Automated Scheduling for NASA's Deep Space Network

Mark D. Johnston, Daniel Tran, Belinda Arroyo, Sugi Sorensen, Peter Tay, Butch Carruth, Adam Coffman, Mike Wallace

■ *This article describes the Deep Space Network (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 that provides DSE functionality to all DSN users through a standard web browser, as part of a peer-to-peer schedule negotiation process for the entire network. The system has been deployed operationally and is in routine use, and is in the process of being extended to support long-range planning and forecasting and near real-time scheduling.*

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 70 meter antenna and several 34 meter antennas (figure 1), 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 its large antennas can be found in the paper by W. A. 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

Figure 1. Three of the Deep Space Network 34 Meter Antennas at the Goldstone Deep Space Communications Complex in California.

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 (Bell 1993; Biefeld and Cooper 1991; Chien et al. 1997; Fisher et al. 1998; Guillaume et al. 2007; Kan, Rosas, and Vu 1996; Loyola 1993; Werntz, Loyola, and Zendejas 1993). Currently, the DSN scheduling process is centered on the service preparation subsystem (SPS), which provides a central database for the real-time schedules and for the auxiliary data needed by the DSN to operate the antennas and communications equipment (for example, view periods, sequence-of-events files). The current project to improve scheduling automation is designated the service scheduling software, 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 previous activity-oriented approach that specified individual activities); (2) unifying the scheduling software and databases into a single integrated suite covering real time out through as much as several years into the future; and (3) development of a peer-to-peer collaboration environment for DSN users to view, edit, and negotiate schedule changes and conflict resolutions.

The collaboration environment is described elsewhere (Carruth et al. 2010); this article focuses on the first and second areas and some of their ramifications. (For additional information see Clement and Johnston [2005]; Johnston and Clement [2005]; Johnston et al. [2009]; Johnston et al. [2010].)

The request-driven paradigm shifts the emphasis

Typical number of tracking passes per week	425
Number of users (missions, science users, and maintenance)	37
Typical pass duration	5.25 hours
Assets	12 antennas at 3 sites (to be augmented to 16 by 2020)
Asset loading	~80–95 percent
Scheduling time scale	Preview schedule 17–26 weeks ahead Conflict free 8 weeks ahead

Table 1. Some Characteristics of the DSN Scheduling Problem.

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 and on the reasons for the request-driven approach taken by S^3. We then briefly describe the scheduling request specification itself, which is how DSN users of S^3 convey their service requests to the system. These requests are processed by the DSN scheduling engine (DSE) to expand into tracking passes and integrate them into an overall schedule, all the while seeking to minimize conflicts and request violations. We conclude with an overall summary and brief description of plans for future development.

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. Currently the DSN supports 37 spacecraft or service users, counting all those with regular requirements for scheduled time on any antenna. The mission users span a wide range of distance and orbit type: high Earth orbit, lunar orbit, solar orbit, probes at Mercury, Venus, Mars, and Saturn (and en route to Jupiter and Pluto/Charon), and to comets and asteroids, out to the two *Voyager* spacecraft in interstellar space. Ground-based users conduct radio science and radio astronomy using the antennas, including coordinated programs with international partners. Other activities that must be scheduled include routine and special maintenance, calibration, engineering, and test activities. The collected set of DSN users imposes a very wide range of usage requirements on the network due to differing designs and operating modes. Some users require occasional contacts of only a few hours per week, but this ranges up to continuous coverage during certain mission phases, such as post-launch and during critical mission events. At the present time, a typical week includes between 400 and 500 scheduled activities on the antennas of the three DSN complexes; a portion of such a schedule is shown in figure 2 in the S^3 web GUI.

Phases of the DSN Scheduling Process

The DSN scheduling process consists of three phases, which do not have sharply defined boundaries. Below we describe these phases as they exist today; later in this article we discuss plans for how they may change in the future.

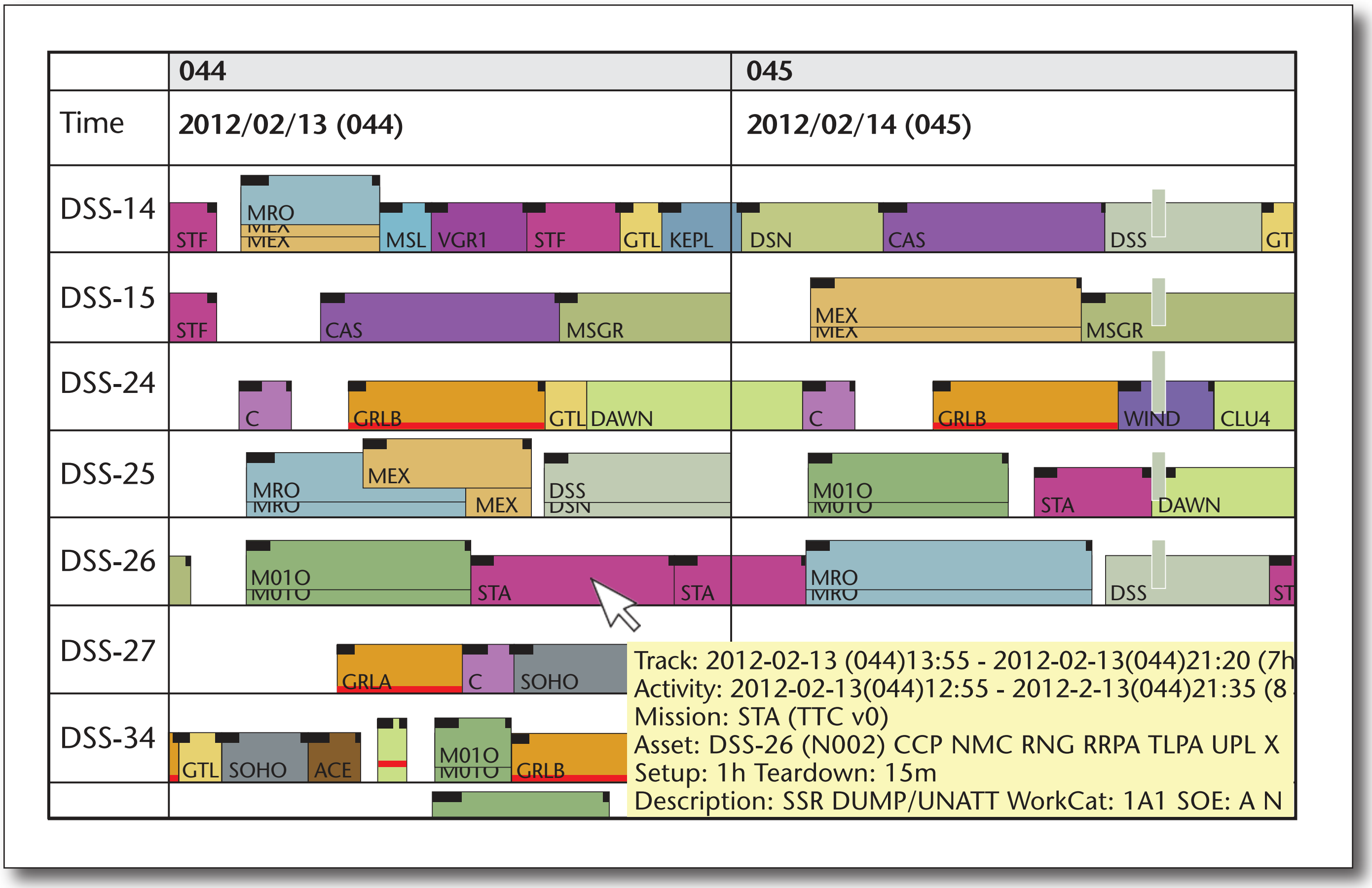
Long-Range Planning and Forecasting

In today's system, long-range planning is based on user-provided high-level requirements, specified in the form of a spreadsheet that is interpreted by analysts and entered into a database at JPL. The forecast software employs a statistical allocation method (Lacey and Morris 2002) to estimate when these requirements translate into DSN loading over various time frames. Long-range planning has several major purposes: studies and analyses, down time analysis, and future mission analysis.

For planning studies and analyses, periods of particular interest or concern are examined to determine where there is likely contention among missions, for example around launches or critical mission events (maneuvers, planetary orbit insertion or landings), or when construction of a new DSN antenna is under investigation. Down time analysis involves identifying periods of time when necessary antenna or other maintenance can be scheduled, attempting to minimize the impact on missions. For future mission analysis, missions can, in proposal phase, request analysis of their proposed DSN coverage as part of assessing and costing proposals for new missions. The time range for long-range planning is generally six months or more into the future, sometimes as much as years.

Midrange Scheduling

The midrange scheduling phase is when detailed user requirements are specified, integrated, negotiated, and all tracking activities finalized in the schedule. Starting at roughly 4–5 months before execution, users specify their detailed scheduling requirements



Track: 2012-02-13 (044)13:55 - 2012-02-13(044)21:20 (7h
Activity: 2012-02-13(044)12:55 - 2012-2-13(044)21:35 (8
Mission: STA (TTC v0)
Asset: DSS-26 (N002) CCP NMC RNG RRPA TLPA UPL X
Setup: 1h Teardown: 15m
Description: SSR DUMP/UNATT WorkCat: 1A1 SOE: A N

Figure 2. Example of an HTML5 Canvas View of a Portion of the DSN Schedule.

Mousing over a track brings up a transient window with detailed information about the activity (lower right). In this view, different missions are color coded, and setup/teardown is indicated by the black segments at the top left and right of each activity. Each time line represents one of the DSN antennas.

on a rolling weekly basis. These requirements include tracking time and services required, constraining time intervals and relationships (for example, minimum and maximum gaps), visibility constraints, and flexibilities. Further discussion of the nature of these requirements and flexibilities is included in the DSN Scheduling Requests section.

Once the deadline passes and all requirements are in, the full set is integrated into an initial schedule in which conflicts are reduced by taking advantage of whatever flexibilities have been specified. This version of the schedule is extremely heavily overloaded, but it does indicate where there are contentious time periods. These contentious areas shift from week to week depending on critical activities, as well as on the slow drift of visibility intervals with time.

There follows an optimization step where an experienced DSN scheduler interactively edits the schedule and further reduces conflicts by taking advantage of unspecified flexibilities and making further adjustments. At the conclusion of this phase, the schedule usually contains fewer than 30 conflicting sets of activities. It is then released to the scheduling user community who negotiate to reduce conflicts and further optimize coverage for their missions.

It is important to note that, unlike many other scheduling domains, the DSN follows a collaborative approach to developing the conflict-free schedule. DSN users follow a peer-to-peer approach to resolving conflicts. Users create change proposals, which are suggestions as to how different 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; escalation very rarely occurs in practice. When negotiation concludes, the schedule is conflict free or has only a few waived conflicts for specific reasons. This is considered the negotiated schedule that missions use to plan their integrated ground and spacecraft activities, including the development of on-board command loads based in part on the DSN schedule.

Following this point, changes to the schedule may still occur, but new conflicts may not be introduced (by policy). There is a continuing low level of no-impact changes and negotiated changes that occur all the way down to real time.

Near Real-Time Scheduling

The near real-time phase of DSN scheduling starts roughly 2–3 weeks from execution and includes the period through execution of all the scheduled activities. Late changes may occur for various reasons (sometimes affecting the midrange phase as well). For example, users may have additional information or late changes to requirements for a variety of reasons; DSN assets (antennas, equipment) may experience unexpected down times that require adjustments to the schedule to accommodate; or spacecraft emergencies may occur that require extra tracking or changes to existing scheduled activities. For many missions that are sequenced well in advance, late changes cannot be readily accommodated.

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. The main elements of a scheduling request are service specification, timing constraints, track relationships, priority, preferences, repetitions, and nonlocal time line constraints.

Service Specification

S^3, through 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 on-board 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 antenna, or MSPA); while more than one at a time may be sending data to Earth, only one at a time may be receiving data sent from Earth.

A more detailed description of service alias functionality is provided in the description of the DSN scheduling engine that follows.

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); quantization (whether scheduled activity times are to occur on 1-minute or 5-minute boundaries); view periods (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); and events, which are general time intervals that constrain when tracks may be allocated. Event examples include day of week, time of day (for accommodating shift schedules, daylight, and others); (orbit/trajectory events (occultations, maneuvers, surface object direct view to Earth). Different event intervals may be combined (with optional inversion), and applied to a request.

Track Relationships

In some cases, contacts need to be sufficiently separated so that on-board data collection has time to accumulate data but not overfill on-board 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 safe mode. 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 that 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 S^3 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 view period or event timing interval.

Repetitions

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. S^3 provides for repeated requests, analogous to repeated or recurrent meetings in calendaring systems, in order to minimize the repetitive entry of detailed request information.

Nonlocal Time Line Constraints

Some users have constraints that affect allocations in a nonlocal manner, meaning that an extended time period and possibly multiple activities may have to be examined to tell whether some preferred condition is satisfied. Examples of these constraints include *n* of *m* tracks per week should be scheduled on southern hemisphere tracking stations; *x* hours of tracking and ranging per day must be scheduled from midnight to midnight UTC; the number and timing of tracks in a week should not allow the on-board recorder to exceed its expected capacity.

The DSN Scheduling Engine

The DSE is the component of S^3 responsible for 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; and 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 (for example, 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 3, showing the overall dependencies of the interface message types on the various software modules.

Architecture

The DSE is based on ASPEN, the planning and scheduling framework developed at Jet Propulsion Laboratory and previously applied to numerous problem domains (Chien et al. [2000]; see also Chien et al. [2012] for a comparison with various time line–based planning and scheduling systems). 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 (figure 4) with multiple running instances of ASPEN, each available to serve a single user at a time. We use a middleware tier to link the ASPEN instances to their clients, on-board 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. In addition, 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 AMAs 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 also 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. Finally, the SMA process can be restarted, for example, 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-

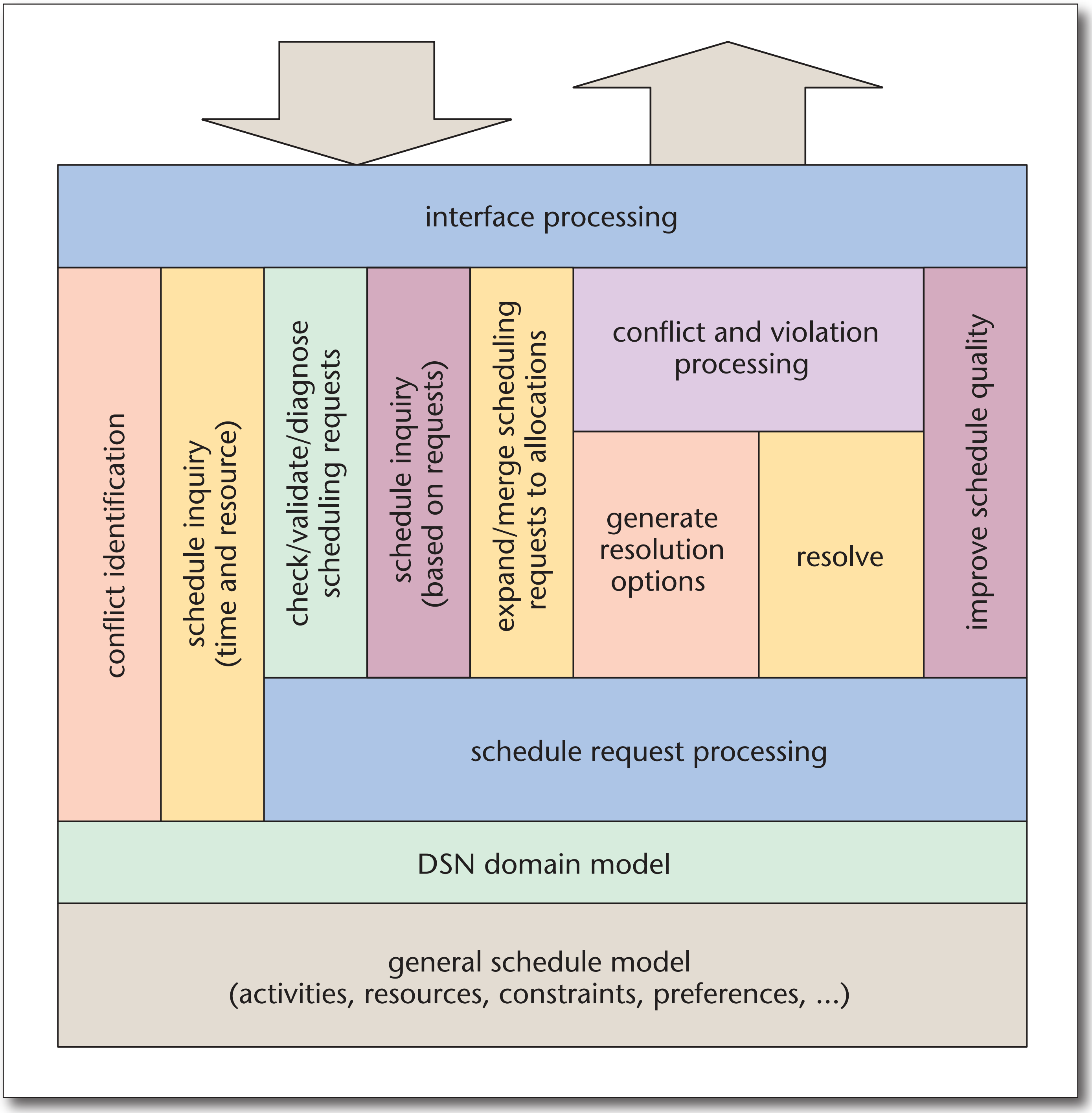

interface processing
conflict identification
schedule inquiry (time and resource)
check/validate/diagnose scheduling requests
schedule inquiry (based on requests)
expand/merge scheduling requests to allocations
conflict and violation processing
generate resolution options
resolve
improve schedule quality
schedule request processing
DSN domain model
general schedule model (activities, resources, constraints, preferences, ...)

Figure 3. A Block Diagram of the DSE Architecture.

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.

Next we discuss some of the challenges related to modeling (DSN services, multiple simultaneous spacecrafts, nonlocal time line constraints) and schedule generation and repair algorithms. We also discuss the design of service aliases inasmuch as they underpin all of the DSE functionality.

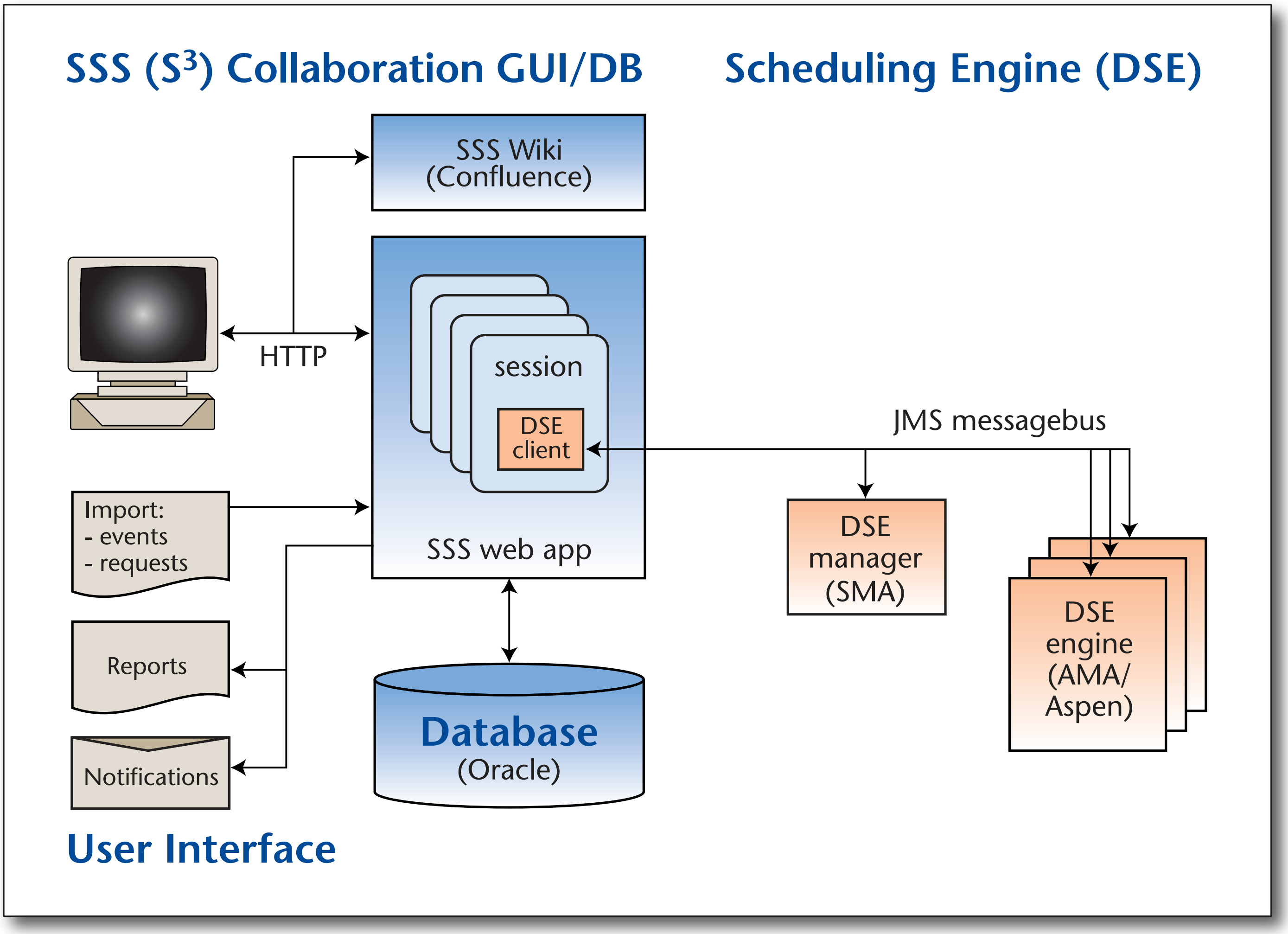


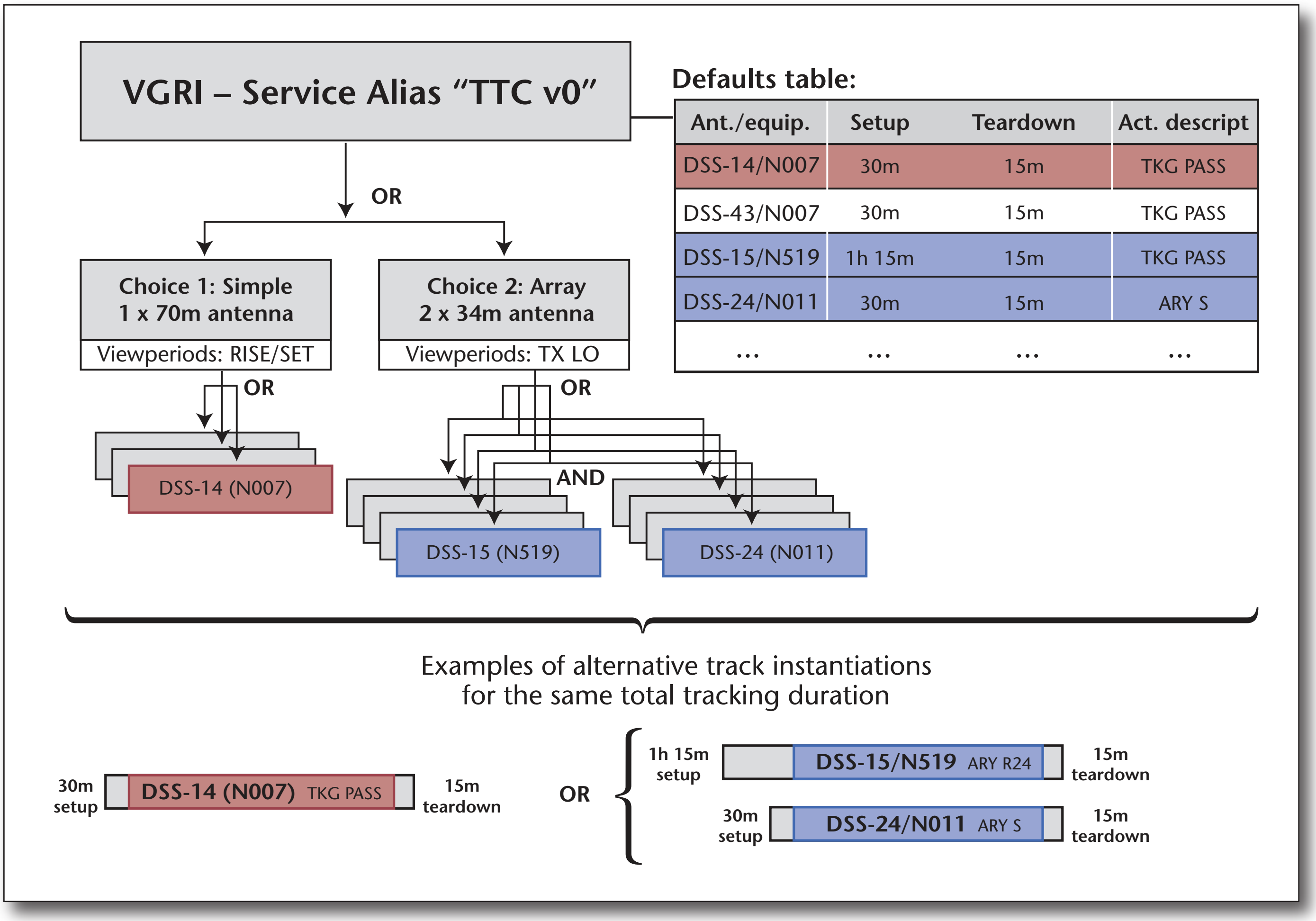
Figure 4. An Overview of the S^3 System Architecture.

The DSN scheduling engine manages and provides a set of servers that respond to users' requests for scheduling services through the S^3 web application.

Modeling of DSN Services

One of the challenges of modeling the DSN scheduling domain is the wide range of options available for making 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, for example, to optimize the schedule or to reschedule 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 5.

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: (1) one or more choices for how antennas and equipment can be allocated to meet the user's DSN requirements; (2) for each choice, which sets of antenna and equipment are acceptable; and (3) 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, and if applicable, a specific sequence of events that define all steps that occur during the track.



Defaults table:

Ant./equip.	Setup	Teardown	Act. descript
DSS-14/N007	30m	15m	TKG PASS
DSS-43/N007	30m	15m	TKG PASS
DSS-15/N519	1h 15m	15m	TKG PASS
DSS-24/N011	30m	15m	ARY S
…	…	…	…

Figure 5. An Illustration of the Structure of a Service Alias Representing a Choice Between a Single Antenna and Multiple Antenna (Array) Implementation of the Same Tracking Duration.

Red highlights the information related to a single-track choice (left) and blue that related to a two-antenna array choice (right). More complex aliases are used to represent up to four station arrays, specialized ranging tracks (DDOR), separate uplink and downlink options for multiple spacecraft tracked all at once, and maintenance activities that affect an entire complex or the entire DSN at once.

A choice within an alias represents a high-level configuration option. For example, some missions may require either a single 70-meter antenna, or two or more arrayed 34-meter 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.

The example illustrated in figure 5 shows how the definition of a service alias for the spacecraft *Voyager 1* encapsulates the alternative options of scheduling on a single 70-meter antenna, or alternatively on a pair of compatible 34-meter antennas. The scheduling user need only specify the service alias name TTC v0, a widely used acronym for telemetry, tracking, and commanding, and the scheduling engine will ensure that any associated activities are compatible with the service alias. Service aliases are versioned over time and can be phased in and out of usage as spacecraft or ground system capabilities change.

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 original alias is sufficient to ensure that only legal configurations can be generated by the scheduling system. Examples of possible alias overrides include limits to a single antenna versus an arrayed configuration; limits to one or more DSN complexes (Goldstone, Canberra, or Madrid); limits to a specific antenna subnet (70 meter, 34 meter, and others); and limits 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 view period must be used for scheduling, that is, geometrical rise and set versus higher elevation transmitter limits; whether the activity is downlink or uplink only, which is used when scheduling MSPA activities (described in the next section); special activity description suffixes that must be included to indicate certain types of activities; and an effective date and time range.

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 Antenna Scheduling

A general capability for the DSN is for a single antenna to communicate with several spacecraft at once (called multiple spacecraft per antenna, 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 rescheduling of tracks as all missions need to agree to a track change. Also, in the event of a real-time 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. No more than two tracks may be downlinking to the antenna simultaneously. No more than one track may be uplinking from the antenna at any time. Only one track per mission can exist within an MSPA group (single track per mission). Only two missions can be scheduled to occur simultaneously. Antenna equipment may be shared between MSPA tracks. 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. Tracks may be reconfigured 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 mis-

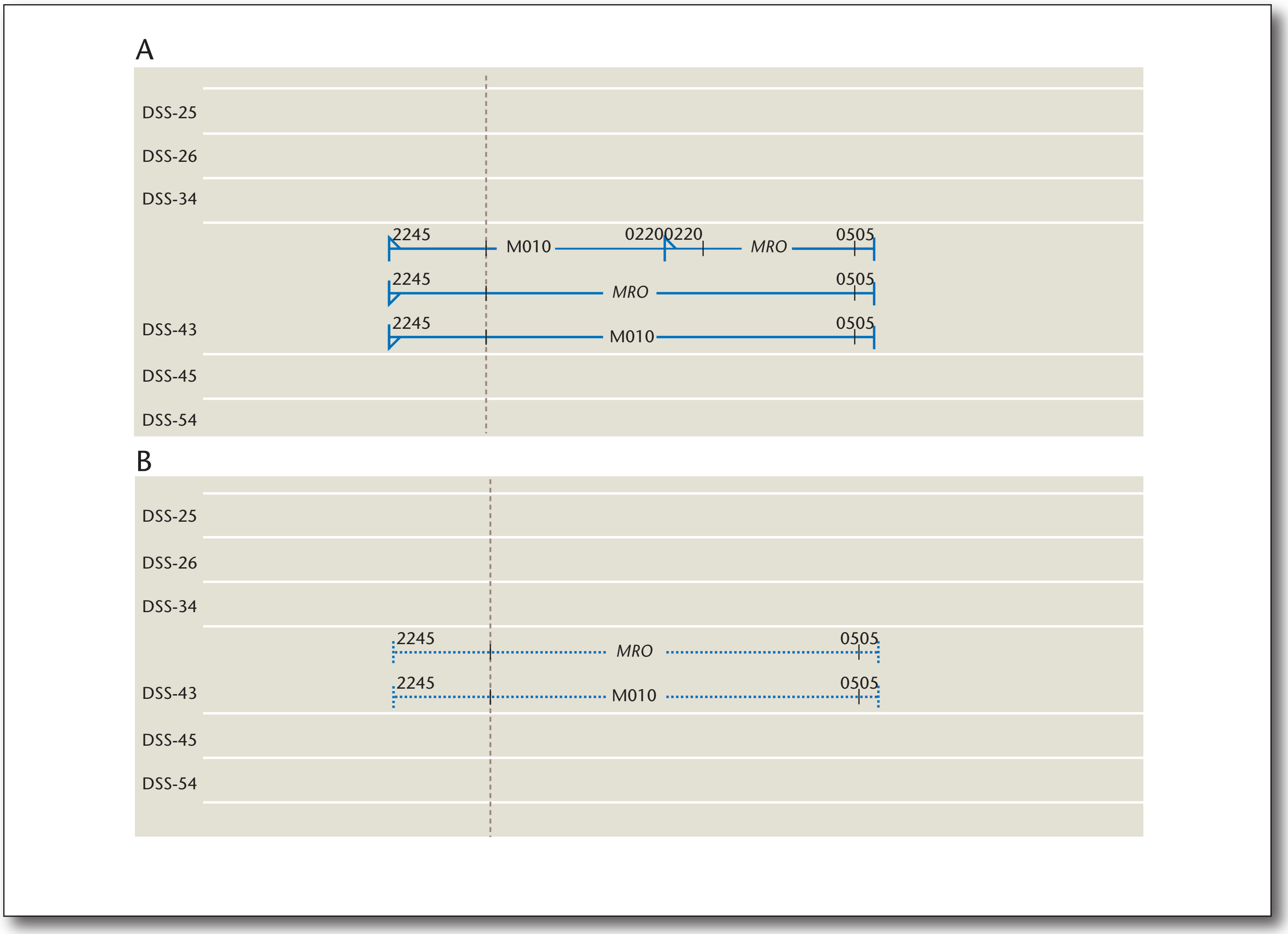


Figure 6. An Example of S³ Scheduled MSPA Activities and Their Corresponding Legacy Tracks.

(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 represents just the equipment that is 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.

sions, 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 figure 6.

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 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, an S^3 activity can be easily added, deleted, or modified using the schedule

Request Type	Examples	Parameters
Total time	8 hours of tracking per day 6 hours of uplink tracking each midnight to midnight UTC 24 hours of specific activity types per week summed over four different but related spacecraft	mission(s) service aliases time frame (1 day, 1 week, etc.) min/max tracking times with yellow/red limit
Tracking gaps	6–12 hour gap between tracks, measured midpoint to midpoint Gaps no greater than 8 hours measured EOT to BOT	mission service aliases min track gap max track gap yellow limits measured by (BOT-BOT, EOT-EOT, midtrack to midtrack)
DSN complex distribution	3 of 10 tracks per week must be scheduled at Canberra DSN complex At least one track per week must be scheduled at each DSN complex	mission duration list of (complex, count)
Recorder	Do not exceed on-board recorder volume capacity limit	mission track overhead duration recorder collection rate (X units/s) yellow/red recorder max capacity recorder downlink rates (antenna, downlink rate X units/s) initialization rule

Table 2. Time Line Requirement Types, with Examples and Parameters.

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 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 7); track reconfiguration time (the track reconfiguration time occurs during the pretrack setup time, instead of the during the tracking time); and 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. 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 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.

Time Line Constraints and Preferences

The initial development of S^3 has focused on the most frequently encountered types of scheduling request types, which directly affect how DSN antenna allocations are to be constructed. A second broad category of scheduling requirements includes those that indirectly affect allocations in a nonlocal manner. There can be a tradeoff between satisfying these types of requirements, versus the direct requirements noted previously.

We have denoted these types of scheduling requests as time line constraints or preferences, since they are best assessed by considering the overall time line of activities (or subset of activities) for a DSN service user over some time period. Table 2 includes a more detailed list of major time line requirement types and their parameters.

Because these requests have a varying degree of preference, and therefore need to be accessible to the judgement of the scheduling users, we have pursued their incorporation into S^3 in two phases; (1) as inte-

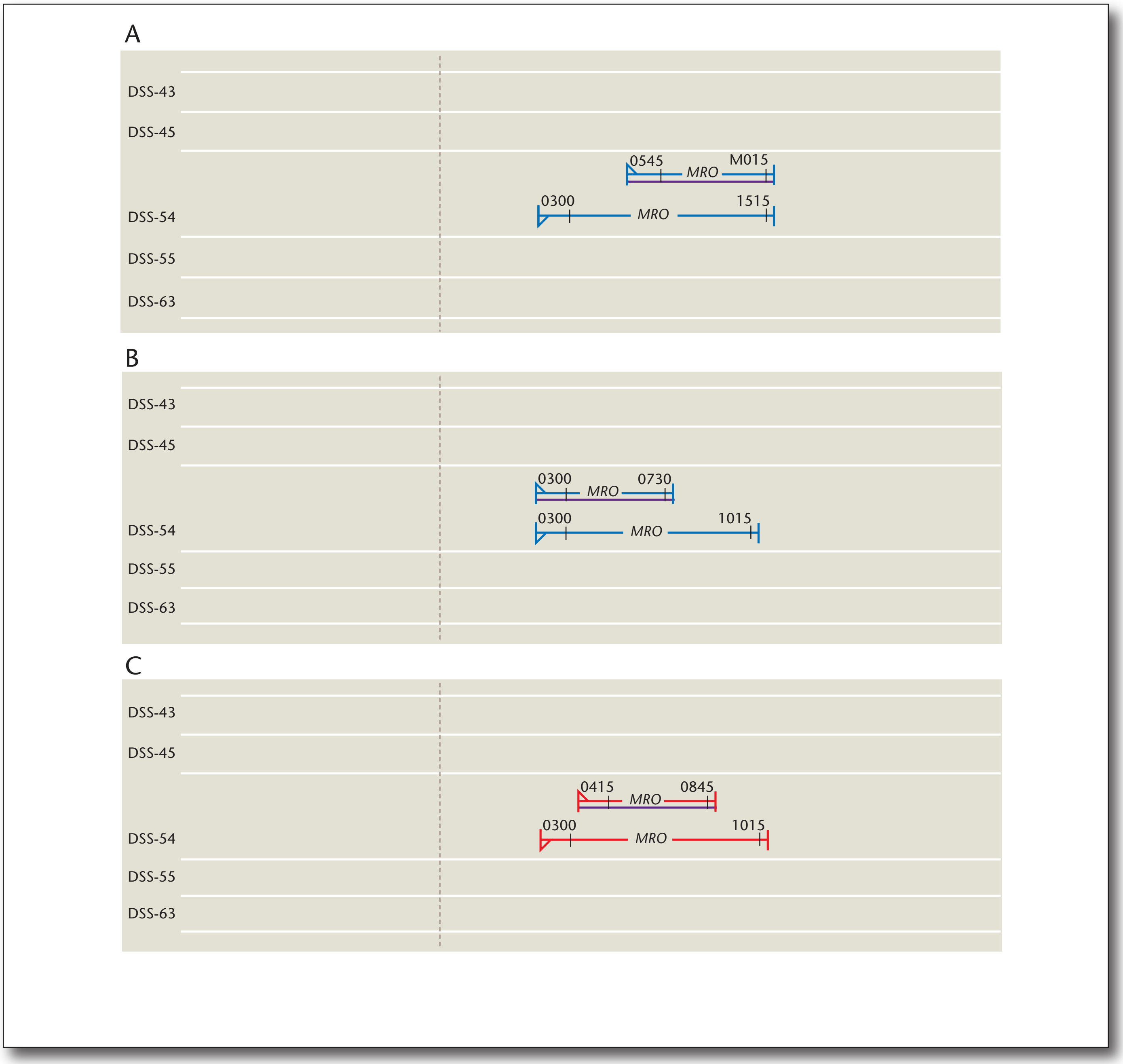


Figure 7. An Example of Multiple Track Reconfiguration Conflicts.

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

grated with the scheduling system graphical user interface (GUI), for visualization along with the actual schedule itself; and (2) as incorporated into the DSE algorithm set, for invocation as strategies or heuristic repair and rescheduling options that can be included or not into the normal scheduling process

Integration with the S^3 GUI has built upon the deployed S^3 HTML5 canvas-based GUI (see figure 2), which has enabled the rapid extension of the GUI to additional visualization elements. Examples of the visualization of each of the major categories of time line requirements follow.

The total time requirement applies to about 25 percent of the DSN user set, but over a wide range of time scales, from a full week on down to a fraction of a single day. An example for the GRAIL A/B mission (two spacecraft in lunar orbit) is shown in figure 8a.

The tracking gaps time line requirement applies to about a third of the DSN user set. In some cases, the gaps of concern are only for certain activity types, as illustrated in figure 8b where gaps are only significant between adjacent ranging passes.

About 20 percent of users have DSN complex distribution requirements, but this varies depending on the phase of the mission. These requirements are typically driven by navigation considerations, where it is important to have ranging data from widely separated baselines in order to reduce ephemeris errors. Examples are shown in figure 8a–c, where satisfaction or violation of the distribution requirement is clearly visible.

While most missions have on-board recorders, only a handful can potentially be modeled simply enough to include in the early stages of DSN scheduling. For those missions with uniform data collections rates and well-defined downlink rules, the recorder time line requirement can provide early visibility into recorder capacity and how it is affected by specific scheduling choices. An example is shown in figure 8c for the STEREO A/B spacecraft.

By providing a highly visual view of these time line constraints and preferences, users who are working on schedule changes to resolve conflicts can immediately see whether their proposed changes would introduce any violations. Presently, many scheduling users have custom scripts that they use to evaluate proposals from other users, but by providing for common models and visibility, feedback can be provided much more rapidly. This feedback has the potential to reduce the overall negotiation process effort and duration.

Overview of Scheduling Strategies

There are a few basic design principles around which the DSE algorithms have been developed, derived from the 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, that is, 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) and 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 autogeneration 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 autogeneration, or by end users when working on specific conflicted portions of the schedule.

Expanding Requests to Tracks

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 at 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 and durations 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.

Relaxation proceeds in two stages, based on schedule content, then on constraint parameters. In the first phase, if no conflict-free allocation can be found, the engine successively ignores lower priority activities, then those of equal priority, and finally all preexisting activities. If there is still no satisfying allocation, then requirement parameters are relaxed in the following order: (1) timing relationships, (2) gap and overlap parameters for split tracks, and (3) constraining event windows. Ultimately, only antenna-to-spacecraft visibility intervals are considered and an activity of the specified duration is created to overlap one of these.

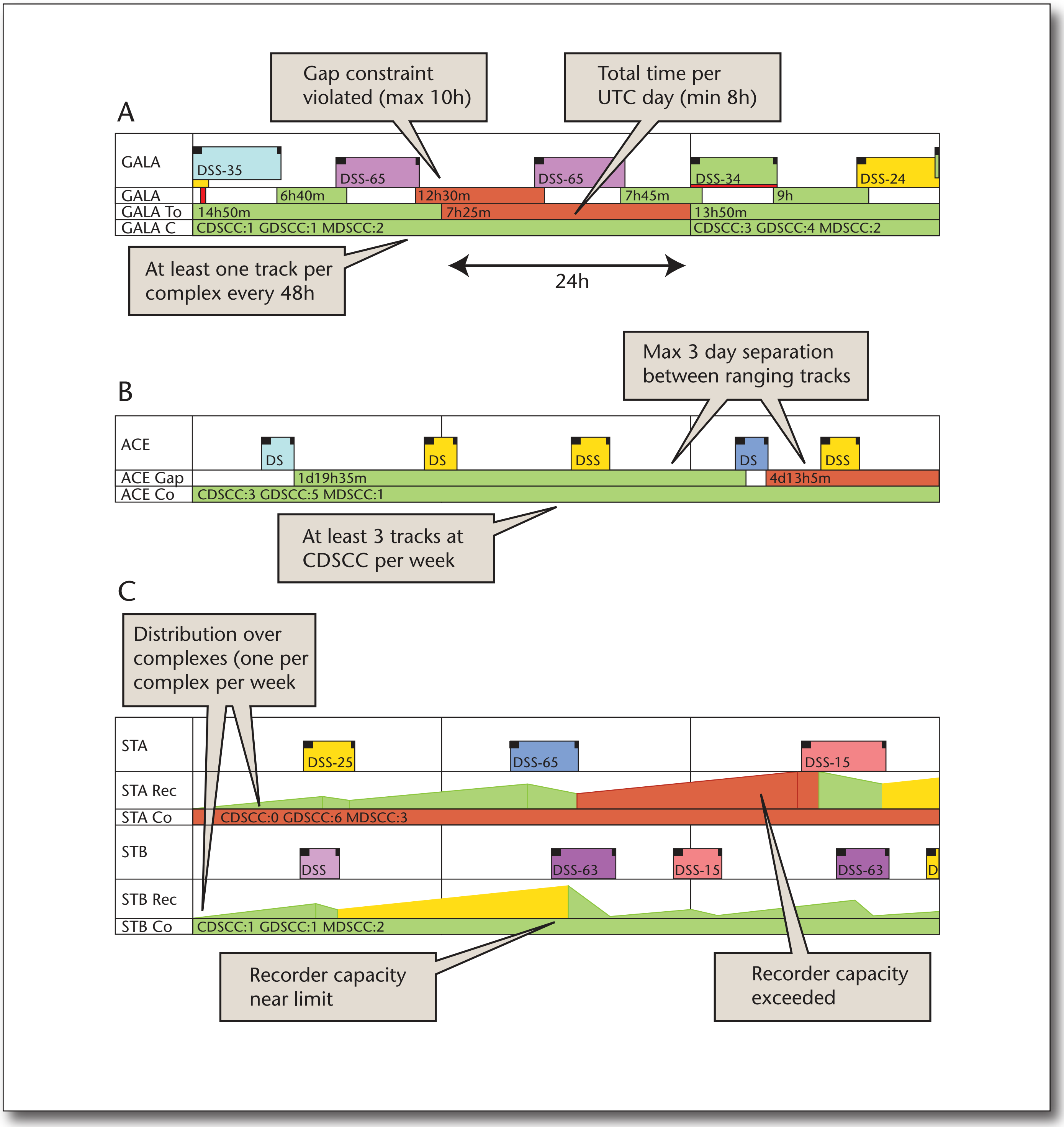


Figure 8. Time Line Constraints for Three Representative Spacecraft, Depicted in the S^3 Scheduling HTML5 GUI.

(a) Example of multiple time line requirements applied to a single spacecraft, here GRAIL A, one of a pair of lunar orbiters. There is a gap constraint and a minimum tracking time constraint in a 24-hour UTC day (both violated and colored red); there is also a requirement to track on all three DSN complexes within a 48-hour period (satisfied). (b) Example of a gap constraint between ranging passes only, that is, ignoring the intervening tracking passes. In this example, the second maximum gap requirement has been violated and the resulting interval is colored red. (c) Example of a recorder time line constraint applied to the STEREO A/B mission pair, showing the violation of the constraint in an interval where the accumulated data would exceed the recorder capacity. Note that the recorder volume drops more quickly when a 70-meter contact (such as DSS-63) is scheduled, due to the higher downlink data rate. The STEREO spacecraft also have a requirement to schedule at least one track per week at each complex, here satisfied only for STEREO B.

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, an 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 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 reassign 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 with timing constraints to two preceding and two following activities, and additionally have irregular and highly restrictive interval constraints. 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

Once an initial schedule has been generated, conflicts and/or violations may exist in the schedule due to the relaxation of constraints (Johnston and Giuliano 2011). The DSE provides schedule repair algorithms to reduce conflicts or violations. These algorithms 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.

The stochastic relayout algorithm generates a new schedule based on adjustments made to 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, and so on. Each new schedule generated attempts to reduce the number of track conflicts and request violations, thus addressing the issue with single-requirement 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 more than 40 percent.

Conclusions

We have described the DSN scheduling engine 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 more than a year, the DSE with only a test GUI 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 full operation, with a browser-based GUI supporting a geographically distributed user base engaged in collaborative peer-to-peer scheduling.

At the present time, the S^3 software is being extended to handle long-range and forecasting functionality. By necessity, there are many similarities between the DSN mid- and long-range planning and scheduling functions. Underlying both is the set of current and future DSN assets, including antennas and equipment, some coming into service and others being decommissioned. Both are based on DSN usage requirements from a varying mission set with a wide range of time-dependent tracking and navigation needs. Both are charged with arriving at an ultimately feasible allocation of DSN resources by balancing user needs and resolving periods of resource contention.

Building on these similarities, the first phase of development of the loading analysis and planning software (LAPS) will make direct use of a number of capabilities already deployed operationally in the midrange S^3 software (see figure 9), including the model of DSN asset availability for antennas and equipment, user and mission types, multispacecraft constellations, and MSPA groupings and their special scheduling rules. Additionally, LAPS will be able to

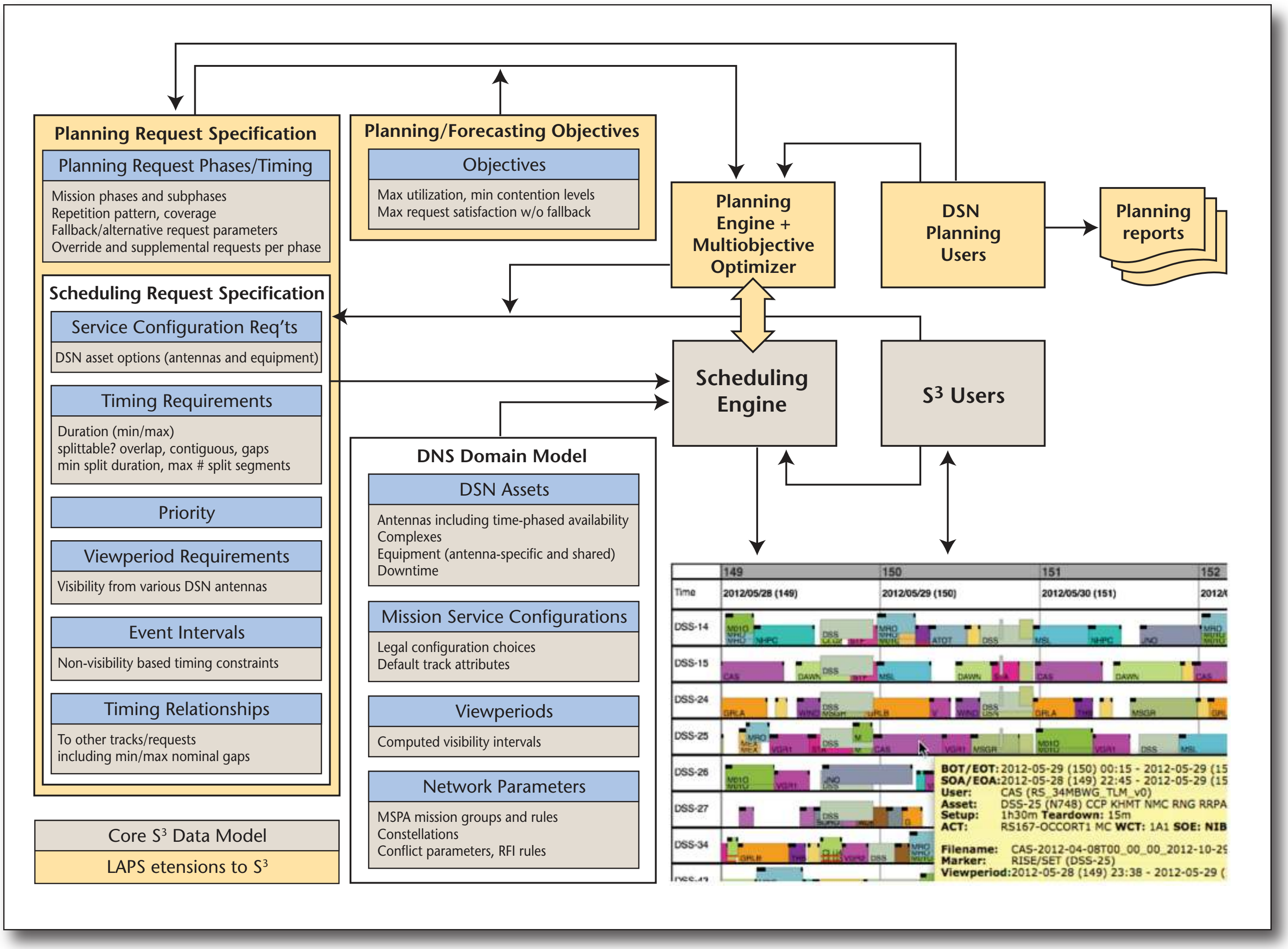

Planning Request Specification
Planning Request Phases/Timing
Mission phases and subphases
Repetition pattern, coverage
Fallback/alternative request parameters
Override and supplemental requests per phase
Scheduling Request Specification
Service Configuration Req'ts
DSN asset options (antennas and equipment)
Timing Requirements
Duration (min/max)
splittable? overlap, contiguous, gaps
min split duration, max # split segments
Priority
Viewperiod Requirements
Visibility from various DSN antennas
Event Intervals
Non-visibility based timing constraints
Timing Relationships
To other tracks/requests
including min/max nominal gaps
Core S³ Data Model
LAPS etensions to S³
Planning/Forecasting Objectives
Objectives
Max utilization, min contention levels
Max request satisfaction w/o fallback
DNS Domain Model
DSN Assets
Antennas including time-phased availability
Complexes
Equipment (antenna-specific and shared)
Downtime
Mission Service Configurations
Legal configuration choices
Default track attributes
Viewperiods
Computed visibility intervals
Network Parameters
MSPA mission groups and rules
Constellations
Conflict parameters, RFI rules
Planning Engine + Multiobjective Optimizer
DSN Planning Users
Planning reports
Scheduling Engine
S³ Users
Time
DSS-14
DSS-15
DSS-24
DSS-25
DSS-26
DSS-27
DSS-34
BOT/EOT: 2012-05-29 (150) 00:15 - 2012-05-29
SOA/EOA: 2012-05-28 (149) 22:45 - 2012-05-29
User: CAS (RS_34MBWG_TLM_v0)
Asset: DSS-25 (N748) CCP KHMT NMC RNG RRPA
Setup: 1h30m Teardown: 15m
ACT: RS167-OCCORT1 MC WCT: 1A1 SOE: NIB
Filename: CAS-2012-04-08T00_00_00_2012-10-29
Marker: RISE/SET (DSS-25)
Viewperiod: 2012-05-28 (149) 23:38 - 2012-05-29

Figure 9. The DSE Data Model and Key Data Flows.

The figure illustrates user interactions with the DSN scheduling engine. The extension of midrange S^3 functionality to support long-range planning and forecasting is highlighted.

invoke the DSE algorithms used in the midrange process, which will allow for fully detailed what-if generation of hypothetical midrange schedule periods in those cases where sufficient detail is available to warrant this level of analysis.

Several other areas are also being addressed with additional capabilities including the following: (1) a planning request representation to allow for more abstract and high-level specification of allocation needs than the scheduling requirement model allows (for example 3x 8hr tracks/week on 34-meter BWG for the 6 months of interplanetary cruise); at the same time, planning requests will be convertible automatically into midrange scheduling requests in order to minimize duplicate data entry and speed up the midrange process; (2) the capability to define and run planning scenarios in an automated way, such as to assess a range of options for down time placement; to evaluate nominal and fallback requirement options for resource contention periods; and to quantify the impact of a mission's alternative launch dates on projected resource loading; and (3) a multiobjective optimization mechanism to automatically generate a portfolio of candidate plans/schedules optimizing the trade-offs among multiple quantitive objectives.

The incorporation of multiobjective optimization (for example, Brown and Johnston [2013]; Johnston [2006]) into LAPS offers a new way to optimize DSN resource allocations, taking into account that there is no single objective that captures all of the disparate goals and objectives that are important. Multiobjective optimization has been employed in a wide variety of problem domains, including scheduling for science missions and generating some requirements inputs to the DSN midrange process (Johnston and Giuliano 2011).

Beyond long-range planning and forecasting, future work includes extending the scope of S^3 to support near real-time scheduling and cross-network scheduling scheduling capabilities.

Extending the scope of S^3 to support near real-time scheduling, the third phase of the DSN scheduling process, covers the period from execution out to some number of weeks in the future. Extending S^3 to support this phase involves some challenging technical problems of integration with existing systems and support for contingency scheduling (for example, launch slips, unplanned asset down time) as well as operation at the remote DSN complexes; at the same time, bringing the information model of S^3 into the real-time domain will allow for improved decision making considering options that are not now accessible

In addition to the Deep Space Network, NASA also operates two other networks with similar communications and navigation support for certain types of missions: these networks are the Space Network (SN) and 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 intervals and service definitions.

Acknowledgments

The research described in this article 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. 1993. Scheduling Deep Space Network Data Transmissions: A Lagrangian Relaxation Approach. In *Applications of Artificial Intelligence 1993: Knowledge-Based Systems in Aerospace and Industry.* SPIE 1963. Bellingham, WA: Society of Photo-Optical Instrumentation Engineers.

Biefeld, E., and Cooper, L. 1991. Bottleneck Identification Using Process Chronologies. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence,* 218–224. San Francisco: Morgan Kaufmann Publishers.

Brown, M., and Johnston, M. D. 2013. Experiments with a Parallel Multiobjective Evolutionary Algorithm for Scheduling. Paper presented at the 5th International Workshop on Planning and Scheduling for Space, Baltimore, MD, 22–25 October.

Carruth, J.; Johnston, M. D.; Coffman, A.; Wallace, M.; Arroyo, B.; and Malhotra, S. 2010. A Collaborative Scheduling Environment for NASA's Deep Space Network. Paper presented at the 10th International Conference on Space Operations (SpaceOps 2010), Huntsville, AL, 25–30 Apr.

Chien, S.; Johnston, M. D.; Frank, J.; Giuliano, M.; Kavelaars, A.; Lenzen, C.; and Policella, N. 2012. A Generalized Timeline Representation, Services, And Interface For Automating Space Mission Operations. Paper presented at the 12th International Conference on Space Operations (SpaceOps 2012), Stockholm, Sweden, 11–15 June.

Chien, S.; Rabideau, G.; Knight, R.; Sherwood, R.; Engelhardt, B.; Mutz, D.; Estlin, T.; Smith, B.; Fisher, F.; Barrett, T.; Stebbins, G.; Tran, D. 2000. ASPEN — Automating Space Mission Operations Using Automated Planning and Scheduling. Paper presented at the 1st International Conference on Space Operations (SpaceOps 2000), Toulouse, France, 19-23 June.

Chien, S. A.; Hill, R. W. J.; Govindjee, A.; Wang, X.; Estlin, T.; Griesel, M. A.; Lam, R.; Fayyad, K. V. 1997. A Hierarchical Architecture for Resource Allocation, Plan Execution, and Revision for Operation of a Network of Communications Antennas. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*. Piscataway, NJ: Institute of Electrical and Electronics Engineers. dx.doi.org/10.1109/ROBOT.1997.606798

Clement, B. J., and Johnston, M. D. 2005. The Deep Space Network Scheduling Problem. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence,* 1514–1520. Menlo Park, CA: AAAI Press.

Fisher, F.; Chien, S.; Paal, L.; Law, E.; Golshan, N.; and Stockett, M. 1998. An Automated Deep Space Communications Station. In *Proceedings of the 1998 IEEE Aerospace Conference,* volume 3. Piscataway, NJ: Institute of Electrical and Electronics Engineers. dx.doi.org/10.1109/AERO.1998.685791

Guillaume, A.; Lee, S.; Wang, Y.-F.; Zheng, H.; Hovden, R.; Chau, S.; Tung, Y.-W.; Terile, R. J. 2007. Deep Space Network Scheduling Using Evolutionary Computational Methods, 1–6. In *Proceedings of the 2007 IEEE Aerospace Conference*. Piscataway, NJ: Institute of Electrical and Electronics Engineers.

Imbriale, W. A. 2003. Large Antennas of the Deep Space Network. New York: Wiley. dx.doi.org/10.1002/0471728497

Johnston, M. D. 2006. Multi-Objective Scheduling for NASA's Deep Space Network Array. Paper presented at the 5th International Workshop on Planning and Scheduling for Space (IWPSS-06), Baltimore, MD, 22–25 October.

Johnston, M. D., and Clement, B. J. 2005. Automating Deep Space Network Scheduling and Conflict Resolution. Paper presented at the 2005 International Symposium on Artificial Intelligence, Robotics, and Automation for Space, Munich, Germany 5–9 September.

Johnston, M. D., and Giuliano, M. 2011. Multi-Objective Scheduling for the Cluster II Constellation. Paper presented at the 7th International Workshop on Planning and Scheduling for Space 2011. IWPSS-2011 Darmstadt, Germany, June 8–10.

Johnston, M. D.; Tran, D.; Arroyo, B.; and Page, C. 2009. Request-Driven Scheduling for NASA's Deep Space Network. Paper presented at the 6th International Workshop on Planning and Scheduling for Space (IWPSS-06), Pasadena, CA, 19–21 July.

Johnston, M. D.; Tran, D.; Arroyo, B.; Call, J.; and Mercado, M. 2010. Request-Driven Schedule Automation for the Deep Space Network. Paper presented at the 10th International Conference on Space Operations (SpaceOps 2010), Huntsville, AL, 25–30 Apr.

Kan, E. J.; Rosas, J.; and Vu, Q. 1996. Operations Mission Planner — 26M User Guide Modified 1.0. JPL Technical Document D-10092. Pasadena, CA: Jet Propulsion Laboratory, California Institute of Technology.

Lacey, N., and Morris, D. G. 2002. JPL RAPSO Long-Range Forecasting. Paper presented at the 12th AAS/AIAA Space Flight Mechanics Meeting, San Antonio, Texas, 27–30 January.

Loyola, S. J. 1993. PC4CAST: A Tool for DSN Load Forecasting and Capacity Planning. NASA Technical Report 19940009911. Pasadena, CA: Jet Propulsion Laboratory, California Institute of Technology.

Werntz, D.; Loyola, S.; and Zendejas, S. 1993. FASTER — A Tool for DSN Forecasting and Scheduling, 230–235. Paper presented at the 9th American Institute of Aeronautics and Astronautics (AIAA) Computing in Aerospace Conference, October 19–21, San Diego, CA.

Mark D. Johnston is a principal scientist in the Planning and Execution Systems section at the Jet Propulsion Laboratory, California Institute of Technology. His research interests include reactive planning and scheduling, multiobjective optimization, and evolutionary algorithms. His background is in physics, with a bachelor's degree from Princeton and Ph. D. from MIT. He has worked on a wide range of planning and scheduling technologies in both space and commercial domains, and is the originator of the Spike scheduling system, used for Hubble Space Telescope as well as a number of other space- and ground-based observatories. He currently leads the JPL Deep Space Network Service Management development team, and also the SCaN network integration project increment 1 team.

Daniel Tran is a senior member of the technical staff in the Artificial Intelligence Group at the Jet Propulsion Laboratory, California Institute of Technology. Tran received a B.S. in computer engineering from the University of Washington and M.S. in computer science from the University of Southern California. He was the flight software lead for the autonomous sciencecraft experiment, cowinner in the 2005 NASA Software of the Year competition. He was also software lead for the service scheduling software (SSS), the primary system used for midrange scheduling of NASA's Deep Space Network.

Belinda Arroyo manages the scheduling process for NASA's Deep Space Network. She has more than 20 years experience at JPL in mission scheduling, mission operations, and deep space navigation, and is the recipient of a number of awards in these areas. She has overseen the deployment of the new scheduling software described in this article, and the consequent adaptation of scheduling processes.

Sugi Sorensen is a senior systems engineer at the Jet Propulsion Laboratory. He currently provides systems engineering support to the Deep Space Network project in the scheduling process realm. He also serves as the operations engineer for the service scheduling software (SSS) system. His previous work at JPL includes systems engineering support for the Planetary Data System and he served as a group supervisor and product manager in the Advanced Concepts Engineering group where he specialized in human computer interface design and web-based information systems.

Peter Tay is a user support specialist at the Jet Propulsion Laboratory, using the service scheduling software (SSS) for the Deep Space Network midrange scheduling process. He is responsible for reducing hard contentions to a manageable level for the DSN scheduling community, before they begin their negotiation process. Tay has previously supported various missions as flight project schedule engineer for 16 years, including the Mars missions, Cassini, Voyagers 1 and 2, Stardust, and Genesis, as well for several non-NASA missions.

Butch Carruth is founder and president of Innovative Productivity Solutions, Inc., specializing in software-based task automation. His work in the space industry began in 2000 as architect of the science planning software for the Cassini mission to Saturn. He and his team have been working with the Jet Propulsion Laboratory on software solutions for the Deep Space Network since 2006. He continues to support a variety of deep space and low Earth orbit missions in a software development and data management capacity.

Adam Coffman is a senior developer at Innovative Productivity Solutions where he started in 2002 after graduating with a computer science degree from Trinity University in 2001. He started on the Cassini mission in 2001 and as been focused on collaborative scheduling solutions for the Deep Space Network since 2006.

Mike Wallace is a software engineer at IPS, Inc. He works on database systems, website design, and data analysis. He has worked on multiple NASA software projects including visualization tools for mission planning and mission design, telemetry analysis and storage, and antenna scheduling.