Automated Scheduling for NASA's Deep Space Network

Mark D. Johnston and Daniel Tran
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive, Pasadena CA USA 91109
{mark.d.johnston, daniel.tran} @jpl.nasa.gov

Abstract

This paper describes the DSN Scheduling Engine (DSE) component of a new scheduling system being deployed for NASA's Deep Space Network. The DSE provides core automation functionality for scheduling the network, including the interpretation of scheduling requirements expressed by users, their elaboration into tracking passes, and the resolution of conflicts and constraint violations. The DSE incorporates both systematic search and repair-based algorithms, used for different phases and purposes in the overall system. It has been integrated with a web application which provides DSE functionality to all DSN users through a standard web browser. The system is in the process of deployment to operational status.

1 Introduction

NASA's Deep Space Network (DSN) provides communications and other services for planetary exploration missions as well as other missions beyond geostationary orbit, supporting both NASA and international users. It also constitutes a scientific facility in its own right, conducting radar investigations of the moon and planets, in addition to radio science and radio astronomy. The DSN comprises three antenna complexes in Goldstone, California; Madrid, Spain; and Canberra, Australia. Each complex contains one 70m antenna and several 34m antennas, providing S-, X-, and K-band up and downlink services. The distribution in longitude enables full sky coverage and generally provides some overlap in spacecraft visibility between the complexes. A more detailed discussion of the DSN and it's large antennas can be found in (Imbriale 2003).

The process of scheduling the DSN is complex and time-consuming. There is significantly more demand for DSN services than can be handled by the available assets. There are numerous constraints on the assets and on the timing of communications supports, due to spacecraft and ground operations rules and preferences. Most DSN users require a firm schedule around which to build spacecraft command sequences, weeks to months in advance. Currently there are several distributed teams who work with missions and other users of the DSN to determine their service needs, provide these as input to an initial draft schedule, then iterate among themselves and work with the users to resolve conflicts and come up with an integrated schedule. This effort has a goal of a conflict-free schedule by eight weeks ahead of the present, which is frequently hard to meet in practice. In addition to asset contention, many other factors such as upcoming launches (and their slips) contribute to the difficulty of building up an extended conflict-free schedule.

There have been various past efforts to increase the level of scheduling automation for the DSN (Biefeld and Cooper 1991; Bell 1992; Loyola 1993; Werntz et al. 1993; Kan et al. 1996; Chien et al. 1997; Fisher et al. 1998; Guillaume et al. 2007). Currently, the DSN scheduling process is centered around the Service Preparation Subsystem (SPS) which provides a central database for schedules and for the auxiliary data needed by the DSN to operate the antennas and communications equipment (e.g. viewperiods, sequence-of-events files). The current project to improve scheduling automation is designated the Service Scheduling Subsystem, or S^3, which will be integrated with SPS. There are three primary features of S^3 that are expected to significantly improve the scheduling process:

1. Automated scheduling of activities with a *request-driven approach* (as contrasted with the current activity-oriented scheduling);
2. *Unifying the scheduling software and databases* into a single integrated suite covering realtime out through as much as several years into the future;
3. Development of a *peer-to-peer collaboration environment* for DSN users to view, edit, and negotiate schedule changes and conflict resolutions.

The second and third of these areas are described elsewhere (Carruth et al. 2010). This paper focuses on the first area and some of its ramifications — see also the following for additional information: (Clement and Johnston 2005; Johnston and Clement 2005; Johnston et al. 2009; Johnston et al. 2010).

The request-driven paradigm shifts the emphasis from individual specific resource allocations to a more abstract scheduling request specification or "language", and on the scheduling algorithms that work with this specification to generate, maintain, and improve the schedule. In the following sections, we first provide some background on the DSN scheduling problem (Section 2), and on the reasons for the request-driven approach taken by S^3. We then briefly describe the scheduling request specification itself (Section 3), which is how DSN users of S^3 will describe their service requests to the system. These requests are processed by the DSN Scheduling Engine (DSE, Section 4) to expand into tracking passes, integrate them into an overall schedule, all the while seeking to minimize conflicts and request violations. We conclude with an overall sum-

tracking passes per week	425
number of users (missions, science users, and maintenance)	35
average pass duration	5.25 hours
assets	13 antennas at 3 sites
asset loading	88.5%
scheduling timescale	• preview schedule 17-26 weeks ahead • conflict free 8 weeks ahead

Table 1. Some characteristics of the DSN scheduling problem.

mary and brief description of plans for future development (Section 5).

2 Overview of DSN Scheduling

The DSN antennas and supporting infrastructure are heavily used. Characteristics of the network's assets and typical usage are listed in Table 1. The DSN follows a collaborative approach to developing the conflict-free schedule. Users provide their requests which are integrated into an overall initial or "preview" schedule. This version of the schedule is extremely heavily overloaded, but it does indicate where there are contentious areas. These contentious areas shift from week to week depending on critical activities, as well as on the slow drift of visibility intervals with time.

The DSN users follow a peer to peer approach to resolving conflicts. Users create proposals which are suggestions as to how sets of users could modify their tracking passes to resolve conflicts. The affected users can concur or reject these suggestions, and counter with suggestions of their own. Over the course of a few weeks, convergence is reached and the schedule reaches a negotiated "conflict free" status. Should users not come to agreement among themselves, there is an escalation process to adjudicate irreconcilable conflicts that is very rarely used in practice.

3 DSN Scheduling Requests

DSN users represent their needs to the S^3 software system as *scheduling requests*. Each such request is interpreted by the DSN Scheduling Engine (DSE), as described below. The main elements of a scheduling request are:

- **Service specification**. S^3, via the DSE, provides an abstraction level on top of DSN asset specifications that may be referenced by users much more simply than specifying all of the possible options. At the physical level, the spacecraft onboard electronics (frequency band, data rates, encoding), radiated power, distance, along with the DSN antennas, receivers and transmitters and other equipment, determine what space and ground configurations are feasible. The abstraction level provided in S^3 is called a "service alias" such that a single service alias encapsulates a wide range of options, preferences, and associated information that is required to schedule the network. For example:
  - some users need the added sensitivity of more than one antenna at a time and so must be scheduled as antennas arrays using two or more antennas at once (as many as four at a time)
  - for navigation data, there are special ranging scenarios that alternate the received signal between the spacecraft and a nearby quasar, over a baseline that extends over multiple DSN complexes
  - for Mars missions, there is a capability for a single antenna to communicate with several spacecraft at once (called Multiple Spacecraft Per Aperture, or MSPA): while more than one at a time may be sending data to Earth, only one at a time may be uplinking

  A more detailed description of service alias functionality is provided in Section 4 below.
- **Timing constraints**. Users need a certain amount of communications contact time in order to download data and upload new command loads, and for obtaining navigation data. How this time is to be allocated is subject to many options, including whether it must be all in one interval or can be spread over several, and whether and how it is related to external events and to spacecraft visibility. Among the factors that can be specified in a schedule request are:
  - *reducible*: whether and how much the requested time can be reduced, for example to resolve conflicts
  - *extendable*: whether and how much the request time can be extended, should the option exist
  - *splittable*: whether the time must be provided in one unbroken track, or can be split into two or more separate tracks
  - *split duration*: if splittable, the minimum, maximum, and preferred durations of the split segments; the maximum number of split segments
  - *split segment overlap*: if the split segments must overlap each other, the minimum, maximum, and preferred duration of the overlaps
  - *split segment gaps*: if the split segments must be separated, the minimum, maximum, and preferred duration of the gaps
  - *viewperiods*: periods of visibility of a spacecraft from a ground station, possibly constrained to special limits (rise/set, other elevation limits), and possibly padded at the boundaries
  - *events*: general time intervals that constrain when tracks may be allocated; examples include: (a) day of week, time of day (for accommodating shift schedules, daylight, ...), (b) orbit/trajectory events (occultations, maneuvers, surface object direct view to Earth, …). Different event intervals may be com-

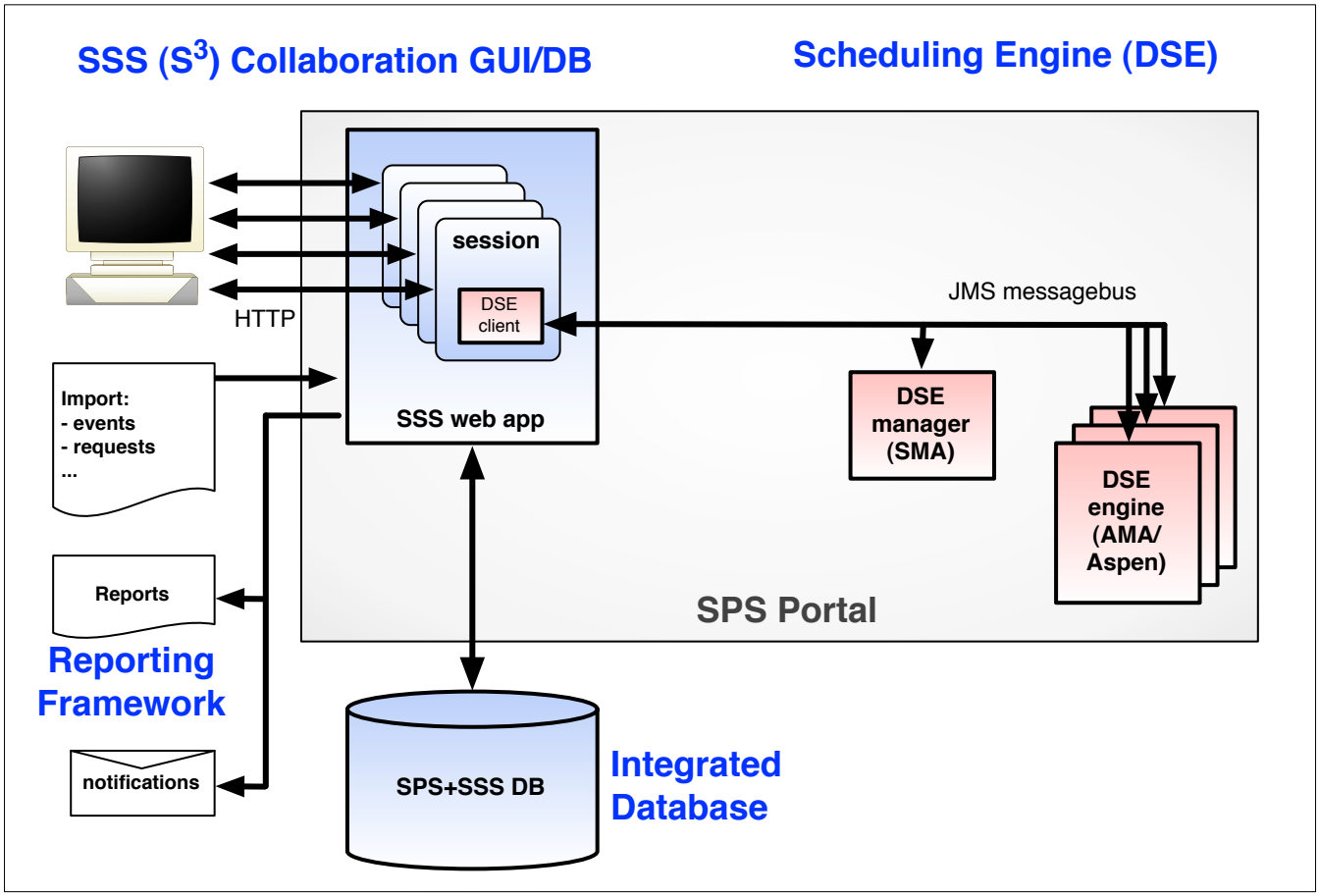


Figure 1. An overview of the S³ system architecture. The DSN Scheduling Engine (DSE) manages and provides a set of servers that respond to user's requests for scheduling services via the S³ web application.

bined (with optional inversion), and applied to a request.

- **Track relationships**. In some cases, contacts need to be sufficiently separated so that onboard data collection has time to accumulate data but not overfill onboard storage. In other cases, there are command loss timers that are triggered if the time interval between contacts is too long, placing the spacecraft into safemode. During critical periods, it may be required to have continuous communications from more than one antenna at once, so some passes are scheduled as backups for others.
- **Priority**. The DSN currently has a priority scheme which ranges from 1-7 with 7 being nominal tracking, and 1 representing a spacecraft emergency. Priority is relatively infrequently used, but it does have the effect that the scheduling engine will try to avoid conflicts with higher priority activities if possible. Depending on their degree of flexibility, missions trade off and compromise in order to meet their own requirements, while attempting to accommodate the requirements of others. As noted above, one of the key goals of S³ is to facilitate this process of collaborative scheduling.
- **Preferences**. Most preferences are incorporated in the service alias and timing requirements described above, but some are directly representable in the scheduling request. For example, users may choose to schedule early, centered, or late with respect to the viewperiod or event timing interval.

One characteristic of DSN scheduling is that, for most users, it is common to have repeated patterns of requests over extended time intervals. Frequently these intervals correspond to explicit phases of the mission (cruise, approach, fly-by, orbital operations). These patterns can be quite involved, since they interleave communication and navigation requirements. S³ provides for repeated requests, analogous to repeated or recurrent meetings in calendaring systems, in order to minimize the repetitive entry of detailed request information.

4 The DSN Scheduling Engine (DSE)

The DSN Scheduling Engine (DSE) is the component of S³ responsible for:

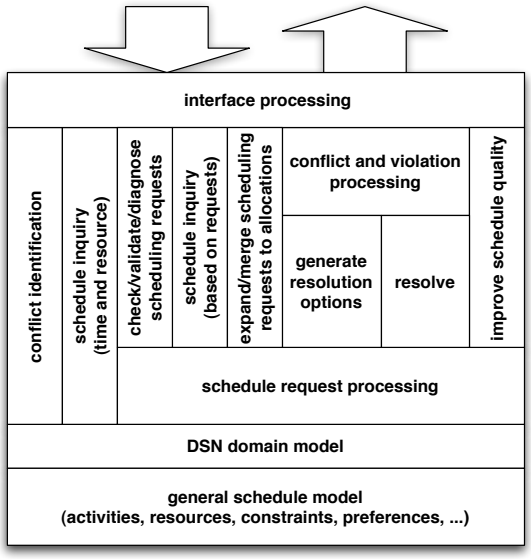


Figure 2. A block diagram of the DSE architecture.

- expanding scheduling requests into individual communications passes by allocating time and resources to each
- identifying conflicts in the schedule, such as contention for resources and any violations of DSN scheduling rules, and attempting to find conflict-free allocations
- checking scheduling requests for satisfaction, and attempting to find satisfying solutions
- identifying scheduling opportunities, based on resource availability and other criteria, or meeting scheduling request specifications
- searching for and implementing opportunities for improving schedule quality

Schedule conflicts are based only on the activity content of the schedule, not on any correspondence to schedule requests, and indicate either a resource overload (e.g. too many activities scheduled on the available resources) or some other violation of a schedule feasibility rule. In contrast, violations are associated with scheduling requests and their tracks, and indicate that in some way the request is not being satisfied. Conflicts and violations are permitted to exist in the schedule — both are identified by the scheduling engine, recorded in the S^3 database, and made visible to users working with the schedule. The scheduling engine provides algorithms to reduce or eliminate both conflicts and violations where possible, as described below. A block diagram of the DSE is shown in Figure 2.

Architecture

The DSE is based on ASPEN, the planning and scheduling framework developed at JPL and previously applied to numerous problem domains (Chien et al. 2000). In the S^3 application there may be many simultaneous scheduling users, each working with a different time segment or different private subset of the overall schedule. This has led us to develop an enveloping distributed architecture with multiple running instances of ASPEN, each available to serve a single user at a time (Figure 1). We use a middleware tier to link the ASPEN instances to their clients, via an ASPEN Manager Application (AMA) associated with each running ASPEN process. A Scheduling Manager Application (SMA) acts as a central registry of available instances and allocates incoming work to free servers. This architecture provides for flexibility and scalability: additional scheduler instances can be brought online simply by starting them up: they automatically register with the singleton SMA process, and are immediately available for use. Other features of this architecture include:

- Each AMA provides a heartbeat message to the SMA every few seconds; the absence of an AMA signal is detected as an anomaly, reported by the SMA, which can automatically start additional AMA instances to compensate
- To roll out new software versions or configuration changes, the SMA can automatically terminate AMA's when they become idle, then start up instances on the new version. This provides uninterrupted user service even as software updates are installed
- The SMA allocates free AMA instances to incoming clients, distributing work over all available host machines and thus balancing the load
- The SMA can be configured to automatically start additional AMA instances in case the base set on a host all become busy; in this way, service can gracefully degrade in that all users may see slower response times, but none are locked out of the system entirely.
- The SMA process can be restarted, e.g. to move it to another host, and upon starting up it will automatically locate and register all running AMA instances in the environment, without interrupting ongoing user sessions.

The DSE communicates with clients using an XML-based messaging protocol, similar in concept to HTTP sessions, but with session state maintained by one of the AMA servers, and with responses to time-consuming operations returned asynchronously. Each active user has one or more active sessions which has loaded all the data related to a schedule that user is working on. This speeds the client-server interaction, especially when editing scheduling requests and activities, when there can be numerous incremental schedule changes.

We have described elsewhere the main scheduling algorithms implemented in the DSE (Johnston et al. 2009; Johnston et al. 2010); here we provide a high-level overview of the current set of algorithms used for scheduling. We also discuss the design of service aliases inasmuch as they underpin all of the DSE functionality.

Modeling of DSN Services

One of the challenges of modeling the DSN scheduling domain is the wide range of options available for making

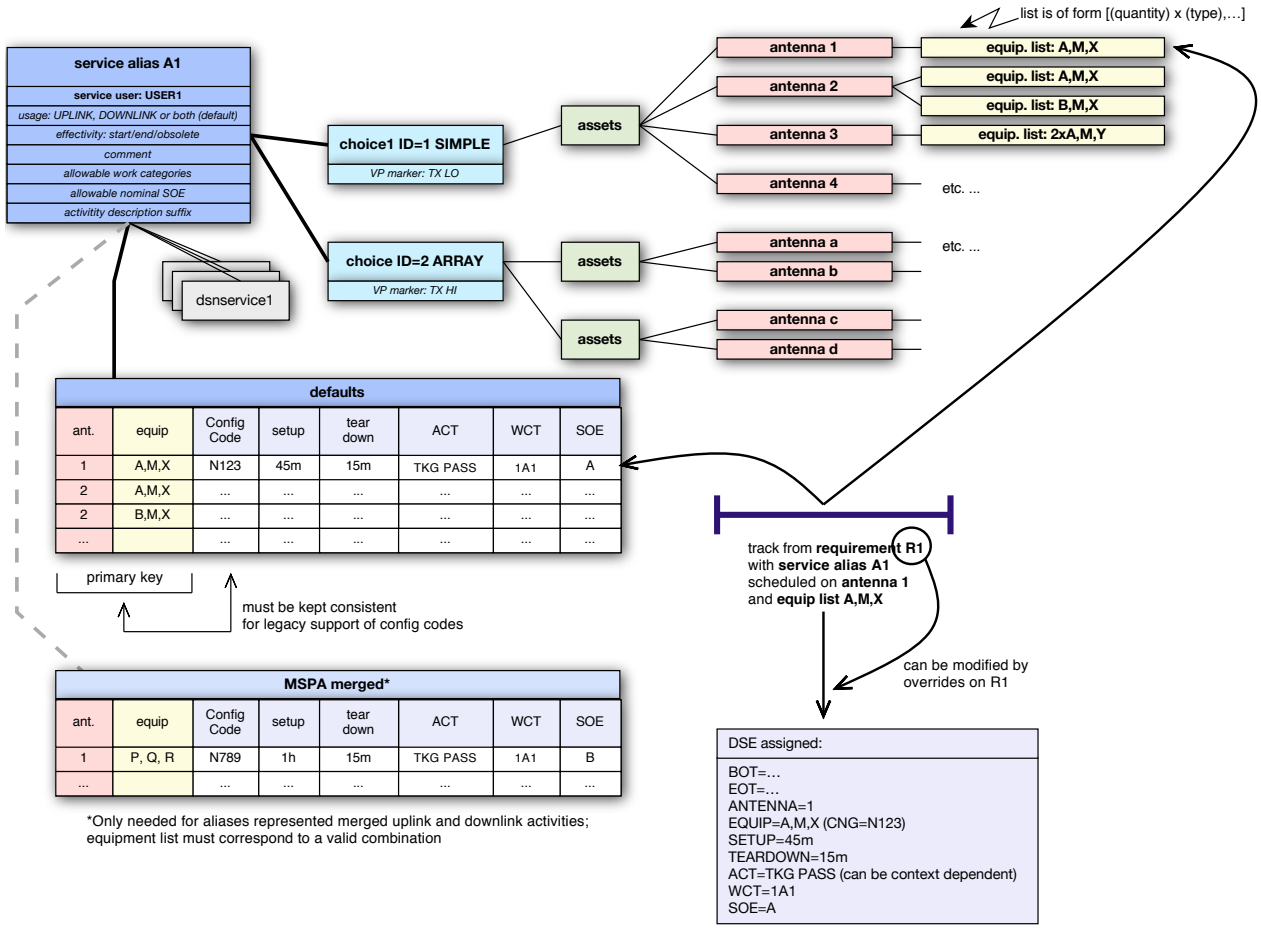


Figure 4. Illustration of the DSE service alias design, showing the information associated with an example service alias A1. Items shown in bold are required, while optional items are in italics. The choice tree structure describes the antenna and equipment options that exist, while the defaults table provides the addition data need to fill in a track's attributes.

use of the network. As previously described, one of the primary attributes of a scheduling request is the specification of the DSN services that are needed, which must be transformed into a set of specific resource reservations to satisfy the request. It has been a key element of the DSE design that users can specify their needs at a more general and abstract level, and that the system will translate into the details, ensuring the right antennas and equipment are scheduled. This has the obvious advantage that there is flexibility in the implementation of a request that can be used by the DSN systems, e.g. to optimize the schedule or to re-schedule on short notice in case assets go down. At the same time, the scheduling system needs to handle a very detailed specification of requested tracking time, down to the selection of individual antennas and equipment types to be reserved. A design to accommodate this spectrum of possibilities has been developed and implemented in the DSE, and is illustrated in Figure 4.

Each DSN service user or mission must define one or more service configurations, which are referred to by a name or "alias". Each configuration specifies the following information:

- one or more choices for how antennas and equipment can be allocated to meet the user's DSN requirements
- for each choice, which sets of antenna and equipment are acceptable
- for each antenna/equipment combination, what are the default values for associated tracking parameters, such as:
  - setup and teardown time before and after the track
  - the 16 character activity description for the track
  - a standardized work category used to identify the kind of activity
  - if applicable, a specific sequence of events that define all steps that occur during the track

A "choice" within an alias represents a high-level configuration option. For example, some missions may require either a single 70m antenna, or two or more arrayed 34m antennas. Each of these possibilities corresponds to very different antenna selections, while still satisfying the requirements of the overall service specification. Within a choice, all acceptable sets of antennas/equipment combinations must be specified, in preference order (if applicable).

Antenna/equipment combinations within a single antenna choice are in the form of a single list, while those in array choices contain multiple such lists. The same antenna may play different roles within these options, for example as a reference or slave antenna depending on how the equipment is to be configured.

Depending on the nature of the activity, different times must be scheduled for the activity setup (before tracking starts) and teardown (after it completes). Typical setup times are 30 to 90 minutes, while teardown times are usually shorter. The alias definition specifies the default (minimum) setup and teardown time for each antenna/ equipment option. In special circumstances these times may be lengthened, but may not be shortened without violating DSN operational rules (and causing a setup or teardown conflict).

Once aliases are defined and validated, their usage in DSE is straightforward. Whenever a user creates a scheduling requirement, a service alias must be specified. The selected alias then determines all the remaining DSN asset requirements and options, while the remainder of the requirement goes on to specify parameters such as timing, duration, and relationships to other tracks. By separating the definition of aliases from their usage, it becomes easier to validate them to ensure that any selection is a legal DSN configuration for that service user.

Most DSN service users will define at least several aliases corresponding to their commonly used scheduling configurations. For example, one alias might specify downlink-only configurations, while another might be used for both downlink and uplink: the latter requires the allocation of transmitters as well as receivers and decoders.

In addition to specifying which service alias applies to a given requirement, the DSE provides a capability for *overriding* the definition of that alias in any requirement in which it is used. An alias override can only *restrict* the full set of choices allowed by the alias, not add additional ones. As a result, validating the alias is sufficient to ensure that only legal configurations can be generated by the scheduling system. Examples of possible alias overrides include the following limits:

- to a single antenna vs. an arrayed configuration
- to one or more DSN complexes (Goldstone, Canberra, or Madrid)
- to a specific antenna subnet (70m, 34m, ...)
- to a single specific antenna and equipment combination

In addition to filtering the set of antenna and equipment choices, users can also override the default values associated with any choice. For example, a particular requirement might need an extended setup time, or customized activity description string that differs from the default. These can be specified using alias overrides.

In addition to antenna and equipment options, certain other attributes of any corresponding activities are also specified by the alias. These include:

- which kind of viewperiod must be used for scheduling, i.e. geometrical rise and set vs. higher elevation transmitter limits
- whether the activity is downlink or uplink only, which is used when scheduling MSPA activities as described in the next section
- special activity description suffixes that must be included to indicate certain types of activities
- an effective date and time range, which allows for phasing alias definitions in or out of service as ground or spacecraft configurations change

Service alias definitions are currently captured in XML files that specify all properties of the alias. They are reported in HTML format for users to use and review. A key design feature of the service alias concept in the DSE is that the same XML files are used by the DSE as the domain-specific model of DSN activities and assets, and in the S^3 GUI as the set of all legally selectable choices. Any changes to assets, aliases, or other mission parameters are immediately reflected in the DSE as well as the GUI, without code changes.

Multiple Spacecraft Per Aperture (MSPA) Scheduling

A general capability for the DSN is for a single antenna to communicate with several spacecraft at once (called Multiple Spacecraft Per Aperture, or MSPA); while two missions may downlink simultaneously to the antenna, only one may uplink. There are many benefits to sharing antenna time with multiple missions: it provides better utilization of the DSN resources and minimizes antenna setup time needed to support individual tracks. However, there are several drawbacks as well: with multiple missions involved, it increases the complexity of re-scheduling of tracks as all missions need to agree to a track change. Also, in the event of a realtime antenna or equipment failure, it increases the number of missions affected. At this time, only Mars missions are able to be part of MSPA groups, though other missions that occupy the same part of the sky, such as Cluster, have also been scheduled as MSPA in the past.

MSPA tracks also have several unique constraints that must be represented within the scheduling engine:

- downlink – no more than two tracks may be downlinking to the antenna simultaneously.
- uplink – no more than one track may be uplinking from the antenna at any time.
- single track per mission – only one track per mission can exist within a MSPA group.
- two simultaneous missions – only two missions can be scheduled to occur simultaneously.
- shared equipment – antenna equipment may be shared between MSPA tracks.
- setup and teardown – special rules exist for setup and teardown values are used for each MSPA track. These

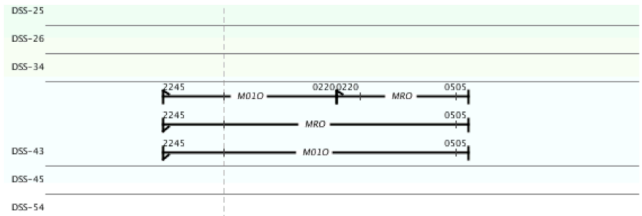
values are dependent on the temporal location of each track. The track with the earliest start time has a different setup value than any tracks that start later.

- track reconfiguration – tracks may be re-configured midway through execution to uplink/downlink or downlink only tracks. There can only be one reconfiguration per track.

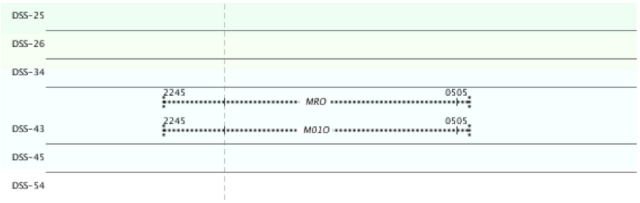
Prior to S^3, MSPA tracks were represented in the same manner as regular tracks. To indicate that a track is a member of an MSPA group, users would manually enter a unique coded string in the track's 16 character activity description field. This string contained required information such as whether the track is uplinking or downlinking, the relative priorities of missions within the group, the reconfiguration time within the track, and the reconfigured equipment state. Using the 16 character activity description field to represent MSPA track details has led to several artificial constraints in the system: a limited number of groups allowed per day, an inability to specify *multiple* track reconfigurations in one MSPA group, and a limited number of consecutive downlinks.

These limitations have led S^3 to represent MSPA tracks in a different manner. For MSPA-capable missions, the tracking time required for uplink and downlink may differ. Therefore, S^3 services for uplink-only and downlink-only tracks were introduced, specifying only the equipment used for each tracking type. These services are then referenced by the requirements, with different required tracking times specified for uplink and downlink. The engine will then schedule these tracks and attempt to combine them into single tracks where possible. Representing separate uplink and downlink tracking time allows for more flexibility in scheduling individual tracks and removes several of the artificial constraints required by use of the 16 character activity description field. However, to support existing tools and interfaces, a "legacy" representation of the tracks is still required. In this "legacy" view, the separate uplink-only and downlink-only tracks are "merged" together and the activity description fields automatically populated to represent reconfiguration parameters and times. This process is illustrated in Fig. 3.

The need to merge S^3 uplink-only and downlink-only tracks to "legacy" tracks introduced several issues that needed to be addressed. Given the unique constraints of MSPA tracks and how they are grouped together, the possibility arises that the S^3 tracks are organized in a manner such that merging them into legacy tracks is impossible. This is mitigated by the ensuring that when the scheduling engine generates tracks for MSPA-capable missions, the tracks are properly grouped together. However, with user control over track parameters, a S^3 activity can be easily added, deleted, or modified using the schedule editor. For each group of MSPA tracks, the scheduling engine will report when it is infeasible to generate "legacy" tracks. When this occurs, the scheduling engine will report a conflict and then output the S^3 tracks as the "legacy" tracks and assign a special error code in the track's activity description. The types of MSPA conflicts reported are:



(a) An example S^3 MSPA grouping on DSS-43 where the uplink-only and downlink-only tracks are represented separately. Two spacecraft (MRO and M01O) are downlinking simultaneously while the uplink occurs for M01O at the beginning of the track, then transitions to MRO until the end of the track. The equipment specified for each track represent just those that are needed to support the uplink and downlink services individually.

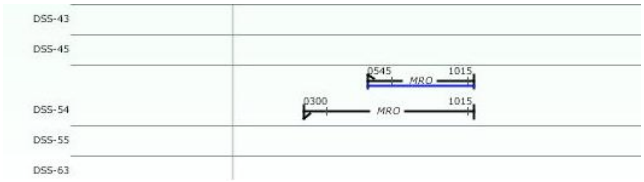


(b) The same MSPA grouping example as in Figure 1, but represented in the "legacy" track view. In this view, the uplink and downlink tracks are merged together and the activity description field contains the track reconfiguration times and supporting equipment needed.

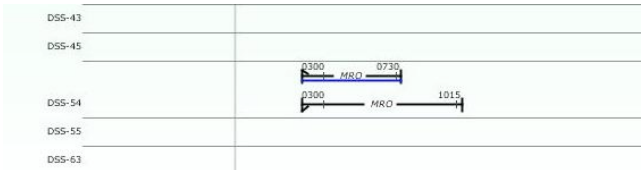
Figure 3. An example of a) S^3 scheduled MSPA activities and b) their corresponding "legacy" tracks

- Multiple Uplinks – more than one track simultaneously uplinking on the same antenna.
- Multiple Downlinks – more than two tracks simultaneously downlinking on the same antenna.
- Multiple Missions – more than two missions simultaneously tracking.
- Multiple Track Reconfigurations – more than one track reconfiguration is occurring in the merged uplink-only and downlink-only tracks. This occurs when both the start or end of the tracks differ (see Figure 4)
- Track Reconfiguration Time – The track reconfiguration time occurs during the pre-track setup time, instead of the during the tracking time.
- Downlink Coverage – An uplink-only track is not fully covered by a downlink-only track. Uplink-only tracks were introduced in S^3 and must be fully merged with downlink-only tracks in order to correctly produce "legacy" tracks.

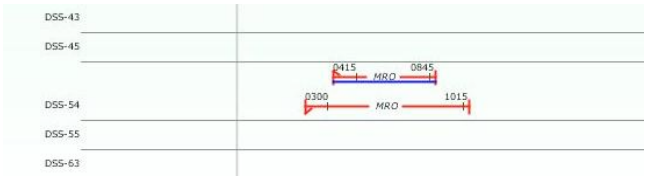
In addition, to ensure that the merged "legacy" tracks meet the service specifications of the user, embedded within the uplink and downlink requirement service aliases are a common set of legal antenna/equipment combinations for the merged tracks (Figure 4). For a "legacy" track to be considered legal, the merged configuration must be present in the service aliases for that mission. If the antenna/equipment combination is not present, it is reported as a



(a) When these two MRO tracks are merged together into a legacy track, the track will start as a downlink-only and be reconfigured midway at 0545 to uplink and downlink.



(b) When these two MRO tracks are merged together into a legacy track, the track will start as a uplink and downlink and be reconfigured midway at 0730 to downlink-only.



(c) An example of an Multiple Track Reconfiguration conflict. If the two tracks are merged together into a legacy track, it would begin as downlink-only, reconfigure to uplink and downlink at 0415, and then reconfigure again at 0845 to downlink-only. There is a limit of only one track reconfiguration for MSPA tracks

Figure 4. An example of multiple track reconfiguration conflicts.

requirement service violation, prompting the user to make the appropriate updates to the tracks. Alternatively, the user may also invoke the scheduling engine to attempt to resolve the violation.

Overview of Scheduling Strategies

There are a few basic design principles around which the DSE algorithms have been developed, derived from the basic role of the DSE as the provider of intelligent decision support to DSN schedulers. In support of schedule repair and negotiation, it is critically important that the DSE follow a "no surprises" paradigm, i.e.

- no unexpected schedule changes: all changes to the schedule must be requested, explicitly or implicitly, and the same sequence of operations on the same data must generate the same schedule
- even for infeasible scheduling requests, attempt to return something "reasonable" in response, possibly by relaxing aspects of the request; along with a diagnosis of the sources of infeasibility, this provides a starting point for users to handle the problem

In contrast to this mode of operation is an auto-generation phase of the scheduling process where the goal is to integrate scheduling requests from all users. The result is an initial schedule with minimal conflicts and violations to serve as a starting point for collaborative conflict resolution. In this mode, maintaining schedule stability is not an objective, and a much broader range of changes to the scheduled activities is allowable, provided that overall conflicts are reduced. The DSE supports both modes of operation with a portfolio of algorithms that can be invoked by the S^3 system for auto-generation, or by end users when working on specific conflicted portions of the schedule.

Expanding Requests to Tracks

Initial Layout: The initial layout algorithm is the primary algorithm users invoke to generate tracks to satisfy the specifications of the request. It is also used to remove any existing tracks and regenerate them around whatever other activities already exist in the schedule. The algorithm consists of a series of systematic search stages over the legal track intervals, successively relaxing request constraints each stage if no solution is found. The systematic search algorithm is a depth-first search algorithm over the space of available antenna/equipment start times and durations for each scheduling request. The set of legal antenna/equipment for scheduling is defined in the request service alias specification, while the search space of legal start times is defined by the request quantization value (usually 5 minutes).

The successive relaxation of constraints allow for tracks to be generated even though the scheduling request may be infeasible (in isolation or within the context of the current schedule), and provides the user a starting point to make corrective changes. These changes may range from modifying the scheduling request to introduce more tracking flexibility, to contacting other mission schedulers to negotiate different request time opportunities. One of the limitations of the initial layout algorithm is its ability to schedule collections of requests associated with track relationships. As it iterates over these requests, tracks may be generated without regard to the feasibility of generating tracks for the future requests in the collection. As a result, it is prone to creating violations for users whose requests are highly interconnected.

STN Scheduling: To address the limitations of the initial layout algorithm with interconnected requests, early work has begun using a Simple Temporal Network (STN) to generate tracks for a targeted set of DSN users. The algorithm can be described in two parts: pruning of the legal intervals for each request based on the STN, followed by a systematic search of the pruned legal intervals for a solution.

In pruning the legal intervals, a STN is first initialized with track time constraints on the request boundaries and the track relationships. After propagation, the STN is then used to make a first pass at pruning the legal intervals based on the earliest legal start time and the latest legal end time for each request. A second attempt at pruning the the legal intervals is performed by adding additional track time constraints to include the earliest start time and latest end time of a request legal interval. We then systematically begin searching for a solution by temporally assigning a track to the each legal interval and including it into the

STN. If the STN is inconsistent, we re-assign a track into the next legal interval for that request. Once a consistent STN is found, a valid schedule is generated.

Additional work is still required for the STN scheduling algorithm. At present, it is only used for scheduling tracks for a small subset of the DSN users where the requests are tightly connected. It will also need to be extended to support relaxing specific request constraints to generate some tracks. With the current implementation, if a valid solution cannot be found, no tracks are generated. This is undesirable as it provides no feedback to the user to determine what the problem may be in the requests or schedule.

Repairing Conflicts and Violations in the Schedule

Basic Repair: Once an initial schedule has been generated, conflicts and/or violations may exist in the schedule due to the relaxation of constraints. The DSE provides a basic repair algorithm to reduce conflicts or violations. The algorithm will identify the contributing tracks for each conflict or violation, and run the systematic search algorithm on the request. If a solution is found, the new tracks are accepted. If no solution is found, the original tracks are not modified. Note that conflicts and violations are independent, so there are separate versions provided through the user interface for users to invoke. This algorithm is focused on only modifying requirements that are directly contributing to the conflict or violation in order to minimize the impact on the other parts of the schedule. However, in order to resolve certain classes of conflicts, multiple tracks not directly associated with the conflict may need to be modified. A strategy that addresses these types of conflicts is discussed next.

Stochastic Relayout: This algorithm generates a new schedule based on the existing tracks in the schedule. The algorithm loops through each track in the schedule and stochastically updates any or all of the parameters including start time, duration, antenna, etc. Each new schedule generated attempts to reduce the number of track conflicts and request violations addressing the issue with basic repair as it is able to find solutions that require modifying multiple tracks that are not directly related to the conflict/violation. Compared to initial layout and basic repair, this strategy was able to reduce the number of conflicts and violations in several test schedules by over 40%.

5 Conclusions

We have described the DSN Scheduling Engine (DSE) component of the Service Scheduling Software (S^3) system, a new scheduling system for NASA's Deep Space Network. The DSE implements a request-driven approach to scheduling, incorporating a sophisticated request specification language, algorithms for generating tracks, resolving conflicts, and repairing request violations, and a distributed architecture to provide high-availability service to a large number of simultaneous users. For over a year, the DSE provided the first step of the DSN scheduling process by integrating requirements from all users into a "preview" schedule. Currently the S^3 system is in the process of deployment to operations.

Future work includes a number of areas of further research and development that are under consideration:

- forecasting – the DSE scheduling model is based on the explicit expansion of scheduling requests to tracks, taking into account fine-grained constraints and preferences as they affect the resulting tracks (e.g. viewperiods, constraining event intervals, etc.) This is ideal for near- to mid-term scheduling, but also has application to long-range resource planning as well, in that detailed contention scenarios can be explored and assessed. The major additional capability that would be useful in this long-range planning context is a more integrated capability to model uncertain events.
- multi-objective scheduling – like many scheduling problem domains, DSN scheduling is full of tradeoffs among competing objectives, ranging from individual mission users, to system-level utilization and robustness objectives. Multi-objective optimization has been demonstrated in other domains (Johnston 2008; Johnston and Giuliano 2010) to provide powerful insights into optimal tradeoffs. There is every reason to believe this would be useful for DSN schedulers as well.
- cross-network scheduling – NASA has recommended (SCAWG 2006) integrating access to the capabilities provided by its three major networks: DSN, the Space Network (SN), and the Near Earth Network (NEN). For those users who require services from two or all three of these networks, such integration would be a source of significantly improved efficiency and cost savings. S^3 has the potential to serve as a common scheduling platform in this regard – it is interesting to note that nowhere on the S^3 scheduling request editor main UI is there any indication that the user is working with the DSN; this is apparent only when drilling down into the detailed visibility and event intervals, and service definitions.

Acknowledgments

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The authors gratefully acknowledge the support and involvement of the DSN scheduling user community over the course of this work.

References

Bell, C. (1992). Scheduling Deep Space Network Data Transmissions: A Lagrangian Relaxation Approach, Jet Propulsion Laboratory.

Biefeld, E. and L. Cooper (1991). *Bottleneck identification using process chronologies*. Proceedings IJCAI, Sydney, Australia.

Carruth, J., M. D. Johnston, A. Coffman, M. Wallace, B. Arroyo and S. Malhotra (2010). A Collaborative Scheduling Environment for NASA's Deep Space Network. *SpaceOps 2010*. Huntsville, AL, AIAA.

Chien, S., R. Lam and Q. Vu (1997). *Resource Scheduling for a Network of Communications Antennas*. IEEE Aerospace Conference, Aspen, CO.

Chien, S., G. Rabideau, R. Knight, R. Sherwood, B. Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins and D. Tran (2000). *ASPEN - Automating Space Mission Operations using Automated Planning and Scheduling*. SpaceOps 2000, Toulouse, France.

Clement, B. J. and M. D. Johnston (2005). *The Deep Space Network Scheduling Problem*. Innovative Applications of Artificial Intelligence (IAAI), Pittsburgh, PA, AAAI Press.

Fisher, F., S. Chien, L. Paal, E. Law, N. Golshan and M. Stockett (1998). *An automated deep space communications station*. Proceedings IEEE Aerospace Conference, Snowmass at Aspen, CO

Guillaume, A., S. Lee, Y. Wang, H. Zheng, R. Hovden, S. Chau, Y. Tung and R. Terrile (2007). *Deep Space Network scheduling using evolutionary computational methods*. 2007 IEEE Aerospace Conference.

Imbriale, W. A. (2003). *Large Antennas of the Deep Space Network*, Wiley.

Johnston, M. D. (2008). "An Evolutionary Algorithm Approach to Multi-Objective Scheduling of Space Network Communications." *International Journal of Intelligent Automation and Soft Computing* **14**: 367-376.

Johnston, M. D. and B. J. Clement (2005). *Automating Deep Space Network Scheduling and Conflict Resolution*. ISAIRAS-05, Munich, Germany.

Johnston, M. D. and M. Giuliano (2010). *Multi-User Multi-Objective Scheduling for Space Science Missions*. SpaceOps 2010, Huntsville, AL.

Johnston, M. D., D. Tran, B. Arroyo, J. Call and M. Mercado (2010). Request-Driven Schedule Automation for the Deep Space Network. *SpaceOps 2010*. Huntsville, AL, AIAA.

Johnston, M. D., D. Tran, B. Arroyo and C. Page (2009). *Request-Driven Scheduling for NASA's Deep Space Network*. International Workshop on Planning and Scheduling for Space (IWPSS), Pasadena, CA.

Kan, E. J., J. Rosas and Q. Vu (1996). Operations Mission Planner - 26M User Guide Modified 1.0, Jet Propulsion Laboratory.

Loyola, S. J. (1993). PC4CAST: A Tool for DSN Load Forecasting and Capacity Planning, Jet Propulsion Laboratory**:** 170-184.

SCAWG (2006). NASA Space Communications and Navigation Architecture Recommendations, Final Report, Space Communications Architecture Working Group (SCAWG).

Werntz, D., S. Loyola and S. Zendejas (1993). *FASTER - A tool for DSN forecasting and scheduling*. Proceedings of the 9th AIAA Computing in Aerospace Conference, San Diego, CA.