pass times at which it is instructed to begin S-Band downlinks.

Appendix B: Terminal Scheduling Tool



Step 1:
The TLEs for each mission are pulled from the web, and are verified to be up to date.

Step 2:
Using these TLEs, information from the yml file, and the python package SkyField, the passes for all missions are sorted by AOS time.



Step 3:
Conflicts between missions are automatically mitigated, and passes with elevations under the minimum requirement for the UHF and S-Band antennas are deleted. These decisions are displayed in the terminal for reference.

```
    WARNING: CONFLICTS BETWEEN 2 MISSIONS DETECTED:
             CUTE, 2023-06-05T05:15:03UT, 2023-06-05T05:26:08UT, 21.69
            CIRBE, 2023-06-05T05:26:17UT, 2023-06-05T05:37:27UT, 38.65
        Cannot determine how to deconflict UHF based on required el alone.
        Picking pass with highest priority for UHF: CIRBE
        Cannot determine how to deconflict SBD based on required el alone.
        Picking pass with highest priority for SBD: CIRBE

    WARNING: CONFLICTS BETWEEN 2 MISSIONS DETECTED:
            CIRBE, 2023-06-05T05:26:17UT, 2023-06-05T05:37:27UT, 38.65
             CTIM, 2023-06-05T05:40:07UT, 2023-06-05T05:43:19UT, 0.87
        Keeping pass with reasonable el for UHF (NOT a conflict): CIRBE
        Keeping pass with reasonable el for SBD (NOT a conflict): CIRBE

    WARNING: CONFLICTS BETWEEN 2 MISSIONS DETECTED:
             CTIM, 2023-06-06T04:57:32UT, 2023-06-06T05:03:24UT, 3.29
            CIRBE, 2023-06-06T05:05:56UT, 2023-06-06T05:16:29UT, 21.38
        Cannot determine how to deconflict UHF based on required el alone.
        Picking pass with highest priority for UHF: CIRBE
        Keeping pass with reasonable el for SBD (NOT a conflict): CIRBE

    WARNING: CONFLICTS BETWEEN 2 MISSIONS DETECTED:
            CIRBE, 2023-06-06T05:05:56UT, 2023-06-06T05:16:29UT, 21.38
             CUTE, 2023-06-06T05:07:48UT, 2023-06-06T05:19:10UT, 26.12
        Cannot determine how to deconflict UHF based on required el alone.
        Picking pass with highest priority for UHF: CIRBE
        Cannot determine how to deconflict SBD based on required el alone.
        Picking pass with highest priority for SBD: CIRBE
PASS: ALL conflicts within timerange automatically mitigated.
```

```
4. Finding ATS downlink pass times within timerange...

    a. CIRBE ATS downlink pass times within timerange...

        Grabbing all CIRBE ATSs from CIRBE Ops Machine



        Grabbing all CIRBE ATS downlink pass times for each ATS from CIRBE ATS Directory

        Finding all CIRBE ATS downlink pass times within timerange...


            CIRBE ATS cirbe_ats_23_146.prc:
                CIRBE Downlink pass at: 2023/152-05:14:30
                CIRBE Downlink pass at: 2023/152-17:24:43
            CIRBE ATS cirbe_ats_23_153.prc:
                CIRBE Downlink pass at: 2023/153-06:27:58
                CIRBE Downlink pass at: 2023/153-17:04:27
                CIRBE Downlink pass at: 2023/154-06:07:20
                CIRBE Downlink pass at: 2023/154-16:44:04
                CIRBE Downlink pass at: 2023/155-05:46:47
                CIRBE Downlink pass at: 2023/155-16:23:54
                CIRBE Downlink pass at: 2023/156-05:26:20
                CIRBE Downlink pass at: 2023/156-17:36:51
                CIRBE Downlink pass at: 2023/157-06:39:40
                CIRBE Downlink pass at: 2023/157-17:16:02

    b. CUTE ATS downlink pass times within timerange...

        Grabbing all CUTE ATSs from CUTE Ops Machine



        Grabbing all CUTE ATS downlink pass times for each ATS from CUTE ATS Directory

        Finding all CUTE ATS downlink pass times within timerange...


            CUTE ATS cute_ats_23_153.prc:
                CUTE Downlink pass at: 2023/153-04:01:13
                CUTE Downlink pass at: 2023/154-03:54:18
                CUTE Downlink pass at: 2023/154-17:04:17
                CUTE Downlink pass at: 2023/155-03:47:25
                CUTE Downlink pass at: 2023/155-16:57:17
                CUTE Downlink pass at: 2023/156-03:40:32
                CUTE Downlink pass at: 2023/156-16:50:17

    c. CTIM ATS downlink pass times within timerange...

        Grabbing all CTIM ATSs from CTIM Ops Machine



        Grabbing all CTIM ATS downlink pass times for each ATS from CTIM ATS Directory

        Finding all CTIM ATS downlink pass times within timerange...


            CTIM ATS ctim_ats_23_152g_t.prc:
                CTIM Downlink pass at: 2023/154-02:06:20
            CTIM ATS ctim_ats_23_152i_t.prc:
                CTIM Downlink pass at: 2023/154-22:08:50
                CTIM Downlink pass at: 2023/154-23:46:42
                CTIM Downlink pass at: 2023/155-01:25:02
                CTIM Downlink pass at: 2023/155-03:03:01
            CTIM ATS ctim_ats_23_152j_t.prc:
                CTIM Downlink pass at: 2023/155-21:27:40
                CTIM Downlink pass at: 2023/155-23:05:19
                CTIM Downlink pass at: 2023/156-00:43:39
                CTIM Downlink pass at: 2023/156-02:21:41
            CTIM ATS ctim_ats_23_152k_t.prc:
                CTIM Downlink pass at: 2023/156-20:46:28
                CTIM Downlink pass at: 2023/156-22:23:52
            CTIM ATS ctim_ats_23_152l_t.prc:
                CTIM Downlink pass at: 2023/157-00:02:11
                CTIM Downlink pass at: 2023/157-01:40:17
            CTIM ATS ctim_ats_23_152a_t.prc:
                CTIM Downlink pass at: 2023/152-01:50:29
                CTIM Downlink pass at: 2023/152-03:28:41
            CTIM ATS ctim_ats_23_152m_t.prc:
                CTIM Downlink pass at: 2023/157-21:42:22
                CTIM Downlink pass at: 2023/157-23:20:37
            CTIM ATS ctim_ats_23_152n_t.prc:
                CTIM Downlink pass at: 2023/158-00:58:47
                CTIM Downlink pass at: 2023/158-02:36:41
```

Step 4:
ATS downlink times occurring within the schedules time window are determined and printed to the terminal for reference.

```
        CTIM ATS ctim_ats_23_152c_t.prc:
            CTIM Downlink pass at: 2023/152-23:31:07
            CTIM Downlink pass at: 2023/153-01:09:18
        CTIM ATS ctim_ats_23_152d_t.prc:
            CTIM Downlink pass at: 2023/153-02:47:33
            CTIM Downlink pass at: 2023/153-04:25:28
        CTIM ATS ctim_ats_23_152f_t.prc:
            CTIM Downlink pass at: 2023/153-22:50:00
            CTIM Downlink pass at: 2023/154-00:28:02
```

```
5. Checking initial schedule against scheduling requirements...

    a. Checking schedule for overlap...

        WARNING: CONFLICTS BETWEEN 2 MISSIONS DETECTED:
                CIRBE, 2023-06-06T05:05:53UT, 2023-06-06T05:16:26UT, 21.35
                 CUTE, 2023-06-06T05:07:46UT, 2023-06-06T05:19:08UT, 26.13
            Suggest picking pass with highest priority for UHF: CIRBE
            Suggest picking pass with highest priority for SBD: CIRBE
    FAIL: 1 conflicts detected within timerange.

    b. Checking schedule against mission requirements...
        i.   Checking downlinks per day requirements for each mission...
            WARNING: CUTE does NOT meet the minimum number of downlinks per day (2/day) on doy 152!
            WARNING: CUTE does NOT meet the minimum number of downlinks per day (2/day) on doy 153!
            WARNING: CIRBE does NOT meet the minimum number of downlinks per day (2/day) on doy 158!
            WARNING: CUTE does NOT meet the minimum number of downlinks per day (2/day) on doy 158!
            WARNING: CTIM does NOT meet the minimum number of downlinks per day (3/day) on doy 158!
        FAIL: Pass schedule does not meet downlinks per day requirements for each mission.

        ii.  Checking time between downlinks requirements for each mission...
            WARNING: CUTE does NOT meet the minimum number of hours between downlinks (14 hours) around 2023-06-01 04:08:07.946468!
            WARNING: CUTE does NOT meet the minimum number of hours between downlinks (14 hours) around 2023-06-02 04:01:12.312492!
        FAIL: Pass schedule does not meet time between downlinks requirements for each mission.

        iii. Checking pass length requirements for each mission...
        PASS: Pass schedule meets pass length requirements for each mission.

        iv.  Checking uhf elevation requirements for each mission...
        PASS: Pass schedule meets uhf elevation requirements for each mission.

        v.  Checking sbd elevation requirements for each mission...
        PASS: Pass schedule meets sbd elevation requirements for each mission.

    FAIL: Pass schedule does not meet all mission requirements.

    c. Checking schedule against ats downlinks...
    PASS: All passes with ATS downlinks have been marked SBD keep/keep_conflict.

FAIL: Pass schedule does not meet all scheduling requirements, suggest manually deconflicting to mitigate errors.
```

Step 5:
The initial schedule is checked for overlaps, mission requirements, and ATS downlink times. Errors found by these checks are printed to the terminal for reference.

```
INPUTTED (V 0.0) Satellite Passes at BOULDER for 2023-06-01_2023-06-07
Index, Satellite,          Start Time,            End Time, Peak Elevation, UHF Priority, S-Band Priority, Manually Edited
  0,        CTIM, 2023-06-01T01:50:28UT, 2023-06-01T02:01:25UT,         35.01,          Keep,             Keep, False
  1,        CUTE, 2023-06-01T02:37:28UT, 2023-06-01T02:43:24UT,          3.15,          Keep,           Delete, False
  2,        CTIM, 2023-06-01T03:28:41UT, 2023-06-01T03:39:44UT,         45.74,          Keep,             Keep, False
  3,        CUTE, 2023-06-01T04:08:07UT, 2023-06-01T04:19:59UT,         53.87,          Keep,             Keep, False
  4,        CTIM, 2023-06-01T05:06:35UT, 2023-06-01T05:17:33UT,         45.44, Delete_Conflict, Delete_Conflict, False
  5,       CIRBE, 2023-06-01T05:14:47UT, 2023-06-01T05:25:38UT,         27.35, Keep_Conflict,   Keep_Conflict, False
  6,        CUTE, 2023-06-01T05:44:16UT, 2023-06-01T05:53:35UT,          9.93,          Keep,           Delete, False
  7,        CTIM, 2023-06-01T06:45:22UT, 2023-06-01T06:53:09UT,          6.89, Delete_Conflict,          Delete, False
  8,       CIRBE, 2023-06-01T06:48:42UT, 2023-06-01T06:58:42UT,         17.20, Keep_Conflict,           Delete, False
  9,        CUTE, 2023-06-01T15:45:16UT, 2023-06-01T15:53:20UT,          6.18, Delete_Conflict,          Delete, False
 10,       CIRBE, 2023-06-01T15:52:46UT, 2023-06-01T16:01:54UT,         11.63, Keep_Conflict,           Delete, False
 11,        CUTE, 2023-06-01T17:18:17UT, 2023-06-01T17:30:26UT,         84.14, Delete_Conflict, Delete_Conflict, False
 12,       CIRBE, 2023-06-01T17:25:02UT, 2023-06-01T17:36:14UT,         42.60, Keep_Conflict,   Keep_Conflict, False
 13,        CUTE, 2023-06-01T18:54:12UT, 2023-06-01T19:02:10UT,          6.45,          Keep,           Delete, False
 14,        CTIM, 2023-06-01T21:54:26UT, 2023-06-01T22:04:08UT,         14.87,          Keep,           Delete, False
 15,        CTIM, 2023-06-01T23:31:07UT, 2023-06-01T23:42:20UT,         82.03,          Keep,             Keep, False
------------------- 153 ---------------------------------------------------
 16,        CTIM, 2023-06-02T01:09:17UT, 2023-06-02T01:20:15UT,         36.96,          Keep,             Keep, False
 17,        CUTE, 2023-06-02T02:31:11UT, 2023-06-02T02:35:55UT,          1.90,       Delete,           Delete, False
 18,        CTIM, 2023-06-02T02:47:32UT, 2023-06-02T02:58:33UT,         40.78,          Keep,             Keep, False
 19,        CUTE, 2023-06-02T04:01:12UT, 2023-06-02T04:12:57UT,         43.76,          Keep,             Keep, False
 20,        CTIM, 2023-06-02T04:25:27UT, 2023-06-02T04:36:32UT,         60.04,          Keep,             Keep, False
 21,       CIRBE, 2023-06-02T04:54:38UT, 2023-06-02T05:04:35UT,         15.53,          Keep,           Delete, False
 22,        CUTE, 2023-06-02T05:36:55UT, 2023-06-02T05:46:47UT,         12.27,          Keep,           Delete, False
 23,        CTIM, 2023-06-02T06:03:55UT, 2023-06-02T06:12:39UT,         10.01,          Keep,           Delete, False
 24,       CIRBE, 2023-06-02T06:27:57UT, 2023-06-02T06:38:46UT,         29.44,          Keep,             Keep, False
 25,       CIRBE, 2023-06-02T15:33:22UT, 2023-06-02T15:40:48UT,          6.09, Keep_Conflict,          Delete, False
 26,        CUTE, 2023-06-02T15:38:38UT, 2023-06-02T15:45:45UT,          4.46, Delete_Conflict,          Delete, False
 27,       CIRBE, 2023-06-02T17:04:26UT, 2023-06-02T17:15:48UT,         84.25, Keep_Conflict,   Keep_Conflict, False
 28,        CUTE, 2023-06-02T17:11:16UT, 2023-06-02T17:23:26UT,         79.49, Delete_Conflict, Delete_Conflict, False
 29,       CIRBE, 2023-06-02T18:41:11UT, 2023-06-02T18:47:30UT,          3.74, Keep_Conflict,          Delete, False
 30,        CUTE, 2023-06-02T18:46:57UT, 2023-06-02T18:55:35UT,          8.11, Delete_Conflict,          Delete, False
 31,        CTIM, 2023-06-02T21:13:43UT, 2023-06-02T21:22:45UT,         11.03,          Keep,           Delete, False
 32,        CTIM, 2023-06-02T22:49:58UT, 2023-06-02T23:01:08UT,         63.93,          Keep,             Keep, False
------------------- 154 ---------------------------------------------------
 33,        CTIM, 2023-06-03T00:28:08UT, 2023-06-03T00:39:02UT,         40.10,          Keep,             Keep, False
 34,        CTIM, 2023-06-03T02:06:18UT, 2023-06-03T02:17:16UT,         37.43,          Keep,             Keep, False
 35,        CUTE, 2023-06-03T02:25:10UT, 2023-06-03T02:28:18UT,          0.72,       Delete,           Delete, False
 36,        CTIM, 2023-06-03T03:44:14UT, 2023-06-03T03:55:23UT,         77.95, Delete_Conflict, Delete_Conflict, False
 37,        CUTE, 2023-06-03T03:54:17UT, 2023-06-03T04:05:52UT,         35.97, Keep_Conflict,   Keep_Conflict, False
 38,       CIRBE, 2023-06-03T04:34:42UT, 2023-06-03T04:43:18UT,          8.29,          Keep,           Delete, False
 39,        CTIM, 2023-06-03T05:22:29UT, 2023-06-03T05:31:56UT,         13.70, Delete_Conflict,          Delete, False
 40,        CUTE, 2023-06-03T05:29:36UT, 2023-06-03T05:39:56UT,         14.93, Keep_Conflict,           Delete, False
 41,       CIRBE, 2023-06-03T06:07:18UT, 2023-06-03T06:18:35UT,         54.92,          Keep,             Keep, False
 42,       CIRBE, 2023-06-03T07:43:07UT, 2023-06-03T07:48:20UT,          2.55,          Keep,           Delete, False
 43,       CIRBE, 2023-06-03T15:14:35UT, 2023-06-03T15:19:11UT,          1.93,       Delete,           Delete, False
 44,        CUTE, 2023-06-03T15:32:07UT, 2023-06-03T15:38:02UT,          2.87,          Keep,           Delete, False
 45,       CIRBE, 2023-06-03T16:44:02UT, 2023-06-03T16:55:14UT,         49.43,          Keep,             Keep, False
 46,        CUTE, 2023-06-03T17:04:15UT, 2023-06-03T17:16:23UT,         64.39,          Keep,             Keep, False
 47,       CIRBE, 2023-06-03T18:19:15UT, 2023-06-03T18:27:57UT,          9.15,          Keep,           Delete, False
 48,        CUTE, 2023-06-03T18:39:43UT, 2023-06-03T18:48:56UT,          9.95,          Keep,           Delete, False
 49,        CTIM, 2023-06-03T20:33:02UT, 2023-06-03T20:41:11UT,          7.77,          Keep,           Delete, False
 50,        CTIM, 2023-06-03T22:08:46UT, 2023-06-03T22:19:51UT,         48.53,          Keep,             Keep, False
 51,        CTIM, 2023-06-03T23:46:38UT, 2023-06-03T23:57:44UT,         44.80,          Keep,             Keep, False
```

Schedule Display:
The current version of the schedule is printed to the terminal with specialized highlighting to point out errors.

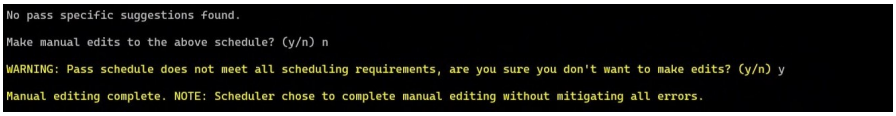
```
---------- 155 ----------
 52,        CTIM, 2023-06-04T01:24:58UT, 2023-06-04T01:35:54UT,         35.32,          Keep,           Keep, False
 53,        CTIM, 2023-06-04T03:02:56UT, 2023-06-04T03:14:06UT,         83.53,          Keep,           Keep, False
 54,        CUTE, 2023-06-04T03:47:22UT, 2023-06-04T03:58:47UT,         29.88,          Keep,           Keep, False
 55,       CIRBE, 2023-06-04T04:15:14UT, 2023-06-04T04:21:05UT,          3.09,          Keep,         Delete, False
 56,        CTIM, 2023-06-04T04:41:01UT, 2023-06-04T04:51:01UT,         18.21,          Keep,         Delete, False
 57,        CUTE, 2023-06-04T05:22:18UT, 2023-06-04T05:33:02UT,         18.02,          Keep,         Delete, False
 58,       CIRBE, 2023-06-04T05:46:44UT, 2023-06-04T05:58:08UT,         76.27,          Keep,           Keep, False
 59,       CIRBE, 2023-06-04T07:21:36UT, 2023-06-04T07:29:23UT,          6.89,          Keep,         Delete, False
 60,        CUTE, 2023-06-04T15:25:48UT, 2023-06-04T15:38:03UT,          1.38,        Delete,         Delete, False
 61,       CIRBE, 2023-06-04T16:23:50UT, 2023-06-04T16:34:31UT,         26.89,          Keep,           Keep, False
 62,        CUTE, 2023-06-04T16:57:14UT, 2023-06-04T17:09:17UT,         51.86,          Keep,           Keep, False
 63,       CIRBE, 2023-06-04T17:57:50UT, 2023-06-04T18:07:56UT,         16.86,          Keep,         Delete, False
 64,        CUTE, 2023-06-04T18:32:30UT, 2023-06-04T18:42:13UT,         12.01,          Keep,         Delete, False
 65,        CTIM, 2023-06-04T19:52:27UT, 2023-06-04T19:59:23UT,          4.92,          Keep,         Delete, False
 66,        CTIM, 2023-06-04T21:27:32UT, 2023-06-04T21:38:28UT,         36.76,          Keep,           Keep, False
 67,        CTIM, 2023-06-04T23:05:11UT, 2023-06-04T23:16:21UT,         51.66,          Keep,           Keep, False
---------- 156 ----------
 68,        CTIM, 2023-06-05T00:43:31UT, 2023-06-05T00:54:27UT,         34.22,          Keep,           Keep, False
 69,        CTIM, 2023-06-05T02:21:33UT, 2023-06-05T02:32:43UT,         67.76,          Keep,           Keep, False
 70,        CUTE, 2023-06-05T03:40:29UT, 2023-06-05T03:51:39UT,         25.04,          Keep,           Keep, False
 71,        CTIM, 2023-06-05T03:59:32UT, 2023-06-05T04:09:56UT,         23.94,          Keep,           Keep, False
 72,        CUTE, 2023-06-05T05:15:02UT, 2023-06-05T05:26:06UT,         21.69, Delete_Conflict, Delete_Conflict, False
 73,       CIRBE, 2023-06-05T05:26:15UT, 2023-06-05T05:37:25UT,         38.62, Keep_Conflict,   Keep_Conflict, False
 74,        CTIM, 2023-06-05T05:40:03UT, 2023-06-05T05:43:16UT,          0.88,        Delete,         Delete, False
 75,       CIRBE, 2023-06-05T07:00:29UT, 2023-06-05T07:09:52UT,         12.77,          Keep,         Delete, False
 76,       CIRBE, 2023-06-05T16:03:51UT, 2023-06-05T16:13:39UT,         15.73,          Keep,         Delete, False
 77,        CUTE, 2023-06-05T16:50:12UT, 2023-06-05T17:02:09UT,         42.02,          Keep,           Keep, False
 78,       CIRBE, 2023-06-05T17:36:44UT, 2023-06-05T17:47:39UT,         29.91,          Keep,           Keep, False
 79,        CUTE, 2023-06-05T18:25:17UT, 2023-06-05T18:35:27UT,         14.34,          Keep,         Delete, False
 80,        CTIM, 2023-06-05T19:12:06UT, 2023-06-05T19:17:13UT,          2.36,          Keep,         Delete, False
 81,        CTIM, 2023-06-05T20:46:17UT, 2023-06-05T20:56:59UT,         28.00,          Keep,           Keep, False
 82,        CTIM, 2023-06-05T22:23:41UT, 2023-06-05T22:34:54UT,         61.54,          Keep,           Keep, False
---------- 157 ----------
 83,        CTIM, 2023-06-06T00:01:59UT, 2023-06-06T00:12:56UT,         34.05,          Keep,           Keep, False
 84,        CTIM, 2023-06-06T01:40:05UT, 2023-06-06T01:51:13UT,         56.01,          Keep,           Keep, False
 85,        CTIM, 2023-06-06T03:18:00UT, 2023-06-06T03:28:42UT,         31.50,          Keep,           Keep, False
 86,        CUTE, 2023-06-06T03:33:36UT, 2023-06-06T03:44:29UT,         21.09,          Keep,           Keep, False
 87,        CTIM, 2023-06-06T04:57:27UT, 2023-06-06T05:03:18UT,          3.30, Delete_Conflict,         Delete, False
 88,       CIRBE, 2023-06-06T05:05:53UT, 2023-06-06T05:16:26UT,         21.35, Keep_Conflict,   Keep_Conflict, False
 89,        CUTE, 2023-06-06T05:07:46UT, 2023-06-06T05:19:08UT,         26.13, Keep_Conflict,   Keep_Conflict, True
 90,       CIRBE, 2023-06-06T06:39:32UT, 2023-06-06T06:49:59UT,         21.76,          Keep,           Keep, False
 91,       CIRBE, 2023-06-06T15:44:09UT, 2023-06-06T15:52:35UT,          8.90,          Keep,         Delete, False
 92,        CUTE, 2023-06-06T16:43:11UT, 2023-06-06T16:54:58UT,         34.38,          Keep,           Keep, False
 93,       CIRBE, 2023-06-06T17:15:54UT, 2023-06-06T17:27:11UT,         57.54,          Keep,           Keep, False
 94,        CUTE, 2023-06-06T18:18:05UT, 2023-06-06T18:28:38UT,         17.03,          Keep,         Delete, False
 95,       CIRBE, 2023-06-06T18:53:59UT, 2023-06-06T18:57:51UT,          1.24,        Delete,         Delete, False
 96,        CTIM, 2023-06-06T20:05:01UT, 2023-06-06T20:15:22UT,         21.37,          Keep,           Keep, False
 97,        CTIM, 2023-06-06T21:42:06UT, 2023-06-06T21:53:22UT,         75.33,          Keep,           Keep, False
 98,        CTIM, 2023-06-06T23:20:21UT, 2023-06-06T23:31:19UT,         34.78,          Keep,           Keep, False
---------- 158 ----------
 99,        CTIM, 2023-06-07T00:58:31UT, 2023-06-07T01:09:37UT,         47.75,          Keep,           Keep, False
100,        CTIM, 2023-06-07T02:36:24UT, 2023-06-07T02:47:19UT,         41.79,          Keep,           Keep, False
101,        CUTE, 2023-06-07T03:26:43UT, 2023-06-07T03:37:18UT,         17.79,          Keep,         Delete, False
102,       CIRBE, 2023-06-07T04:45:42UT, 2023-06-07T04:55:06UT,         12.00,          Keep,         Delete, False

MAGENTA TEXT  = Passes with ATS downlinks marked as SBD Delete
YELLOW TEXT   = Passes with conflicting overlap
RED TEXT      = Passes with ATS downlinks marked as SBD Delete and conflicting overlap
RED HIGHLIGHT = Suggests priority be switched
```

```
Suggest manually editing passes at the following indicies: [89]

Make manual edits to the above schedule? (y/n) y

6. Running manual deconflicter for user to make changes as needed...

Please select the index above to manually edit, or -1 when done: 89

Chosen Pass:
Index, Satellite,           Start Time,             End Time, Peak Elevation, UHF Priority, S-Band Priority, Manually Edited
 89,        CUTE, 2023-06-06T05:07:46UT, 2023-06-06T05:19:08UT,         26.13, Keep_Conflict,   Keep_Conflict, True


Index 89: CUTE at 2023-06-06 05:07:46.979750 has UHF Priority = Keep_Conflict. Suggest changing priority.
Would you like to change the priority? (y/n) y
       Changing      CUTE, 2023-06-06T05:07:46UT, 2023-06-06T05:19:08UT, 26.13 pass uhf_pri from Keep_Conflict to Delete_Conflict


Index 89: CUTE at 2023-06-06 05:07:46.979750 has SBD Priority = Keep_Conflict. Suggest changing priority.
Would you like to change the priority? (y/n) y
       Changing      CUTE, 2023-06-06T05:07:46UT, 2023-06-06T05:19:08UT, 26.13 pass sbd_pri from Keep_Conflict to Delete_Conflict

Please select the index above to manually edit, or -1 when done: -1
```

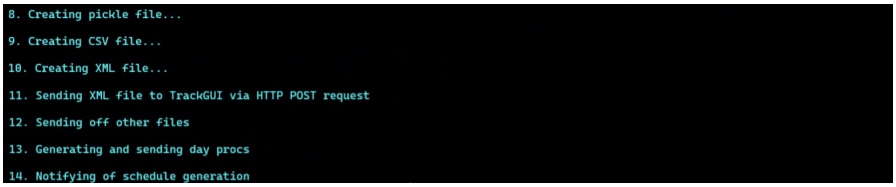
Step 6:
Suggestions for edits are displayed in the terminal and the users is prompted to begin manual editing. The specific index of the schedule chosen for manual editing is displayed with all suggestions.

```
7. Checking manually edited schedule against scheduling requirements...

    a. Checking schedule for overlap...
    PASS: NO conflicts detected within timerange.

    b. Checking schedule against mission requirements...
        i.   Checking downlinks per day requirements for each mission...
            WARNING: CUTE does NOT meet the minimum number of downlinks per day (2/day) on doy 152!
            WARNING: CUTE does NOT meet the minimum number of downlinks per day (2/day) on doy 153!
            WARNING: CIRBE does NOT meet the minimum number of downlinks per day (2/day) on doy 158!
            WARNING: CUTE does NOT meet the minimum number of downlinks per day (2/day) on doy 158!
            WARNING: CTIM does NOT meet the minimum number of downlinks per day (3/day) on doy 158!
        FAIL: Pass schedule does not meet downlinks per day requirements for each mission.

        ii.  Checking time between downlinks requirements for each mission...
            WARNING: CUTE does NOT meet the minimum number of hours between downlinks (14 hours) around 2023-06-01 04:08:07.946468!
            WARNING: CUTE does NOT meet the minimum number of hours between downlinks (14 hours) around 2023-06-02 04:01:12.312492!
        FAIL: Pass schedule does not meet time between downlinks requirements for each mission.

        iii. Checking pass length requirements for each mission...
        PASS: Pass schedule meets pass length requirements for each mission.

        iv.  Checking uhf elevation requirements for each mission...
        PASS: Pass schedule meets uhf elevation requirements for each mission.

        v.  Checking sbd elevation requirements for each mission...
        PASS: Pass schedule meets sbd elevation requirements for each mission.

    FAIL: Pass schedule does not meet all mission requirements.

    c. Checking schedule against ats downlinks...
    PASS: All passes with ATS downlinks have been marked SBD keep/keep_conflict.

FAIL: Pass schedule does not meet all scheduling requirements, suggest manually deconflicting to mitigate errors.
```

Step 7:
The edited schedule is checked for overlaps, mission requirements, and ATS downlink times. Errors found by these checks are printed to the terminal for reference.



Error Message:
The user is once again prompted to manually edit, after the schedule is displayed. If the user chooses not to manually edit while errors are still present, they are once again prompted to return to scheduling. If they still choose not to, a message will print to the terminal noting this choice.



Step 8:
The schedule data is converted into a .pkl file.
Step 9:
The schedule data is converted into a .csv file.
Step 10:
The schedule data is converted into a .xml file.
Step 11:
The .xml schedule file is sent to TrackGUI via HTTP POST request where the passes are automatically ingested.
Step 12:
The .pkl schedule file and the .csv schedule file are sent to various other locations for different scripts to access.
Step 13:
The dayprocs are generated for each mission and sent to that missions corresponding operations computer.
Step 14:
The ground station and smallsat missions operations teams are notified of the creation of the schedule.

During the execution of this script, all pertinent information is logged in the schedule log file for that week, and all error messages are additionally logged in the schedule error log file for that week. This ensures that each weeks scheduling process is well documented, allowing for reference and analysis later down the road.