Overview

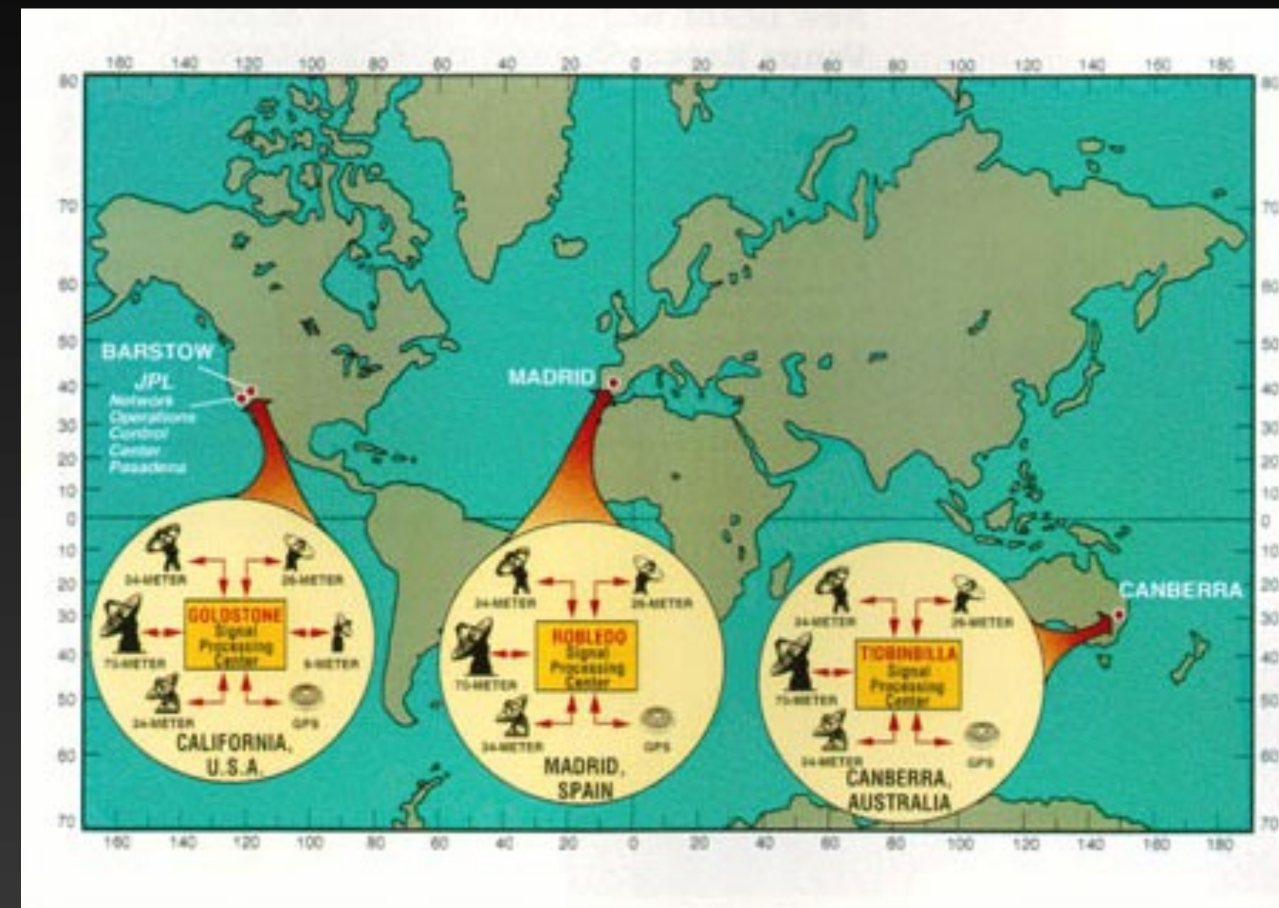
- **Deep Space Network (DSN) scheduling**
  - domain overview
  - scheduling requests, constraints and preferences
  - expanding requests, resolving conflicts, and constraint relaxation
- **Multi-Objective Scheduling**
  - evolutionary algorithms
  - application to DSN
  - application to Cassini science planning

Mark D. Johnston

Jet Propulsion Laboratory, California Institute of Technology

The Current Deep Space Network

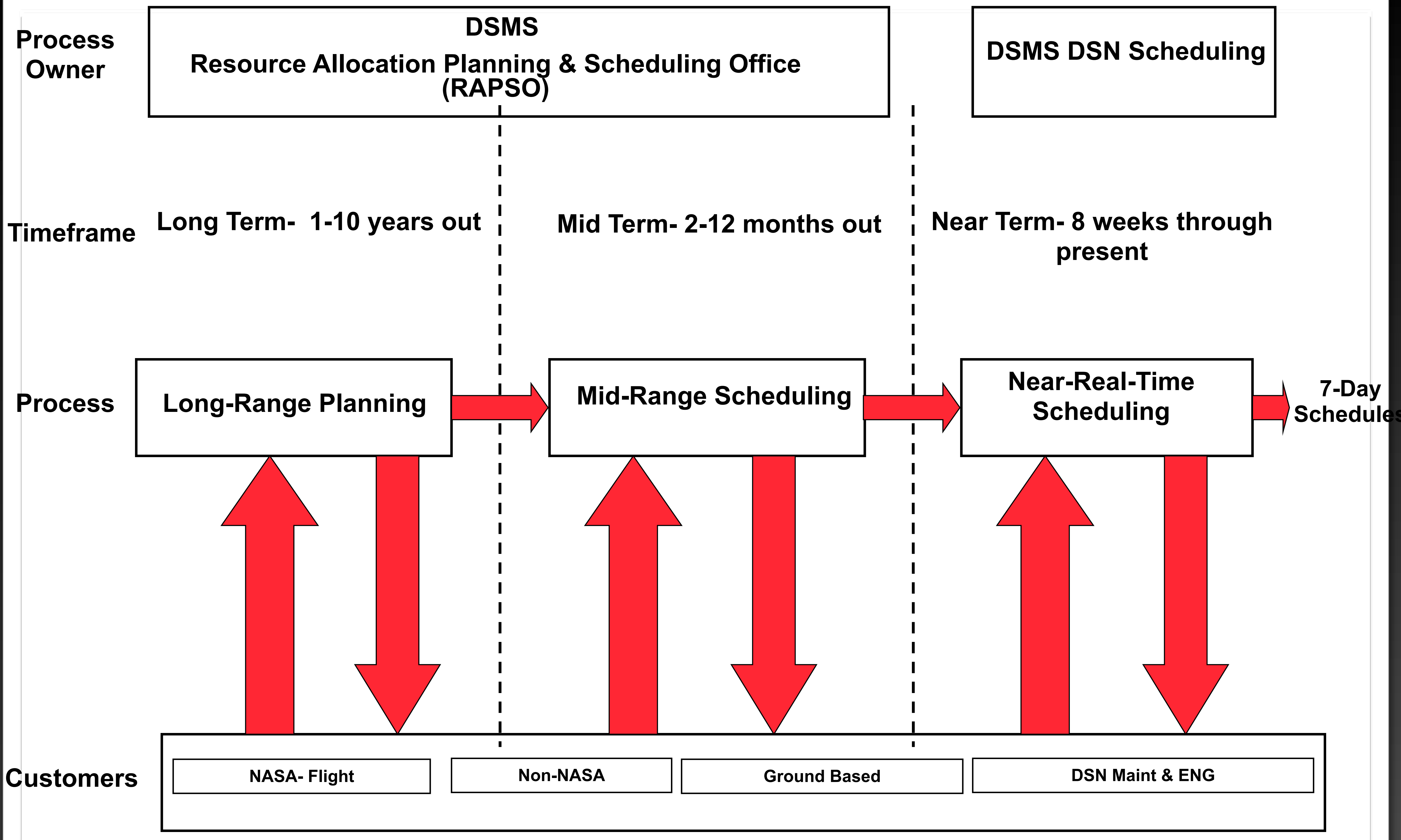
- Current DSN comprises
  - 3 sites roughly equally spaced in longitude
  - 26m, 34m, 70m antennas, at each site
- DSN supports all planetary missions + some earth orbiters
- Drivers on evolution include:
  - more missions (3x by 2030)
  - data rates and volumes increasing by 100x by 2030
  - manned missions place stringent availability and reliability requirements
  - more cluster (multi-spacecraft) missions
  - reduce costs (operations and maintenance)
  - maintain high level of 24x7 support



Deep Space Network Scheduling

- 56 missions
- 12 antennas
  - different capabilities
  - shared equipment
  - geometric constraints
  - human operator constraints
- some schedule as long as 10 years into future
- ~370 tracks & ~1650 viewperiods per week
- ~2000 tracks & ~80000 viewperiods per year
- some require schedule freeze 6 months out
- complicated requirements originally from agreement with NASA with flexibility in antennas, timing, numbers of tracks, gaps, etc.
- schedule centrally generated, meetings and horse trading to resolve conflicts
- ~30 people employed full time to schedule for multiple missions
- similar to coordination operations across missions

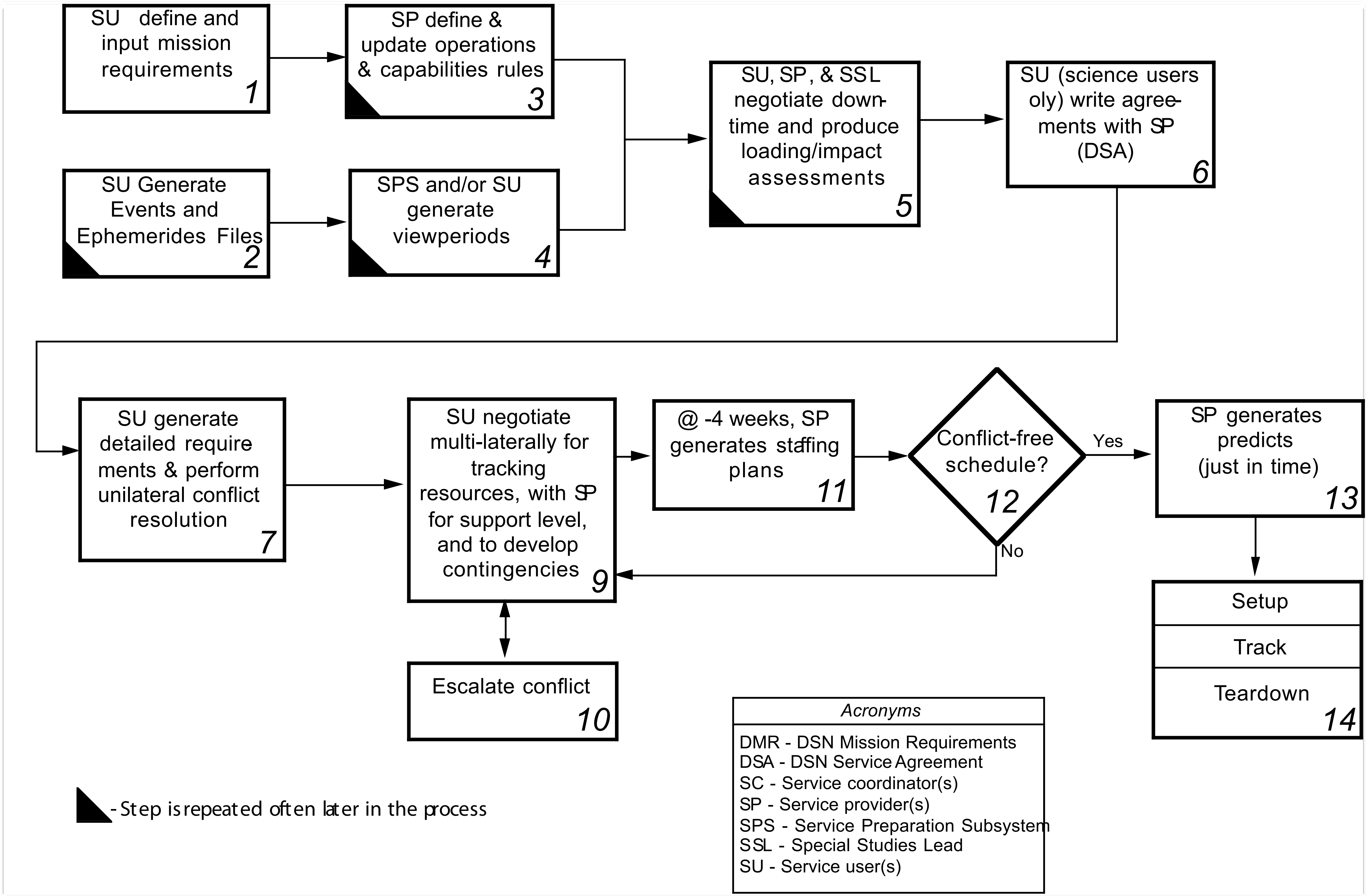
DSN Resource Allocation Process



Key Roles

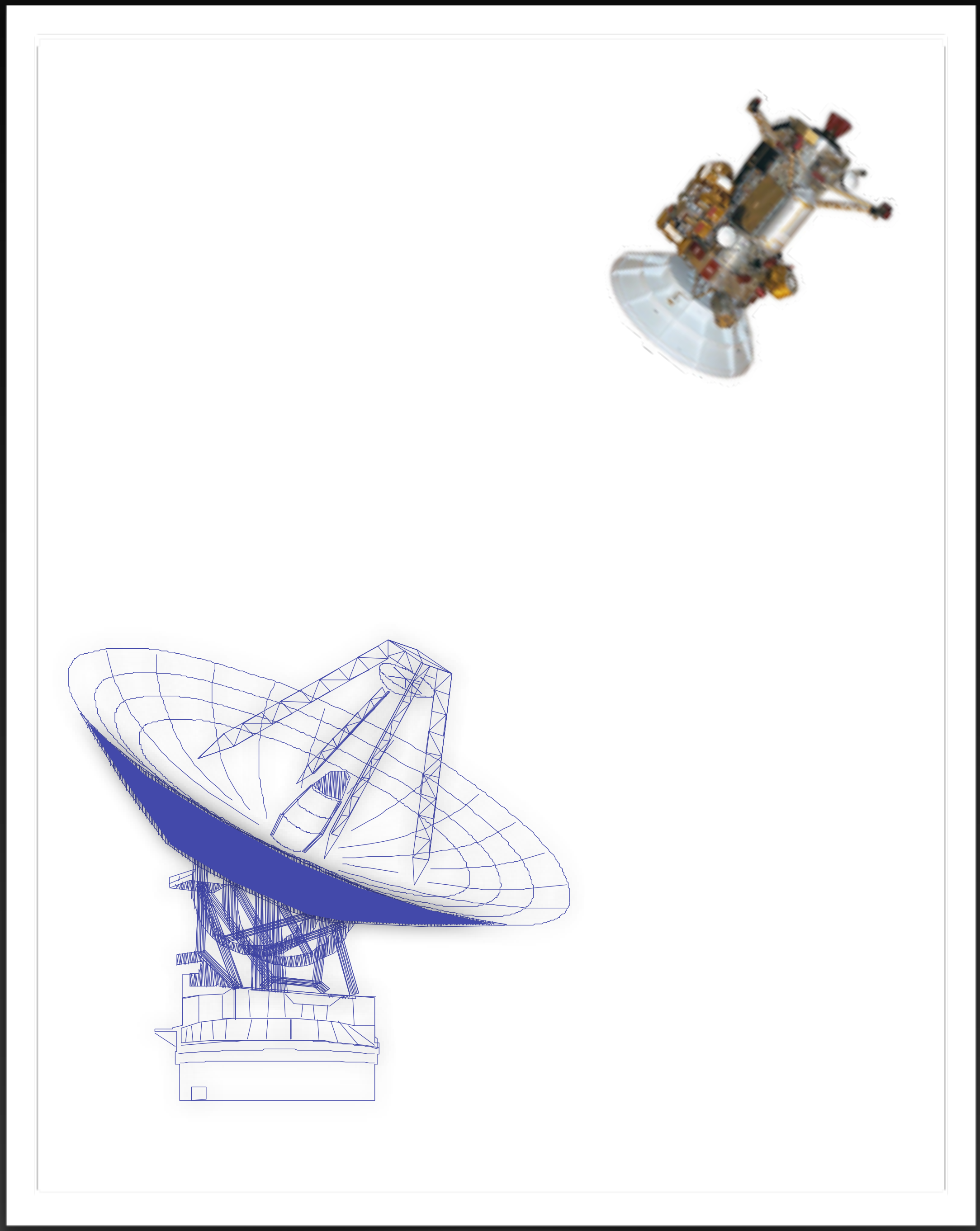
- General
  - **Service User (SU)**- Anyone who requests allocation of DSN assets. These include flight projects' schedulers and mission planners, and DSN maintenance. Also includes those simply requesting reports from SSAS
  - **Service Provider (SP)** - DSN and its agents involved in building, operating, or maintaining DSN assets
  - **Special Studies Lead (SSL)** is responsible for leading long-range loading studies in support of Project formulation activities and coordinating DSN downtime
- New with S^3:
  - **Service Coordinator** is responsible for pushing the community to resolve conflicts and who has the authority to call coordination meetings and to escalate conflicts that have reached an impasse

DSN Scheduling Process



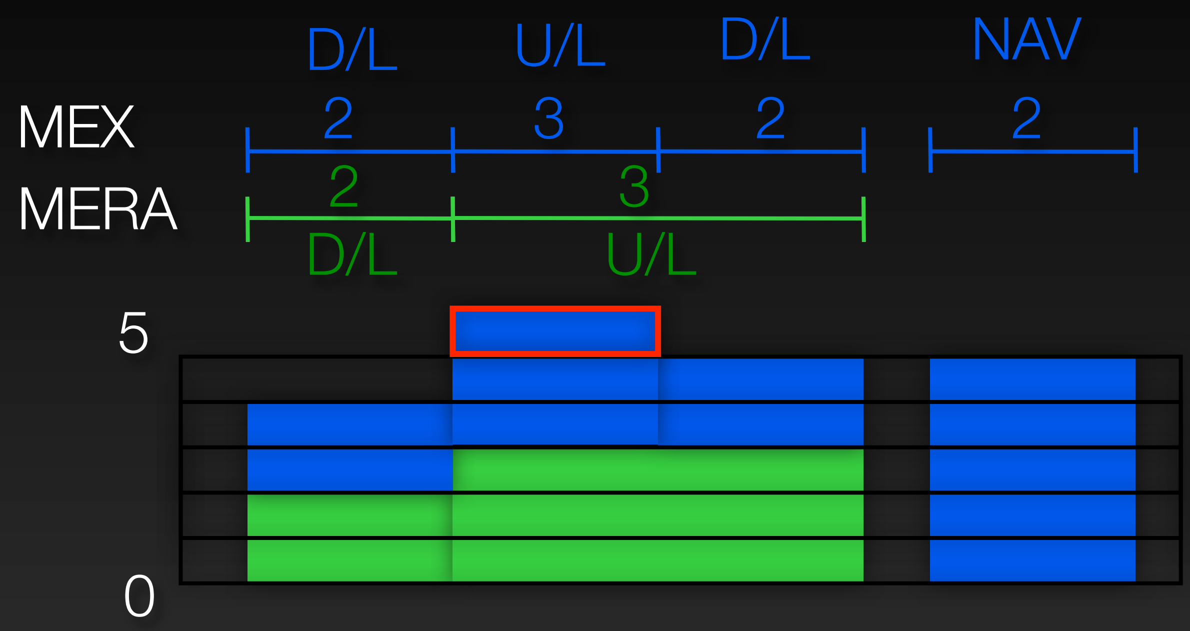
Tracks, Viewperiods & Activities

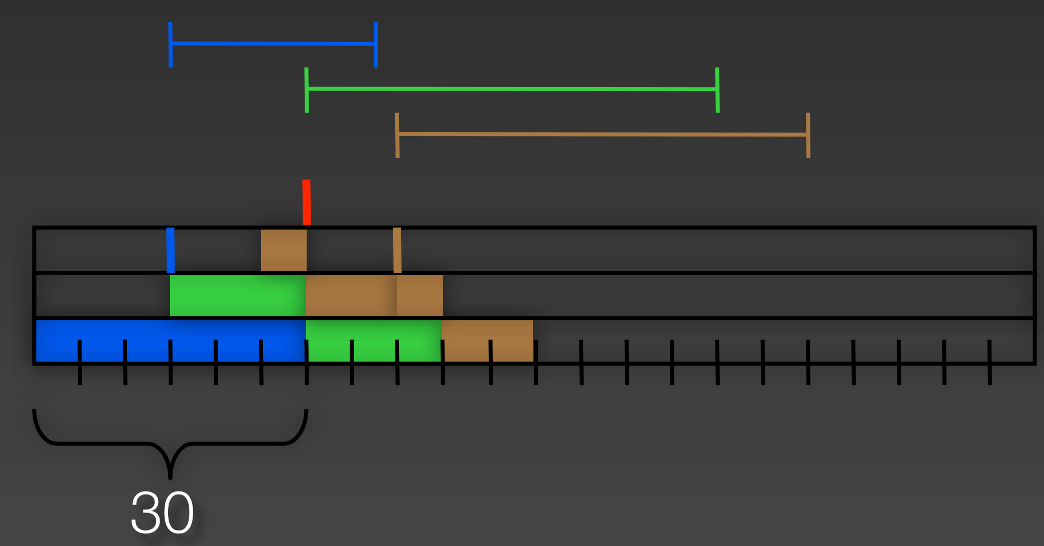
- A track is an allocation of an antenna to a mission over some time interval
- A viewperiod is the time interval when a spacecraft is visible to an antenna
- An activity is a track plus setup and teardown time



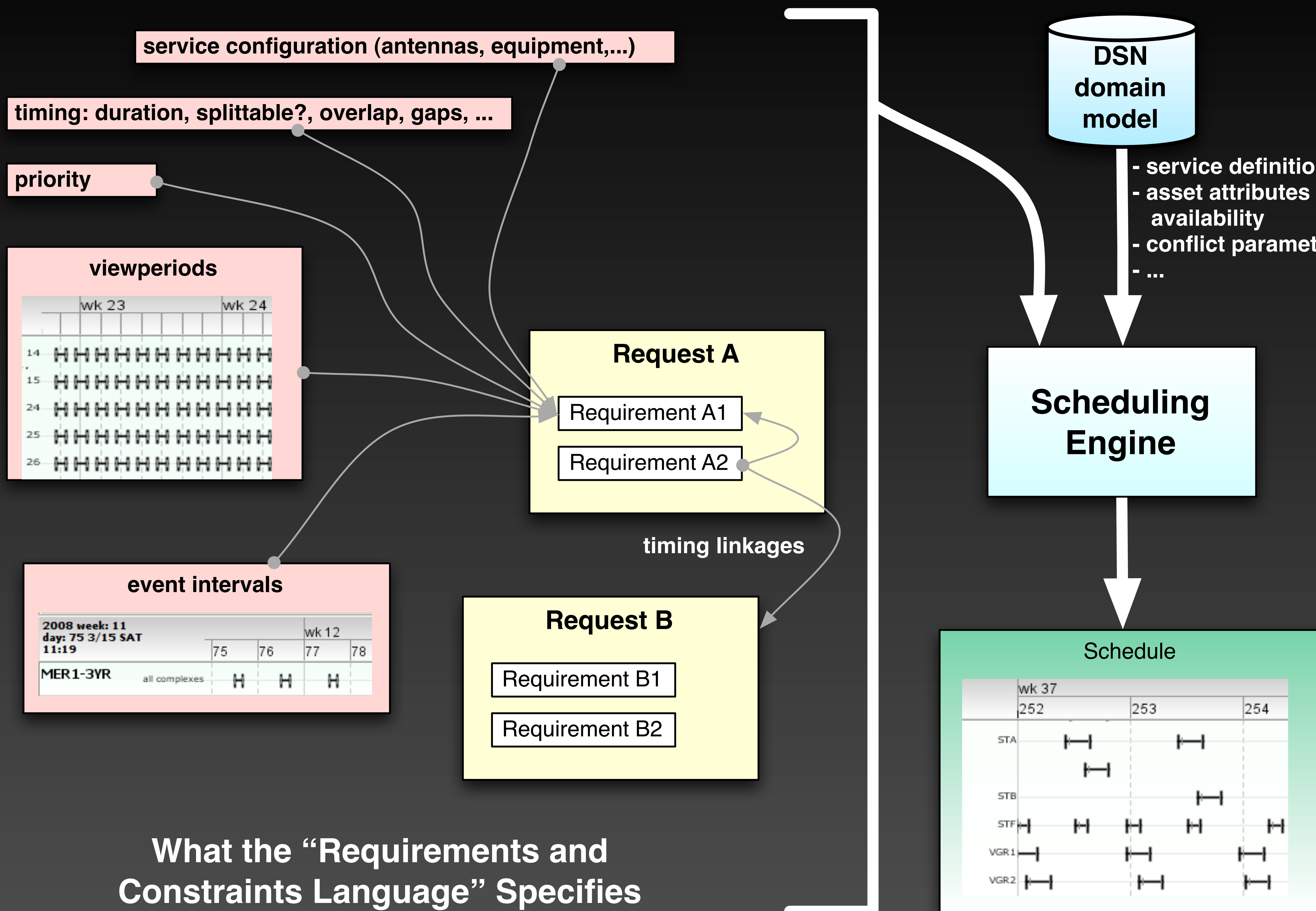
Constraints

- No two spacecraft can use antenna at same time
  - except MSPA where antenna points to both (2 at most) and uplinks to at most one
- Spacecraft must be in view of antenna
- At Goldstone, no track/activity can be scheduled where two other tracks/activities start within 15 minutes
  - except the four Cluster s/c
- At other complexes, no two may start within 5 minutes of each other





What Scheduling Requests Specify...



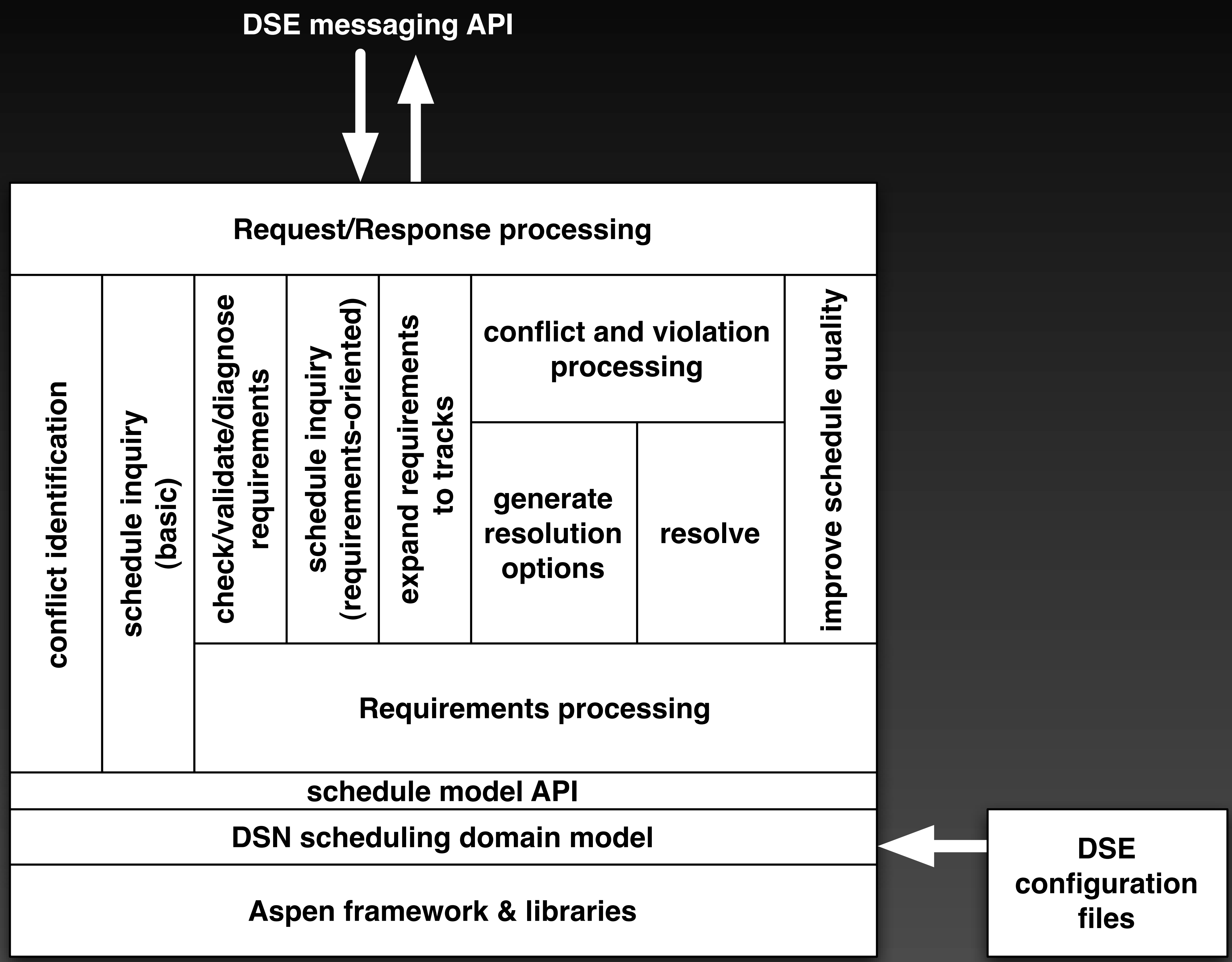
Request-Driven Scheduling

- Challenges
  - request complexity:
    - even apparently simple requests can have complex options
    - detail can become overwhelming
  - complexity makes it hard to check for feasibility
    - interactions of timing constraints with scheduling windows can easily make a request unschedulable
  - accurate representation of scheduling flexibility

Request-Driven Scheduling

- Benefits
  - leveraged effort
    - one request → many scheduled activities
    - reuse: copy/paste/edit requests instead of new from scratch
  - automated continuous schedule validation
    - enables the scheduling system to monitor tracks against constraints and preferences, then notify users of discrepancies
  - automated support for conflict resolution
    - flexibilities described in requests can be used to resolve or suggest resolutions for schedule conflicts
  - traceability
    - all activities trace back to scheduling requests that describe their purpose and intent
    - this is valuable during the schedule conflict negotiation phase, and if rescheduling is required due to equipment outage

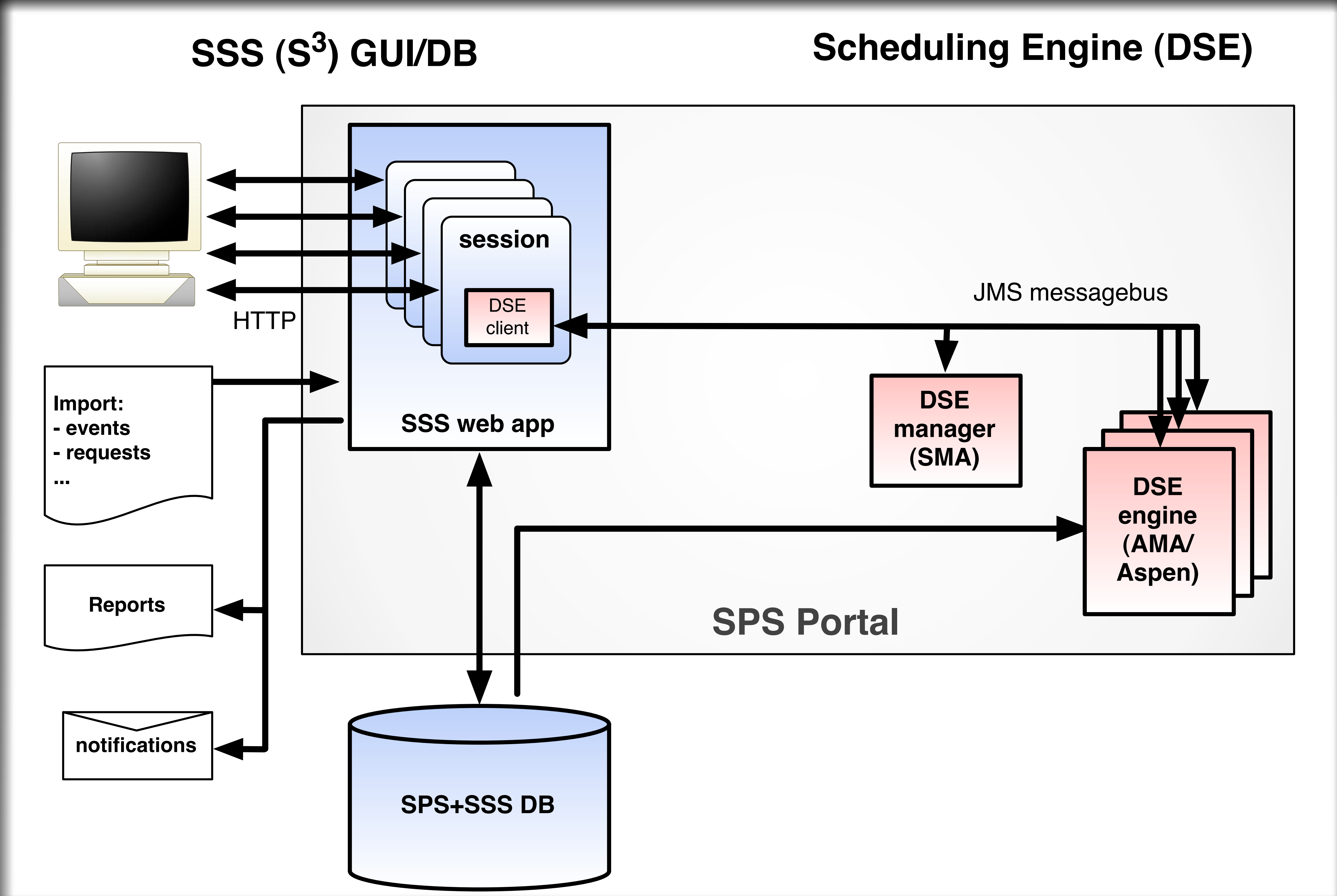
Functional Elements of the DSN Scheduling Engine



DSE Design Principles

- **No unexpected schedule changes**
  - all changes to the schedule must be requested either explicitly or implicitly by the user
  - the same sequence of operations on the same data must yield the same schedule
- **Even for *infeasible* schedule requests, attempt to return something “reasonable” in response**
  - possibly by relaxing aspects of the request, along with a diagnosis of the sources of infeasibility
  - provides a starting point for users to handle the problem

Architectural Overview of S^3 and the DSE

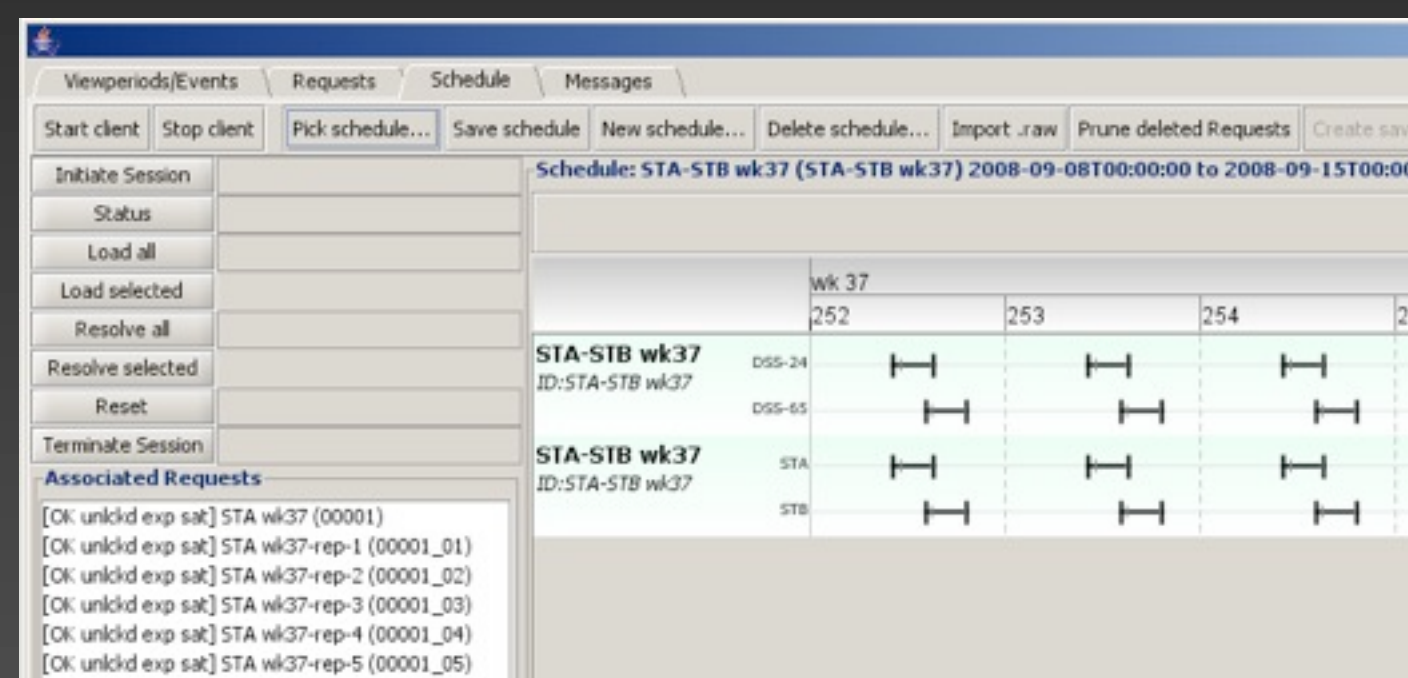


Expanding Requirements

- The engine expands requirements to tracks as follows:
  - try to find conflict- & violation-free allocations
  - if can’t find any, try a series of “fallbacks”:
    - temporarily ignore lower priority tracks, equal priority tracks, all other tracks
    - relax requirement parameters: timing relationships, gap/overlap parameters, event windows, ... ultimately considering only viewperiods
- **Goal:** always try to return something “reasonable” rather than nothing at all (sometimes this can be very hard!)

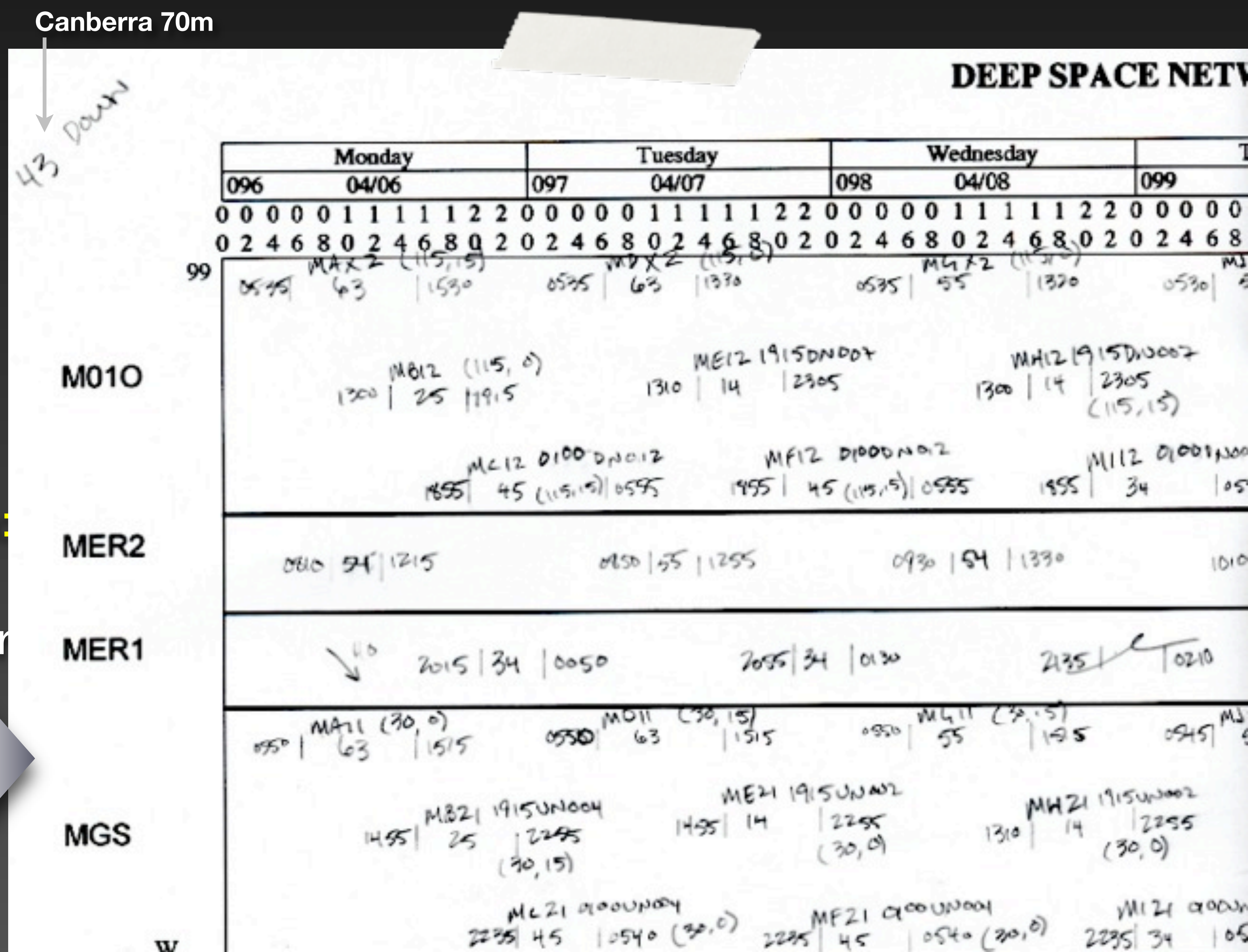
Resolving Conflicts/Violations

- The engine considers as distinct conflicts (on tracks) vs. violations (of requirements)
  - there are basic strategies for focusing on each of these and trying to re-layout requirements to resolve conflicts, fix unsatisfied requirements
  - these strategies exploit flexibilities in the requirements (start time duration, antennas, splitting, gaps, overlaps...) to find good allocations



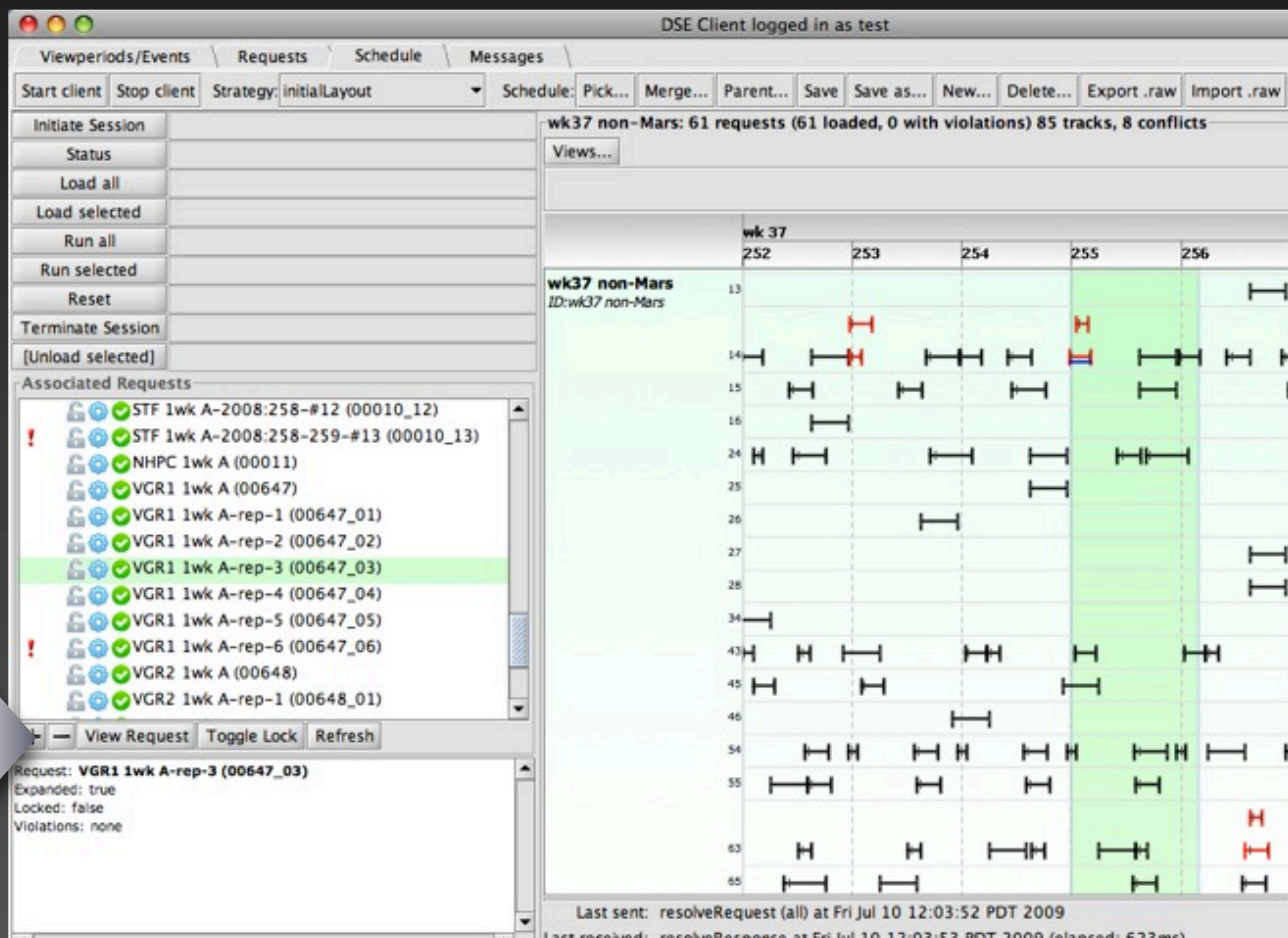
DSE Trial Deployment

- Started a trial deployment in December 2008 to the JPL Multi-mission Resource Scheduling Services (MRSS) team
  - responsible for scheduling 20 of the current 35 active DSN users
    - initial goal: provide automated support for generating the “Mars Integrated” schedule
      - i.e. Mars missions + Cassini + Spitzer Space Telescope
    - then current approach:
      - pencil and paper
      - typed into Excel for delivery to team doing schedule integration



DSE Trial Deployment

- **Started a trial deployment in December 2008 to the JPL Multi-mission Resource Scheduling Services (MRSS) team**
  - responsible for scheduling 20 of the current 35 active DSN users
    - initial goal: provide automated support for generating the “Mars Integrated” schedule
      - i.e. Mars missions + Cassini + Spitzer Space Telescope
    - after:
      - DSE generates integrated schedule
      - all MRSS missions



DSE Status and Future Plans

- Use of DSE has been expanded in June 2009 to incorporate all missions
- Feedback from trial deployment users has been invaluable in defining requirements, tuning functionality
  - far in advance of actual system delivery
- Next steps include
  - extending the request specification language to cover a wider range multi-spacecraft, multi-antenna scheduling scenarios, and to support “distribution” requirements
  - extending scheduling strategies to handle more complex flexibility options
  - user control over relaxation stages in scheduling search

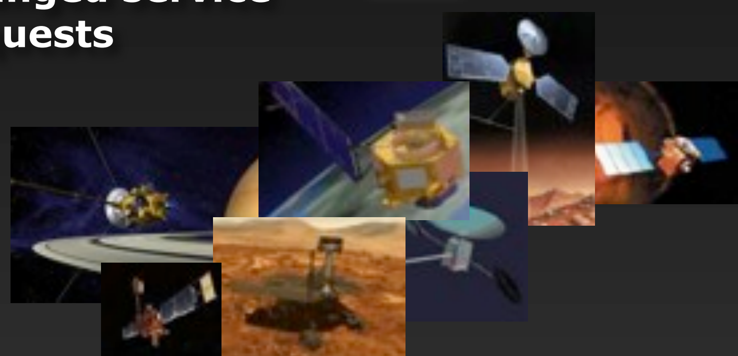
Potential Evolution to an Array Architecture

- A potential array-based network would link together large numbers of less expensive antennas
  - possibly at 3 balanced sites like today's DSN, but possibly unbalanced
  - some designs call for ~400 antennas/site
- Array-based network would offer advantages
  - allocation of antennas could be more granular
  - subsets of the array could be allocated to simultaneous communications with different spacecraft
  - antenna allocation could be time-phased within a single pass
  - unused antennas could be allocated on-the-fly in case of equipment failure or spacecraft emergency
- An array-based network would be much more flexible, but requires an approach to planning and scheduling that can take advantage of this flexibility

Scheduling Context

From multiple users:

- **mission status, capabilities & requirements**
- **new service requests**
- **changed service requests**



scheduler

- **site schedules**

- **site status (availability, maintenance, performance)**
- **weather**
- **execution feedback**

- Optimization objectives —
  - time-varying, from multiple users with very different perspectives, including track-based, service-based, and model based objectives
  - system level objectives include: minimize cost, maximize number of users served
  - no defined way to combine all objectives into a single scalar to optimize

Example user objectives:
Over any specified time interval:
- total coverage time
- # contacts
- min/max contact duration
- min/max gap duration, gap/track ratio
- multi-site simultaneous coverage
- continuous coverage
- “G/T” → # antennas over track (not necessarily constant)

Objectives from a Mission Perspective

- Mission objectives can be categorized based on how they relate to the schedule:
  - pass-based objectives are defined in terms of specific allocations of antennas to missions over some time interval, called a “pass”
    - e.g. 25 antennas at Goldstone dedicated to Mars Phoenix for 6 hours would be a single pass
    - objectives may be defined with respect to attributes of the pass, including duration, timing relative to absolute time or to other passes, etc.
  - service-based objectives are defined at a higher level and refer to missions needs in terms of how well a service requirement is satisfied
    - e.g. a mission may specify that it needs an 8h downlink at a certain data rate every 3 days over some mission phase
  - model-based objectives are even higher level and require the scheduler to model some aspects of mission behavior to asses
    - e.g. need to download data often enough to keep onboard storage from capacity limit

Objectives from a System Perspective

- To some approximation, we can collect the non-mission objectives into a single "system" user, representing the overall operations of an array:
  - satisfy the users of the system
    - captured in user's own objectives
  - minimize operational costs
    - this is largely a function of the system configuration, operations, and maintenance policies, and not likely to be affected much by schedule variations
  - maximize system availability
    - this is a core system objective — maximize the number of users serviced by a given investment in assets and infrastructure
    - counterintuitively, maximize the number of unallocated antennas

Multi-Objective Optimization

- Approach: **multiobjective optimization**
  - keep objectives separate and combine only when necessary
    - do not lose information that separate objectives contain
    - allows an explicit view into the tradeoffs when building and changing the schedule
  - define a very general approach to specifying objectives and constraints
    - penalty function f() applied to schedule as viewed from the user's perspective
    - can switch constraints → objectives to investigate overconstrained situations
  - use multi-objective optimization techniques based on evolutionary algorithms
    - maintain a population of candidate schedules that evolves to optimize $M>1$ objectives
    - population provides an estimate of the Pareto frontier (tradeoff surface)
    - population provides a starting point for schedule changes
    - easy to distribute for processing in parallel
    - enables user visibility into schedule tradeoffs, thus supporting collaborative schedule development, repair, and negotiation

Evolutionary Algorithms

- Formulated as a minimization problem for M objectives subject to K constraints

$$\text{minimize: } \{f_i(\vec{x})\}, \; i = 1 \ldots M$$

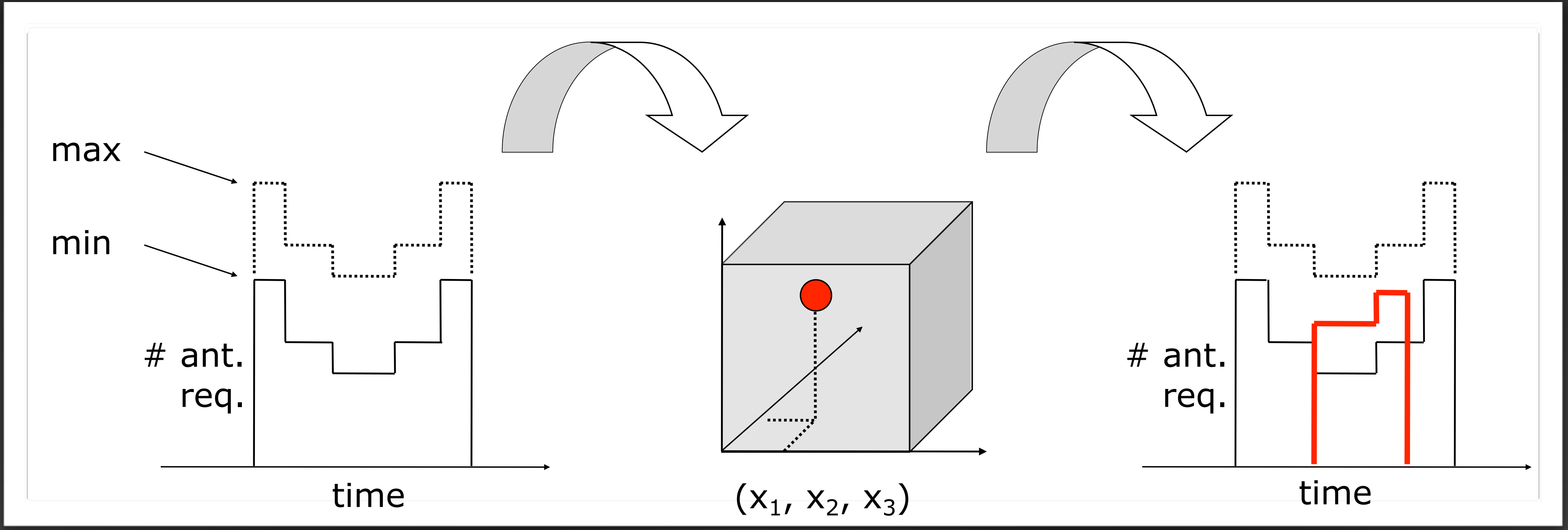
$$\text{subject to: } \left(g_i(\vec{x})\right)^T \leq 0, \; j = 1 \ldots K$$

Here $\vec{x}$ represents a vector in decision space of dimension D. Where necessary below, we refer to the i^{th} member of the population at generation g with $\vec{x}_{i,g}$.

- population of N solution candidates
- With each step (generation), generate a new population following rules for crossover (combining parents to make offspring) and mutation (introducing random variations into offspring)
- Example algorithms with good track records on a variety of problems:
  - NSGA II (Deb et al. 2002)
  - GDE3 (Kukkonen and Lampinen 2005)

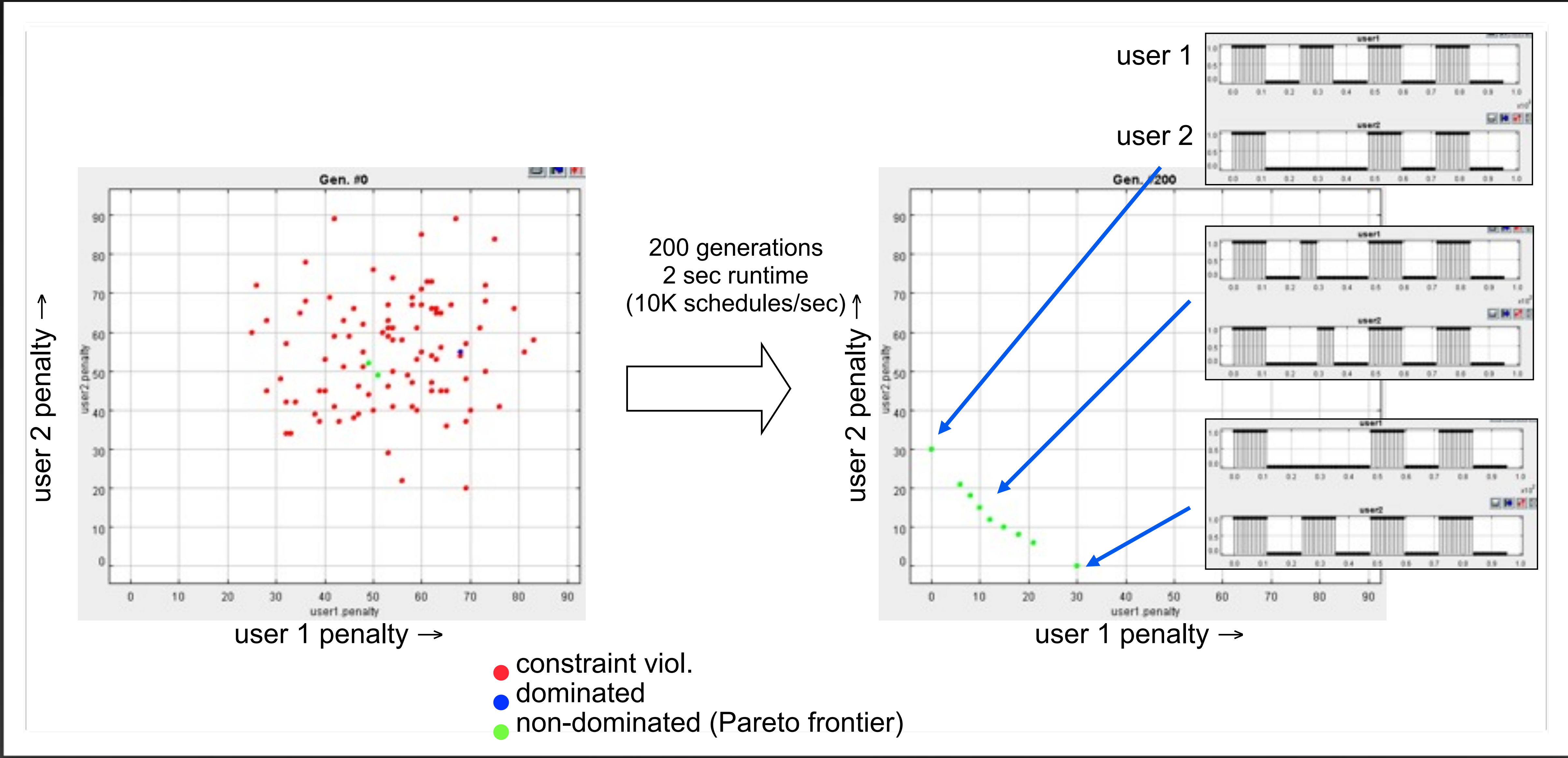
Schedule Model

- Time is quantized to arbitrary buckets
- Boundary conditions allow for contacts outside scheduling interval
  - needed to compute objectives/constraints that depend on contacts external to schedule
- Background of fixed contacts is supported
  - could be higher priority allocations
  - could be a frozen part of the schedule when rescheduling
- Decision variables based on user/viewperiod:



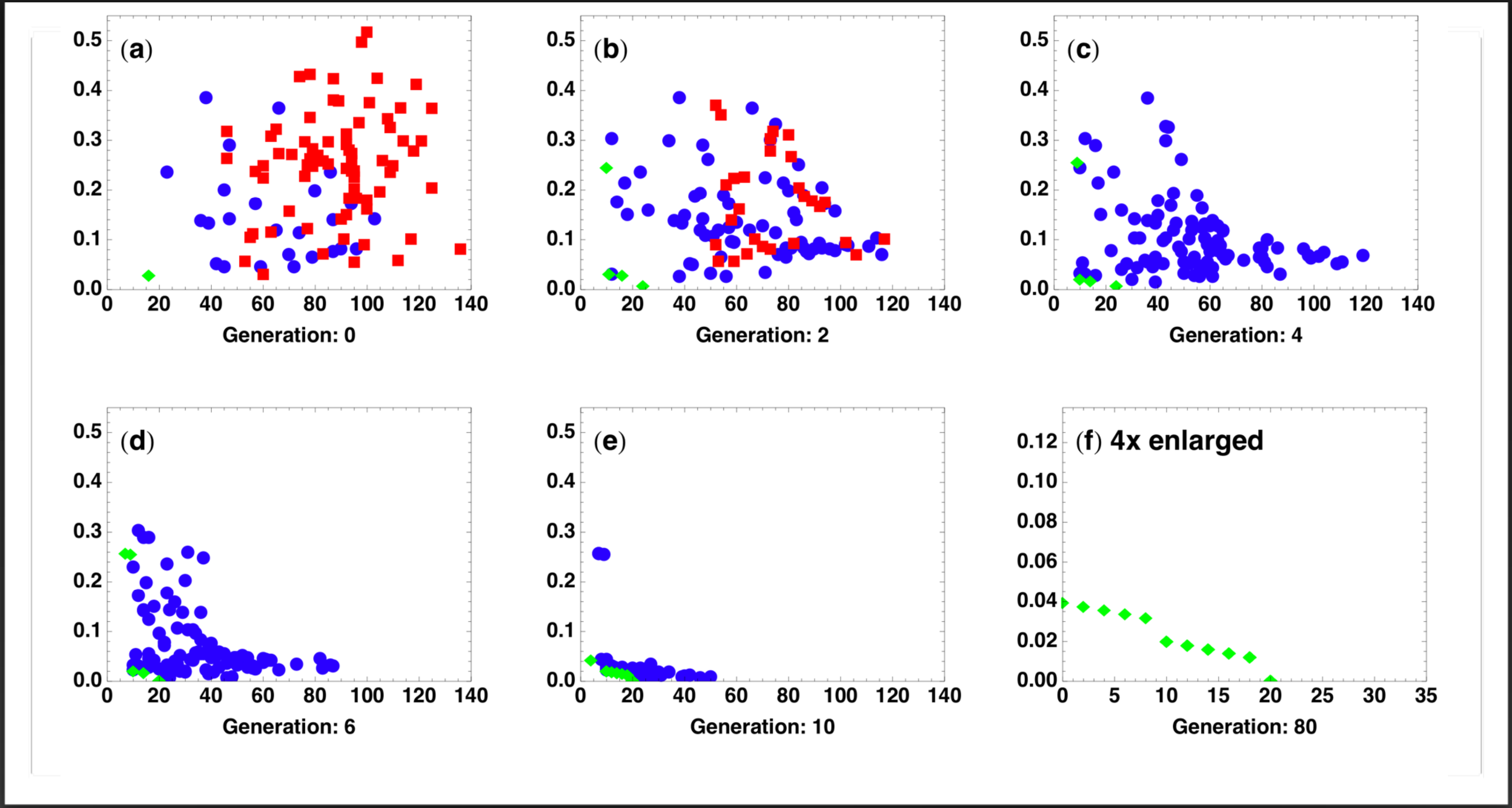
Simple Example

- Two (identical) users, 4 day schedule, 10 antenna "array", but only 5 available on 2nd day ("maintenance")
- Each user requires 5 antennas per contact, prefers 12hr contacts and gaps <18hr, requires at least 3hr contacts; there is one 12hr viewperiod/day
- Initial population randomized (diversity helps search, especially in more complex situations)



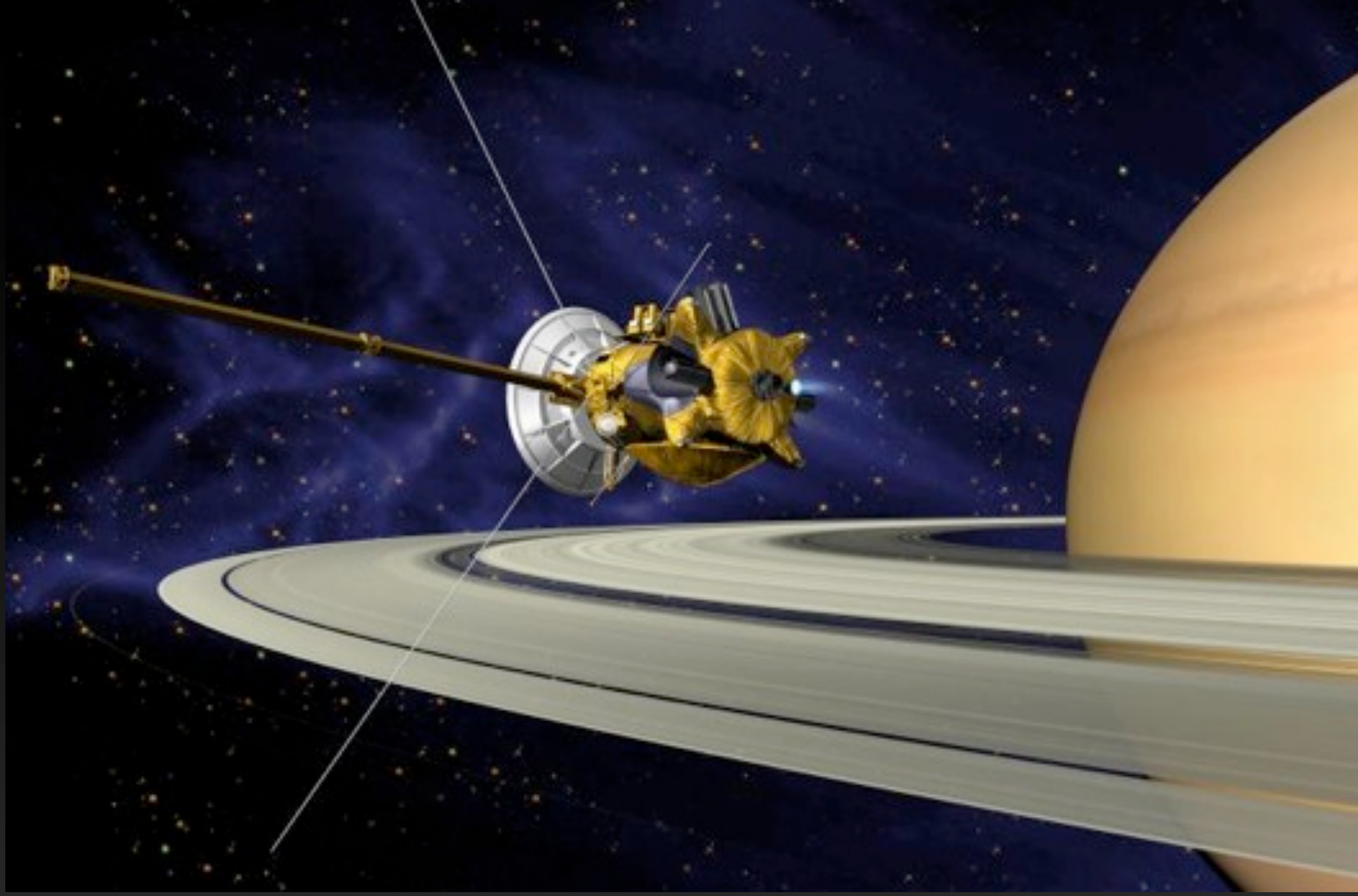
Example Problem: Risk as an Objective

- Two antennas A1, A2, with MTTF, MTTR
  - A1 fails more frequently
  - same availability MTTF/(MTTF+MTTR)
  - consider probability of failure as an objective to minimize



  - schedules on Pareto frontier utilize A2 only

Cassini



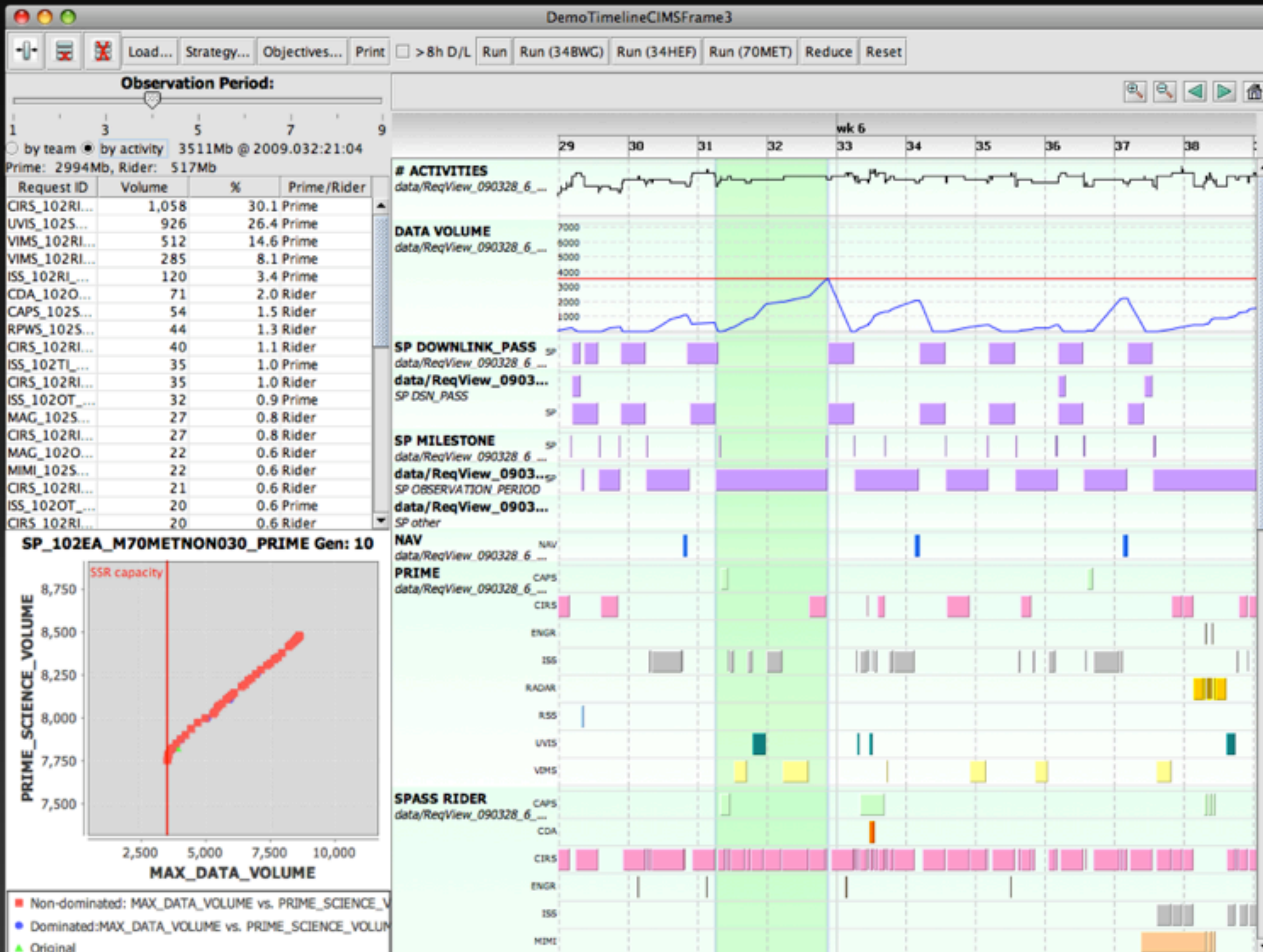
- Saturn orbiter + Titan lander
  - launched 1997
  - arrived at Saturn 2004
- Science instruments include 6 for optical and microwave remote sensing, and 6 for fields/particles/waves investigations
- Spectacular scientific success
  - 260 scientists from 17 countries participating
  - science objectives coordinated by 6 science discipline-oriented teams: Rings, Atmospheres, Titan, Icy Satellites, Magnetosphere, and Cross-Discipline (everything else)
- ~1 Gigabyte per day science data returned
- Prime mission completed; currently in first 2 year extension of prime mission: a second 2 year extension is expected

Cassini Science Planning Challenges

- Cassini has no scan platform: instruments and antenna are fixed to the spacecraft body
  - for most instruments, dumping data conflicts with taking data
  - data dumps have to be scheduled on the NASA Deep Space Network, in contention with other users
  - Cassini's Solid State Recorder (SSR) limits how much data can be stored
    - one 70m contact can empty the recorder, but one 34m contact cannot
  - managing the tradeoffs between collecting, storing, and dumping data, balancing among the different science teams, is a major effort!

Cassini — multi-objective science planning example

- 2 objectives: max science data volume, min SSR overcapacity



Concluding Comments

- **Strengths of multi-objective formulation:**
  - explicit and separate representation of each mission’s objectives, making it easy to consider tradeoffs
  - a population of solutions approximating the Pareto frontier, useful in scheduling selection and as a starting point when revising the schedule
- **Results to date are encouraging in terms of convergence to Pareto frontier, scalability, and diversity of coverage**
  - incorporating robustness as an explicit scheduling goal can be accomplished in several ways
- **Applicability:**
  - modifying an existing schedule quickly and effectively e.g. to respond to changed scheduling requirements or s/c emergencies
  - parallelizing the implementation to take advantage of grid or other distributed computing resources
  - applying to observatory scheduling where scientific and operational tradeosffs lend themselves to multi-objective formulation