

# Long Term Evolution of a Planning and Scheduling Capability for Real Planetary Applications

Principal Author

William "Curt" Eiggemeyer (curt@eraserhead.jpl.nasa.gov)

Co-Authors

Sven U. Grenander (sven@sven.jpl.nasa.gov)

Stephen F. Peter-s (stevep@stevep.jpl.nasa.gov)

Arthur V. Amador (arthur.v.amador@jpl.nasa.gov)

Sequence Automation Research Group (SARG), Section 314  
Jet Propulsion Laboratory, California Institute of Technology  
Mail Stop 301 -2501  
4800 N. Oak Grove Dr.  
Pasadena, Ca. 91109

## Abstract

Plan-IT-II [1,2,3,4,5,6] represents a powerful capability for planning and scheduling of activities to be executed by interplanetary spacecraft. Its development and gradual acceptance by conservative flight projects spans 16 years after a couple of false starts in the prior 3 years. The development has been both evolutionary and revolutionary as lessons have been absorbed and major rewrites have been warranted. The application and adaptation of Plan-IT over a dozen mission domains has resulted in a representation which is rich enough to encompass virtually any interplanetary mission while at the same time allowing its use for ground-based manual, or fully autonomous on-board sequencing. Its application to fully autonomous sequencing domains has demonstrated that the next challenge in its development will be to provide user hooks which ease its use in developing and beta testing strategies or heuristics which will perform as desired under actual mission circumstances.

## Introduction

This paper describes the various Plan-IT efforts we have been involved with over the years and concludes with some lessons learned and recommendations for future enhancements. Plan-IT was originally developed to "enhance" the performance of sequencing/scheduling personnel for spacecraft missions. Immediately, we became aware of the need to represent the problem to the user in a visual manner (Gantt chart style timeline) while minimizing the user's need to understand what the tool was displaying to them. Additionally, the tool needed to interact with users on their own terms, for example, "move this activity here," or "move this activity anywhere else," or "shrink this activity to fit between these activities," etc. In addition, the tool needed to have the ability to be quickly adaptable while having the representational capacity to address complex

sequencing problems. Finally, the tool needed to react quickly to user inputs.

SpaceLab, Space Station Power Scheduling proof of Concept [7], Deep Space Network (DSN) Application [8], and the Comet Rendezvous Asteroid Flyby (CRAF) Demonstration [9] were all performed with the first major version of Plan-IT. The BOS demo [12, 13], Galileo, Mars Pathfinder, Autonomous Nav-Sequencing, Distribution and Automation Technology Advancement of the Colorado Hitchhiker and Student Experiment of Solar Radiation (Data-Chaser), Microspacecraft, Distributed Object Interface demonstration, SIRTIF demonstration, and the Cassini team personnel plan were done with the major revision of Plan-IT called Plan-IT-II. Currently, we are in the midst of adapting Plan-IT-II for New Millennium's Deep Space One Mission. The following descriptions of the various Plan-IT adaptations are in a general chronological order starting in the early 80's, although several of these efforts were done concurrently.

## Space Lab

The SpaceLab effort required Plan-IT to operate within an already existing scheduling system, called Experiment Scheduling Program (ESP). Plan-IT's task for this application was to give the user the ability to tweak an already existing schedule either graphically, by manual edits, or by algorithmic strategies specially coded for the SpaceLab problem domain. This task illustrated the usefulness of the timeline GUI for dealing with scheduling problems.

## Space Station Power Scheduling Proof of Concept

The Space Station Power Scheduling demonstration was one of the first successes of Plan-IT's approach. This ap-

plication required **Plan-IT** work with simple prioritized activities and real-time dynamic changes to update the schedule as changes occurred during its execution.

## Deep Space Network (DSN) Application

Plan-IT was adapted in six months for scheduling the allocation of DSN antennas around the world. Plan-IT enhancements developed for this problem domain included casing the user edits, handling of generic as well as specific requests, and specialized algorithmic strategies. Plan-IT was used as an interim solution to the DSN scheduling problem until the Resource Allocation Planning Helper (RAI.PH) system development was completed. Plan-IT successfully demonstrated its capabilities by reducing the DSN turn-around time for scheduling by two orders of magnitude over existing manual scheduling methods.

## Comet Rendezvous Asteroid Flyby (CRAF) Demonstration

This was the first successful demonstration of Plan-IT's application to deep space missions. Additionally, Plan-IT was combined with a natural language understanding system [10], enabling Plan-IT to take requests for the spacecraft in English form and translate them into activities which were then scheduled. This demonstrated to JPL management that such a "user-natural" scheduling system could make significant contributions to the spacecraft command and control process.

## Earth Observing System (EOS)

Plan-IT was used to prototype two different types of nodes in the EOS distributed planning and scheduling system: the control center for the single complex instrument, and a remote science scheduling tool for that instrument. The prototypes were based upon science requests and specifications for the Advanced Spaceborne Thermal Emission Reflectance Radiometer (ASTER) instrument and for the AM 1 platform.

For the ASTER Instrument Control Center (ICC) prototype, a scheduling algorithm incorporating nominal scheduling policy guidelines was encoded within Plan-IT. Interfaces were added which coordinated schedules with the EOS Control Center at the Goddard Space Flight Center, and accepted specific observation requests from a remote science workstation at the University of Colorado at Boulder. The ICC prototype was the only node in the system which provided full visibility into all constraints affecting, and conflicts involving, the ASTER instrument. A new averaging

resource was added to Plan-IT to support modelling of mission guidelines.

For precise science planning, the ASTER science scheduling tool prototype modelled both "soft" science constraints (desires) and "hard" instrument and spacecraft constraints. A database of science requests and all target visibility opportunities was added to Plan-IT. A new request display showed the set of requests involving targets which could be observed over a specified interval of time. Using this display, the scientist could compare the requests competing for spacecraft resources over that interval. A new opportunities display showed data about specific observable targets over a time interval and provided an interface for subsetting these choices by telescope, pointing angle, original request, target, or individual opportunity. Using this interface, the scientist could assert the desire for specific observations in the scheduling timeline. Instrument activities satisfied instrument, spacecraft, and asserted science observation constraints. These prototypes contributed to the requirements for the EOS ground system.

## Galileo (GLL)

In 1995, Plan-IT-II was adapted for the Galileo Mission to Jupiter as part of the redesign of both flight and ground systems to work around the partial deployment of the spacecraft's high gain antenna. The absence of a functional high gain antenna meant that the mission's data would have to be returned at significantly lower data rates. Flight software was redesigned to employ data editing, compression and buffering to deal with the high data rates produced by the spacecraft's instruments. In Figure 1 on the next page is Galileo's uplink data flow.

Plan-IT-II allowed science planners to model the data production rates of the various onboard instruments as well as the recording, editing, compression and buffering of the data by the spacecraft's main flight software. By using Plan-IT-II, users were able to adjust controls on the spacecraft's instruments and in the main onboard computer to match the production of data with the spacecraft's changing downlink bandwidth capability.

As shown on next page in Figure 2, Plan-IT-II was used to produce sequence planning inputs to the Uplink System's sequence generation software, Seqgen. By modeling the sequence used to command Galileo's tape recorder, Plan-IT-II produced a map of the data that was recorded on the tape recorder. This map was used to create a table of playback directives used by the onboard flight software to select and compress data from the tape recorder before placing it in the spacecraft's downlink buffer.

Plan-IT-II was also used to model the process of playing back encounter sequence data recorded on Galileo's tape

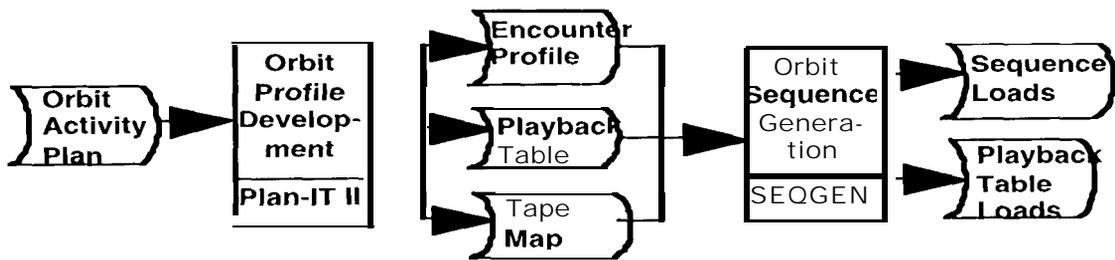


Figure 1: Galileo's Uplink Data Flow

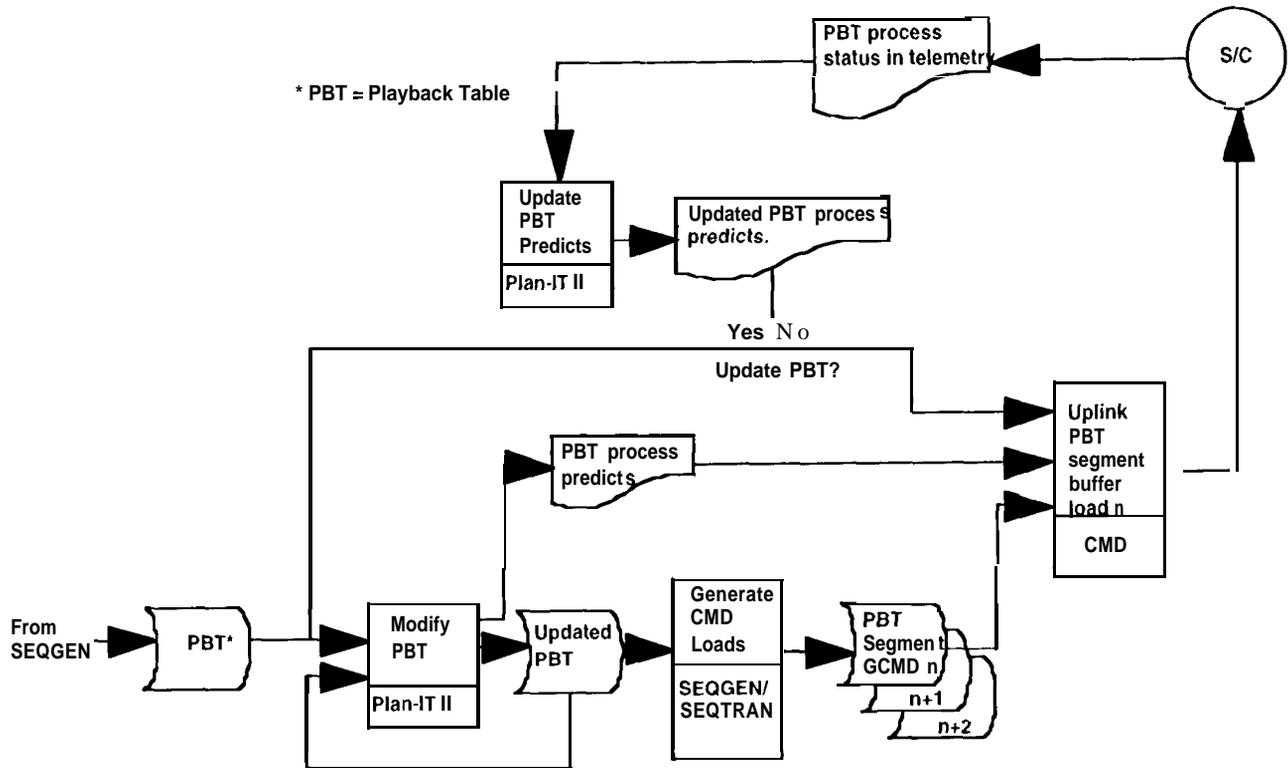


Figure 2: Playback Table Sequencing inputs

recorder. By using the tape recorder map it produced and the table of playback directives, Plan-IT-II was able to predict the volume of data to be returned for the various onboard instruments as well as the schedule of when the data was expected to arrive on the ground. As a playback sequence progressed, actual performance, i.e., actual arrival time of data on the ground and actual data volume, was used to adjust Plan-IT-II's predicts for subsequent playbacks. If required, a new table of playback directives could be generated by Plan-IT-II for uplink to the spacecraft as illustrated in the above figure.

### Cassini Team Projection Study

A short term effort (approximately two weeks) used Plan-IT-II as a forecasting tool for predicting the staffing of Cassini flight teams necessary in order to support various kinds of operations for the project during flight. Once Plan-IT-II was adapted in a couple of days, a naive user could apply it to calculate needed personnel support for various operational modes during the mission. This was a useful learning experience for our team, because we learned that if a naive user gets closely monitored tutorials to learn a subset of the Plan-IT-II commands (in this case over the course of three days), they were able to learn the tool adequately.

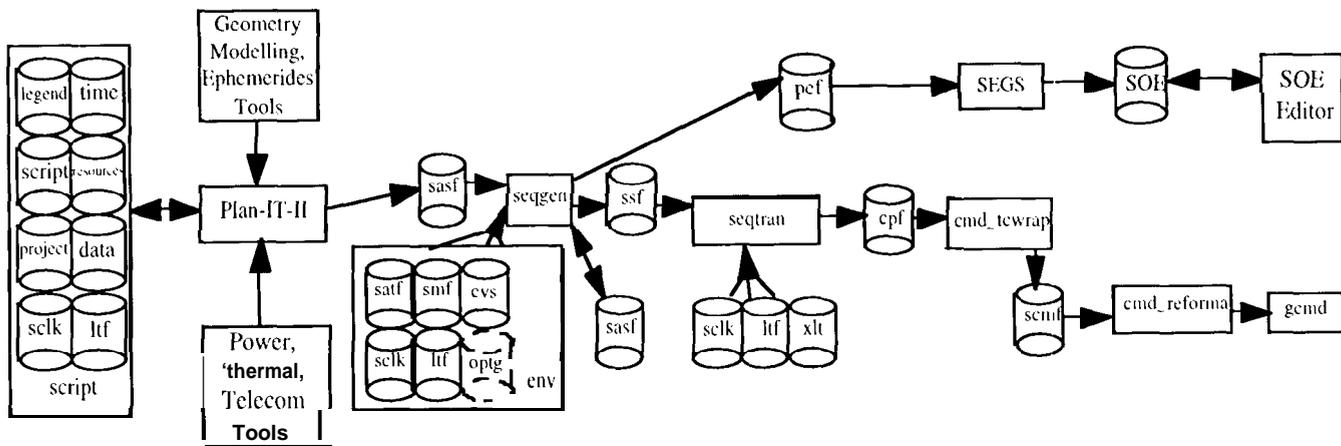


Figure 3: MPF Uplink Process Flow Through Its Tools

### Mars Pathfinder (MPF)

Unlike the Galileo effort, the MPF adaptation of Plan-IT-II was used during the design of the mission as well as for operations. Plan-IT-II was used to generate many surface scenarios to determine the viability of the mission given the rover anti lander designs. In fact, MPF added a sensor anti almost added an additional battery as a result of running these various scenarios through Plan-IT-II. Due to MPF's relatively short development time, the project exploited Plan-IT-II's rapid adaptation capabilities. In addition, the lone MPF adapter designed and coded specialized scheduling heuristics in Plan-IT-II during this period of adaptation and scenario generation in order to ease MPF's activity sequence generation process. Both the rover and lander are modeled to the command level. Plan-IT-II's generated Spacecraft Activity Sequence file (SASF) is validated by another tool, Seqgen. However, since not all commands were represented in Plan-IT-II, those commands not often used were specified during the Seqgen run. MPF also had additional analysis tools that dealt with telecom, power, thermal, ephemerides, and rover geometry modeling which worked in conjunction with the telemetry, power, battery, and thermal models within Plan-IT-II. MPF was the first project in which external tools played a role in how some of Plan-IT-II's resource constraint models were updated.

The uplink process flow illustrates that Plan-IT-II and Seqgen shared the responsibility for generating the sequences for flight. As the level of activity on the spacecraft was low during the flight to Mars, Seqgen was used exclusively for generating the spacecraft sequences during this period. However, intensive operations on the surface of Mars were planned and generated using Plan-IT-II.

### Autonomous Navigation (AutoNav) Demonstration

This was a demonstration that illustrated the potential use of auto-navigation for fire-and-forget deep space missions. Plan-IT-II's case of adaptation made this scenario possible within a month. The scenario was a near encounter sequence with the asteroid, Melpomene. Plan-IT-II acted as the onboard planner/scqucnccr that communicated with two other intelligent subsystems, an auto-navigator and an instrument observation analyzer. The Auto-Nav system was written in Matlab and the observation analyzer was a tool called Seq\_Pointer. Plan-IT-II modeled a simple imaging system, tape recorder, DSN tracking passes, power, and gyros. The sequence consisted of simple activities that represented optical navigation images, engineering, slewing, playbacks of the recorded data over downlink passes, and science image observations that could either be single or mosaic images. Plan-IT-II utilized a simple heuristic whose goal was to keep the spacecraft as busy as possible utilizing a dynamic priority scheme for the various activities that it knew how to execute. However, we did not have time to incorporate any fault recovery capabilities in this demonstration. Plan-IT-II generated the sequence on-the-fly just ahead of real-time execution as the spacecraft progressed on its trajectory.

The execution loop consisted of the following: 1) Plan-IT-II would query the navigator for optical navigation image (opnav) requests for insertion into the sequence beyond the point where the sequence segment had been frozen. This request interval would vary in duration as the frozen segment did depending on where the spacecraft was in the trajectory; 2) Navigation would reply with its list of requests specifying the target ids and their coordinates in RA and DEC; 3) Plan-IT-II would communicate the series of observations that were not presently in conflict to the

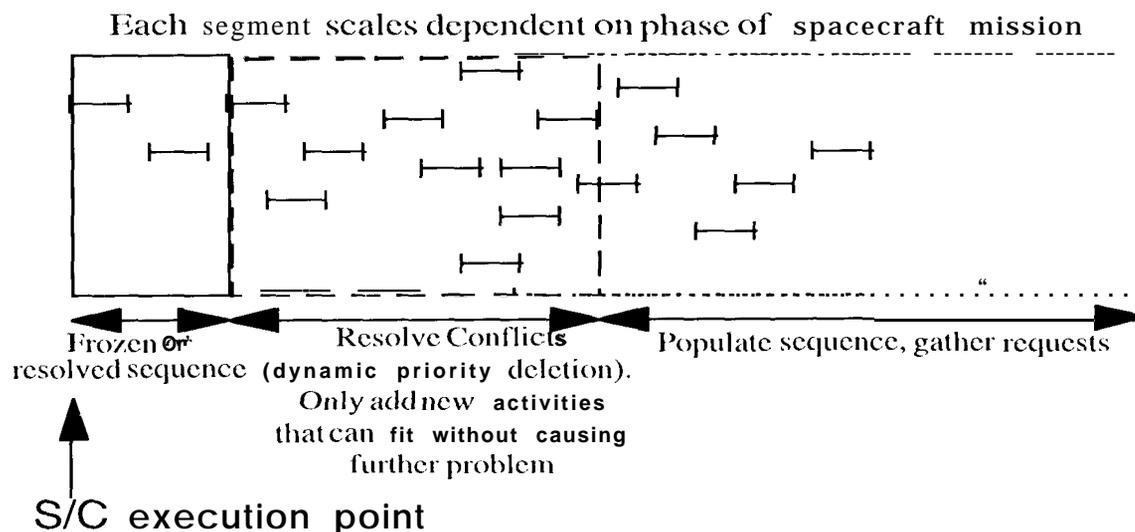


Figure 4: AutoNav Dynamic Scheduling

observation analyzer in order to determine if the image consisted of a single snapshot or required a mosaic of snapshots; 4) the observation analyzer would reply with the snapshot requirements for the observations; 5) Plan-IT-II would then adjust the activity durations for slewing and multiple snapshots if necessary; 6) if conflicts occurred, Plan-IT-I would not waste time doing a focused fix to the schedule but would do a simple priority delete; 7) when Plan-IT-II advanced the execution time bar and an activity requiring imaging was executed, Plan-IT-II would communicate with the observation analyzer to display that observation; X) during steps 1-7, Plan-IT-II would intensively attempt to interleave playbacks (if DSN tracks were available), with engineering activities.

Each time Plan-IT-II would execute the same portion of the trajectory a different sequence of activities would result. This simple approach resulted in reasonable near encounter sequences. The communication handshake between the three major subsystems was the greatest performance impact to the system. However, we were able to generate sequences at this activity level more quickly than real-time (5 day encounter sequence would be generated and executed in 10 minutes).

### Distribution and Automation Technology Advancement of the Colorado Hitchhiker and Student Experiment of Solar Radiation (Data-Chaser)

Plan-IT-II participated with the IMA-Chaser experiment, which involved using several automation technologies with a three instrument payload package that operated from the

Shuttle bay. The CHASER instrument gathers data from the UV, X-Ray, Lyman-Alpha wavelengths. Plan-IT-II was linked with a prototype automation tool that eventually became ASPEN (A Scheduling and Planning Environment). This complete system was called DCAPS (DATA-CHASER Automated Planner/Scheduler).

Plan-IT-II provided the graphical user interface, modelling of resource constraints, and the types of activities that would be executed to accomplish the CHASER objectives. Other constraints imposed on the CHASER instrument by the shuttle were also modelled. The ASPEN prototype was a general planning anti heuristic search engine. For this experiment DCAPS would generate a first cut at the sequence of activities to be performed by CHASER. The user would make "tweaks" or manual edits to this sequence. DCAPS would then be iteratively run improving the sequence interleaved with these user edits. This was the first time a general planning anti heuristic search engine tool was overlaid on top of Plan-IT-II's representation.

### MicroSpacecraft Demonstration

This demonstration was built upon the AutoNav demonstration by interfacing Plan-IT-II with a prototype spacecraft sequence executive (that controlled the execution clock) and simulator. Plan-IT-II was given a simple set of commands that it would pass on to the sequence executive in order for the simulator to run. In addition, another intelligent subsystem that recognized features on asteroids and could track them was interfaced to the system. Plan-IT-II played the role of a sequence generator (as before with AutoNav), except during closest approach with the asteroid, it would hand off commanding control to AFAST and after closest approach would resume its command sequence gen-

eration process. This is a low level effort that is still ongoing.

## Distributed Object Interface

Planning and scheduling of spacecraft activities often involves the iterative use of multiple software tools that deal with different kinds of domain knowledge. In order to facilitate coordinated use of these tools, Plan-IT-II and a geometrical observation planning tool were interfaced using the new Common Object Request Broker Architecture (CORBA) standard. Observations created in the observation planner were loaded into Plan-IT-II, causing the schedule and various resource constraints being modelled to change. Changes made to the observations in Plan-IT-II were immediately reflected back in the geometric display of the observation planner.

## Spacecraft Infrared Telescope Facility (SIRTF)

This was another short term effort (about two weeks total) in which we demonstrated one scheduling approach that SIRTF could use in scheduling their use of a remote telescope. The problem given to us made the following assertions: 1) treat the science requests coming into the schedule as being about 300% oversubscribed; 2) rely on another tool to determine the time necessary for accomplishing the science observation (based on the instrument, the operational mode, and a set of between 5 to 30 varying attributes) [NOTE: this time included enough quiet time to account for slewing]; 3) model data storage, playback to the earth, power, and gyro maintenance; 4) split some observations into a series of cumulative observations; and 5) schedule the observations in a priority-based manner. We set Lrp a week's worth of approximately 350 requests with two 4-hour DSN coverage periods per day. Given this set of inputs Plan-IT-II was able to generate a conflict free schedule within two minutes. Plan-IT-II's scheduling heuristic for this problem domain took about two days of effort to develop.

## Deep Space One (DS1)

Due to the recent redesign of the of the DS1 mission, development time for the ground system is shorter than any other deep space mission project (less than a year). Plan-IT-II has been demonstrated over the years to quickly adapt to various problem domains. Because of the previous successes with Plan-IT-II, the DS1 ground team chose it as the tool to perform the planning and sequencing job for this mission.

Plan-IT-II is presently being adapted as the driving hub of the uplink system for the New Millennium Deep Space One (DS1) project. Plan-IT-II will be used in two phases of the DS1 mission. The first phase is the mission planning, for which Plan-IT-II will be used to generate a high level or abstract cut of the mission scenarios needed in order to accomplish the DS1 objectives. During this phase, Plan-IT-II's modeling of the spacecraft and mission constraints will actually evolve as the operational phase of the mission approaches. The resultant output of the Mission Phase will feed into the actual operations phase of the mission. During the operational phase of the mission, Plan-IT-II will be used to generate the actual sequences of commands that will be uplinked to the spacecraft. In this respect, the ground team can exploit one of Plan-IT-II's capabilities that permits the linking of actual command level relationships to already generated sequences of abstract activities at a later time of mission development. Plan-IT-II's modeling of the spacecraft as well as the constraints imposed by the mission will be extremely detailed during the operational phase of the mission in order to insure the viability of the sequences that will be sent to the spacecraft.

In the figure on the following page, Plan-IT-II will act as a driver for the whole uplink process. Plan-IT-II's job is to coordinate the incoming requests and output various products that can be fed into the gen\_command part of the system. Gen\_command is a batch-oriented system with the job of translating the commands from human readable form into binary form for uplinking to the spacecraft. In addition, other utilities will be used to package files properly for uploading to the spacecraft. Plan-IT-II will feed and direct both gen\_command and these file uploading utilities. Plan-IT-II feeds the various sequences, and real-time commands to gen\_command in the form of a series of Spacecraft Activity Sequence Files (SASF) that can be processed by Seqgen, or Spacecraft Sequence Files (SSF) to be processed by the sequence translation process, "sline" (seqtran\_2000). The reason we have this two-fold submission process is for validation testing. Seqgen is a recognized event and model simulation tool used by spacecraft projects that has undergone rigorous testing and validation. Seqgen readily adapts for checking the commands and their argument syntax, but, requires extensive adaptation for performing the actual modeling of the spacecraft. Since Plan-IT-II has not undergone the same rigorous testing and validation, the ground team will use Seqgen to validate the syntax of the commands and their arguments. However, all of the modeling and constraint enforcement will be handled within Plan-IT-II. Once testing and validation is completed, the uplink system will be shortened by eliminating Seqgen from the process.

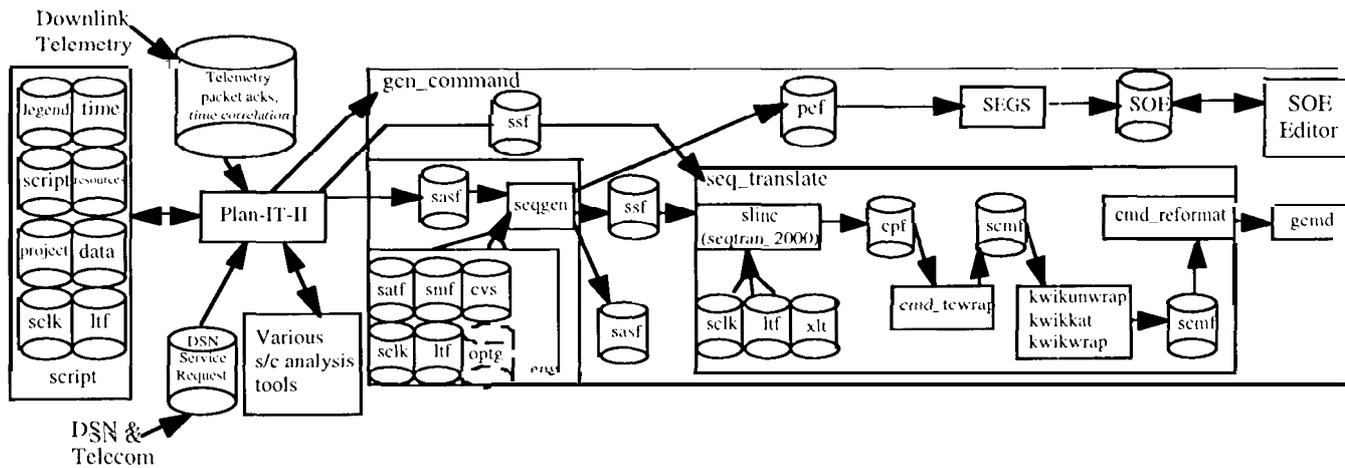


Figure S: DSISequence Process Flow through the tools

During spacecraft operations, requests come from several sources in several different forms. The downlink telemetry system produces “data received” acknowledgments and time correlation updates that must be fed into the next uplink pass to the spacecraft. Additionally, Plan-IT-II will receive from the DSN and telecom teams the view period and station allocations scheduled for DSI in the form of service requests. From the ops, engineering, and science teams Plan-IT-II integrates requests into the sequences. Plan-IT-II’s integration consists of: 1) detailing the requests in the form of legal commands; 2) insuring the mix of requests doesn’t violate any spacecraft models or constraints; 3) generating one or more sequences of these commands in a single run; and finally 4) generating the necessary products before invoking the remaining uplink system. DSI is the first project in which Plan-IT-II not only defines the products that are sent to the spacecraft, but in which Plan-IT-II is also responsible for insuring the viability of these products.

### Spin-Offs

Plan-IT-II has had the challenge of overcoming the LISP phobia of conservative, risk-averse management, so, C++ efforts were pursued to renovate and build a new tool upon the Plan-IT concept. When Seqgen [4] was ported from a mainframe system that Galileo used to UNIX workstations, its GUI’s conceptual origins came from Plan-IT. Another tool called Activity Plan Generator (APGEN) [5] that incorporates a subset of Plan-IT-II’s capabilities was developed with a focus on making the adaptation effort and GUI more friendly to naive users. Both of these tools are in use by various projects.

### Conclusions

Plan-IT owes its extensive capabilities to the years of experience and knowledge that have been incorporated as a result of being applied to a variety of tough real-world problems with extensive complex interactive modeling constraints and activity requests. Many lessons were learned and incorporated as updates and even complete revisions were made to the tool. Our main success lesson learned was to have the internal representations of the tool work in a manner that minimizes the amount of visualization required by the user in order to understand what the tool is doing when it is executing. Even though we have had success illustrating the rapid coding of specialized scheduling/sequencing heuristics for different problem domains, the following two areas need to be explored.

1. Review how the resource constraint models should rate themselves as they respond to activity requests search. Presently, we are using the same subjectively determined rating values we defined at the beginning of Plan-IT’s development. This is important for the future use of Plan-IT in an onboard spacecraft application.

2. Simplify the scheduling heuristic development so naive users can accomplish it on their own. Unfortunately, Plan-IT “wizards” are still required to generate more sophisticated scheduling heuristics. If the representation within the tool could be enhanced to monitor how the user interacts with the tool in deriving a sequence, the heuristics could be learned.