Appears in Proceedings of Space Operations 2012, Stockholm, Sweden, June 2012.

A Generalized Timeline Representation, Services, and Interface for Automating Space Mission Operations

Steve A. Chien[1], Mark Johnston[2]
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109-8099

Jeremy Frank[3],
NASA Ames Research Center, Moffett Field, CA, USA, 94035

Mark Giuliano[4]
Space Telescope Science Institute, Baltimore, MD, USA, 21218

Alicia Kavelaars[5]
GMV USA, Rockville, MD, USA, 20850

Christoph Lenzen[6]
DLR / GSOC Oberpfaffenhofen, 82234 Wessling, Germany

Nicola Policella[7]
European Space Operations Center, European Space Agency, 64293 Darmstadt, Germany

Numerous automated and semi-automated planning & scheduling systems have been developed for space applications. Most of these systems are model-based in that they encode domain knowledge necessary to predict spacecraft state and resources based on initial conditions and a proposed activity plan. The spacecraft state and resources as often modeled as a series of timelines, with a timeline or set of timelines to represent a state or resource key in the operations of the spacecraft. In this paper, we first describe a basic timeline representation that can represent a set of state, resource, timing, and transition constraints. We describe a number of planning and scheduling systems designed for space applications (and in many cases deployed for use of ongoing missions) and describe how they do and do not map onto this timeline model.

I. Introduction

SPACECRAFT mission planning is an important application of automation in planning & scheduling of dynamic systems. Space mission planning often requires modeling of complex operations constraints including:

[1] Senior Research Scientist, Artificial Intelligence Group, MS 301-270, 4800 Oak Grove Drive, Pasadena, CA 91109-8099, Email: steve.chien@jpl.nasa.gov.
[2] Principal Member of Technical Staff, Mission Planning and Execution Section, MS 301-270, 4800 Oak Grove Drive, Pasadena, CA 91109-8099, Email: mark.d.johnston@jpl.nasa.gov.
[3] Lead, Planning and Scheduling Group, NASA Ames Research Center, Moffett Field, CA, USA, 94035, Email: Jeremy.d.frank@nasa.gov.
[4] Branch Chief, Advanced Planning and Scheduling Branch, Operations and Engineering Division, Space Telescope Science Institute, 3700 San Martin Drive, Baltimore MD, USA 21218, Email: giuliano@stsci.edu.
[5] Director of Business Development, GMV USA, 2400 Research Blvd. Suite 400, Rockville, MD 20850, Email: akavelaars@usa.gmv.com.
[6] Mission Planning System Engineer, *DLR / GSOC Oberpfaffenhofen, 82234 Wessling, Germany*, Email: Christoph.Lenzen@dlr.de
[7] Senior Research Engineer, Future Studies Section, European Space Agency - ESA/ESOC , Robert-Bosch-Strasse 5, 64293 Darmstadt, Germany Email: nicola.policella@esa.int.

instrument and subsystem timing and synchronization, thermal, power, data volume, geometry, visibility, and spacecraft pointing. Additionally, because spacecraft are expensive ($100M+ USD or Euro is not unusual), a planning system must be highly reliable and produce operations plans that do not endanger such a valuable asset. Finally, because of the complex nature of observation and/or science operations, priority and optimization are often involved either implicitly or explicitly.

Yet because of the vast potential for automated planning to improve space missions, a long line of planning systems have been applied in a vast range of space missions planning applications (some of which are listed below).

- The SPIKE system as achieved an over 30% increase in observation utilization of Hubble Space Telescope [Johnston et al. 1993]. In addition, SPIKE has been used in multiple orbital and ground based astronomical missions including FUSE [Calvani et al. 2004], Chandra, Subaru [Sasaki et al. 2004], and Spitzer [Kramer 2000]. SPIKE has been refactored for use in JWST long range planning [Giuliano et al, 2011].
- The TRANS system is a mission specific system that allows astronomical scientists to efficiently utilize the Hubble Space Telescope by planning how exposures schedule in abstract orbits [Curtis et al, 1998]. The system utilizes the same core timeline utilities as SPIKE.
- The DCAPS system achieved a 80% reduction in Command generation element of mission operations, 40% increase in science return for DATA-CHASER Shuttle payload, [Chien et al. 1999]
- The ASPEN-MAMM system resulted in a 6x reduction in mission planning effort, enabled dramatically more robust mission due to increased ability to perform what-if trades, 75% reduction in on-call staffing required for anomalies due to increased mission reliability for the Modified Antarctic Mapping Mission (MAMM), [Smith et al. 2002].
- The ASPEN-EO-1 R4 achieved an over $1M/year reduction in operations costs (~30% reduction overall) due to automation in uplink operations, reduction in downtime from ground station anomalies from 5 days to 6-8 hours, [Chien et al. 2005a, 2005b, EO-1, 2006]. ASPEN-EO-1 R5 subsequently improved these gain even further with a 30% increase in EO-1 weekly observations [Chien et al. 2010].
- Mars Exploration Rovers – MAPGEN [Bresina et al. 2005] mixed initiative planning system is used to plan operations for the Spirit and Opportunity rovers at Mars.
- The Flexplan system is currently in use for operations of the EPS Eumetsat, SMOS [Tejo et al 2007], the Lunar Reconnaissance Orbiter (LRO) [Chamoun et al. 2007, Barnoy et al. 2009] and Landsat Data Continuity (LDCM) missions [Barnoy et al. 2012].
- ASPEN-OE resulted in a reduction of 2 operations staff, increase of 26% in mission productivity, and 35% reduction in mission planning errors for the Orbital Express mission [Chouinard et al. 2008].
- MEXAR2 [Cesta et al. 2007] resulted in a 50% reduction in downlink data management planning for Mars Express and increased robustness due to ability to optimize and produce multi-day/week lookahead plans.
- RAXEM enables a 4-6 hours per week savings in uplink bundle planning for Mars Express and increased reliability in uplink operations [Cesta et al. 2008].
- SKeyP [Policella et al. 2009] applied to the SOHO Keyhole Period planning problem resulted in a reduction of operators workload (from different hours to few seconds) and increased robustness due to the ability to optimize visibility windows availability.
- The TerraSAR-X/TanDEM-X Mission Planning System, using GSOC's Pinta/Plato scheduling applications, supplies a fully automated command generation system with permanent order reception and order deadline of 6h before uplink. It supports up to 1000 orders per day [Maurer et al. 2010] and [Geyer et al. 2011].
- The MUSE [Johnston and Giuliano, 2011] system has been used to balance and optimize Cluster II science objectives in operations.

II. A Timeline-based approach to mission planning

Most of the above mission planning systems share a common timeline-based approach to mission planning. In this timeline –based approach, the overall system being planned (the spacecraft and relevant ground systems state) is represented by a set of timelines. Each timeline tracks the reported (past) or projected values of the spacecraft states and resources. These planning systems therefore are performing some form of modeling and simulation of the target system being planned.

This timeline based modeling is then used to plan the target system in a manual, semi-automated, or fully automated scheme. In such as scheme, an activity plan is generated and the resultant timeline modeling is used to verify the safety and validity of the plan. Depending on the exact plan generation scheme this search and modeling may be incremental (i.e. the system may plan activities, model the effects of the activities, and then plan further activities based on analyses of these effects.

In the remainder of this paper we discuss this common core of: (1) timeline modeling capability – the ability to model a range of states and resources commonly occurring in space mission operations and (2) the types of search interfaces supported by the timeline modeling capability, the ability to easily analyze the effects of scheduling activities at particular times and finding times to place activities that meet certain constraints (e.g. do not introduce state or resource violations).

III. Commonality between Timeline-based scheduling systems

Most timeline-based scheduling systems have the ability to model a basic core set of state, resource, and timing constraints. In this section we provide examples of a number of these common modeling functions, using examples from the modeling for the Earth Observing One (EO-1) mission operations as an example. EO-1 operations has a wide range of constraints that can be naturally represented in common planning & scheduling system modeling constructs (see below).

Activity overlap – instances of activities cannot overlap such as those that require an atomic resource. For example, two image sequence parent activities cannot overlap. This is represented by a simple atomic resource (a unit capacity resource) that an image sequence parent activity claims. If a second image activity overlaps it also claims this resource, exceeding the capacity.

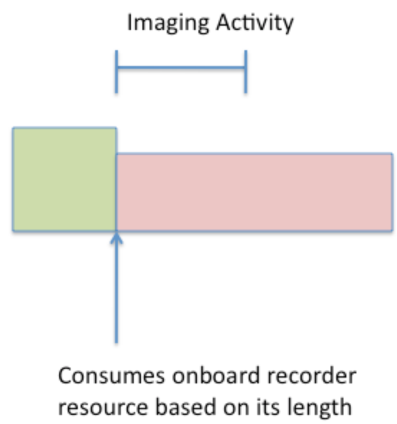


Figure 1: Activity consuming depletable resource

Integer capacity – depletable – this is an integer capacity resource reserved by one activity making a portion of the resource unavailable until it is freed by another activity. For example, EO-1 has a mass storage device primarily for science data. The storage device, called the WARP, has two capacity constraints. First, there is a limitation on the total number of files on the WARP at any given time. The file count is represented as a depletable resource the maximum capacity. When files are created they are counted against the file count resource. When files are deleted after downlink activities the resources are freed. Second, the total size of all of the files (summed) cannot exceed a different bound. This resource is consumed as data is written to a file on the recorder and released when files are deleted (after being downlinked). Usage of these resources can depend on activity parameters – for example the amount of data generated by an imaging sequence is dependent on how long the instrument is imaging as dictated by a function (a base amount plus a fixed rate times the image activity duration). This resource usage is shown in Figure 1.

Discrete states – there are numerous discrete state constraints. These both represent transition constraints and state constraints. For example, the solid state recorder has several states

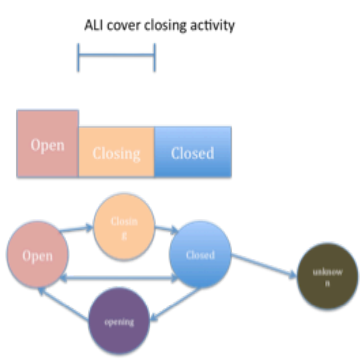


Figure 2: ALI cover state (partial)

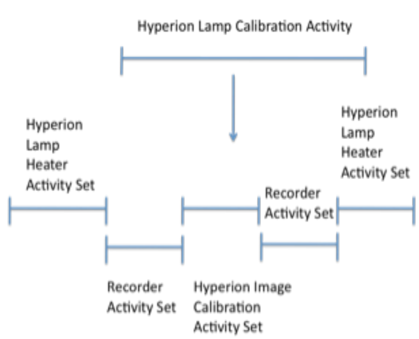


Figure 3: Hyperion Lamp Calibration Activity Decomposition

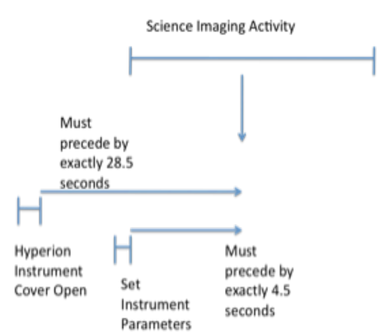


Figure 4: Temporal Constraints in Science Imaging

(record, playback, idle, standby,…). Furthermore, there may only be a specific subset of legal transitions with activities to change the state. The only means for the WARP state to change from one state to another is via execution of an action with a state changer, As another example, the ALI instrument has a cover which has specific activities to change its state, and imaging activities require specific states (dark calibrations require closed state, science images require open state). Figure 2 shows some aspects of constraints on the ALI cover state.

Infinite states – often some aspect of spacecraft state must be mapped onto one or more integer or real values. Two examples of this might be the location of a rover in some surface coordinate system (e.g. lat/lon) or the pointing of a spacecraft in a celestial coordinate system (e.g. Right Ascension and Declination). These states may also have transition constraints or state constraints applied. A transition state might restrict the patch by which the pointing of a spacecraft changes (e.g. a constraint on the flipping of a spacecraft due to maneuver constraints). A state constraint might restrict the pointing of the spacecraft due to an instrument boresight requiring bright body avoidance such as the sun, moon, or other objects.

Decomposition – often a high level activity consists of several lower level activities. These are similar to Hierarchical Task Network planning decompositions. For example, and imaging sequence high level activity consists of a large number of lower level activities including ALI and Hyperion (HSI) prep activities and post activities. Figure 3 shows the first level of decomposition for a Hyperion Lamp Calibration activity set.

Temporal constraint – these are constraints on the relative timing or ordering of two related activities. For example, in an image sequence, the instrument parameters must be set 4.5 seconds before the image start time and the Hyperion instrument covers must be opened 28.5 seconds before the image start time. Most of these temporal constraints are enforced in the decompositions outlined above. These types of constraints might be enforced with some flexibility (e.g. open the instrument covers 28.5 to 32 seconds before the image start time).

Functional dependencies: some of these temporal relationships utilize dependencies upon timeline values or activity parameters. For example, the Hyperion and ALI warm-up times are dependent on the expected temperatures entering into the imaging activity. If the instruments are already warm from prior image sequences the warmup time can be shortened allowing images to be acquired closer together and preventing the instrument from overheating.

To summarize, many timeline-based systems support the following capabilities as a basic functionality:

a) linear, possibly grounded timepoints for events such as activity scheduled start times, etc.
b) finite states, infinite states, depletable and non-depletable resources
c) variable relative time constraints
d) functional parametric dependencies among usages and activity parameters

Additionally, timeline-based systems also generally offer a number of computational services that involve processing the constraints in the presence of a complete or partial schedule. These would typically include the following.

e) the ability to detect constraint violations in the above constraint types. This constraint checking typically requires computation linear in the number of activities & constraints)

   Finally the timeline-based system would often provide services to enable analysis of a schedule with respect to searching the space of possible schedules, relating to adding, deleting, moving, detailing/abstracting activities in the plan. Note that the means by which these services are provided and implemented can vary significantly (e.g. general constraint reasoning engines (e.g. CP, LP IP, or others), rules, or core functions provided to the developer or modeler). These services might typically include:

f) to place an activity (and separately or simultaneously propagate/model) and possibly
g) the ability to query if a specific placement of an activity will violate the constraints
h) the ability to query for valid time placements of an activity with constraints (e.g. what are the valid times for which placing this activity will not violate any of the above constraints)
i) the ability to escape into arbitrarily coded constraint models to check complex constraints and activities such as: maneuvers, power, mobility, thermal.
j) delete an activity and check the above constraints. Move an activity and check the above constraints. An if a system models hierarchy detail/abstract an activity and check the above constraints.

Many timeline-based systems also support the encoding of preferences or soft constraints. However the means by which these types of constraints are encoded is more varied. However, these capabilities would generally bias the plan generation to prefer more or less instances of certain activities, prefer certain timing relationships (either absolute or relative/spacing), preferences on resource usage (or non-usage), and other characteristics of the produced plan.

Table 1 shows a number of timeline-based scheduling systems used for space mission operations and their modeling and search capabilities.

System	States, infinite (b)	Resources (b)	Relative timing	Parametric (d) constraints	Search interfaces (e-j)
APSI	Finite Infinite (by means of parameters)	Yes, depletable and non-depletable	Supported	Yes	e,f,g,h,i,j
ASPEN	Finite, Infinite	Yes, unit, depletable, non-depletable, integral	Yes	Yes	e,f,g,h,i,j
EUROPA	Infinite	Yes	Yes	Yes	e,f,g,h,i,j
flexplan	Supported	Supported	Supported	Supported	All Supported
Mexar2	finite states	reusable resources. Cumulative resource, and binary resource	yes	no	yes
MUSE	Yes	Yes	No	No	e-j
Pinta/Plato	Yes	Yes	Yes	No	e), f), g), h), j)
SKeyP	finite states	reusable resources. Cumulative resource, and binary resource	yes	no	yes
SPIFE	Infinite	Yes	Yes	Yes	e,f,g,i,j
SPIKE	No	Yes	Yes	No	e-j

Table 1: Several timeline-based mission operations systems and their modeling and search capabilities

IV. More information on timeline-based systems

In this section we describe in slightly more detail a number of timeline-based scheduling systems for space mission operations. Specifically, we highlight the evolution of such systems as well as discuss their modeling and search capabilities and provide references for further details.

A. APSI

At the European Space Agency (ESA), in the area of planning and scheduling, the use of timeline-based planning and constraint satisfaction techniques has been exploited and validated initially in hard coded, mission specific tools, such as MEXAR2 [Cesta et al., 2007] and SKeyP [Policella et al., 2009], two decision support systems for respectively the scheduling of on-board memory dumping of Mars Express (MEX) and for the planning of the operations during so-called keyhole periods of SOHO.

Both systems are based on models that focus on the temporal evolution of key components and on the ability to capture relevant domain constraints. This modeling approach allows for reducing the problem to temporal functions representing the amount of data manipulated over time by these key components, such that the constraints given by the instruments filling rate, the on-board memory capacity, and the channel bandwidth are satisfied.

In MEXAR2 the memory dumping problem is formalized as a Max-Flow problem (and the flow network associated to) allowing to re-use Max-Flow algorithms to produce a schedule (for instance this was obtained by modifying the Edmonds-Karp algorithm for generating maximum flow)

In SKeyP the previous approach has been evolved in order to manage not only the synthesis of dumping operations but also the planning of payload activity during a period of limited visibility. Also in this case a Max Flow algorithm is then used as a core of the solving module.

The experience of MEXAR2 paved the way, with the Advanced Planning and Scheduling Initiative (APSI), for further AI prototyping applications development and validation in ESA. APSI was intended to be a mission-independent experimental software platform aimed to develop and validate new planning and scheduling concepts and associated algorithms. On one hand, APSI aimed at creating a software framework to improve the cost-effectiveness and flexibility of the development of planning support tools with a focus on the solving algorithms. On the other hand, APSI strove to bridge the gap between advanced AI planning and scheduling technology and the world of space mission planning.

The heart of APSI is the Timeline Representation Framework (TRF), [Cesta et al., 2009] consisting of a three-layer architecture: a time and parameters layer, a component layer, and a domain layer. The planning domain is modeled as a set of concurrent threads (the timelines) and the problem is to synthesize a set of decisions to obtain a desired behavior and to synchronize the threads. The three layers of the TRF structure are as follows:

1. A common lower level represents the information shared among the timelines, temporal information, and parameter information.
2. A middle level represents the extension point where the modeler plugs the components.
3. An upper level provides a unified, shared representation of the plan.

At the current stage the APSI framework supports:

- To represent
    - linear, possible grounded time-points
    - Interval and Static Interval
    - Finite states, infinite states (by means of parameters), depletable and non-depletable resources
    - functional parametric dependencies among usages and activity parameters
- The ability to detect conflict in the above constraints
- The ability to place an activity (and separately or simultaneously propagate/model)

- Also it is possible:To query, for temporal, parameter, sequence, resource and duration constraints, if a specific placement of an activity will violate the constraints
- To query, for temporal, parameter, sequence, resource and duration constraints, for valid time placements of an activity with constraints
- To escape into arbitrarily coded constraint models, for manoeuvres, power, mobility, thermal, etc.

The APSI framework has been used to develop different planning applications at ESA. The first, called MrSPOCK, was for the planning of Mars Express, jointly prepared by the science operations and the flight control teams [Cesta et al. 2011].

These two groups of human planners iteratively refine a plan containing all activities for the mission. The process starts at the Long Term Plan (LTP) level – three months of planning horizon – and is gradually refined to obtain fully instantiated activities at Short Term Plan (STP) level – one week of planning horizon. This process continuously leads to weekly STPs, which are then further refined every two days to produce final executable plans. The goal of MrSPOCK has been to develop a preplanning optimization tool for spacecraft operations planning for the generation of a pre-optimized skeleton LTP that will be then subject to cooperative science team / operation team refinement.

B. ASPEN

The Automated Scheduling and Planning ENvironment (ASPEN) is an application framework that is used to assemble automated planning applications [Fukunaga et al, 1977, Chien et al. 2000]. At the core of ASPEN is a timeline representation capability that includes state (finite and infinite) and resource (depletable and non-depletable) and constraint engine (e.g. simple temporal network, temporal constraint network, parameter constraint network) capabilities.

ASPEN was developed targeting space mission operations and has been deployed operationally for a number of such missions, most notably the Modified Antarctic Mapping Mission [Smith et al. 2003], the Earth Observing One Mission [Chien et al. 2005, Chien et al. 2010], and the Orbital Express Mission [Chouinard et al. 2008]. ASPEN has a corresponding embedded version CASPER (Continuous Activity Scheduling Planning Execution and Replanning) suitable for use onboard spacecraft. CASPER is flying onboard the EO-1 spacecraft and flew onboard the 3CS mission. ASPEN and CASPER have also been used in a wide range of non-space applications including on platform and shore/ground off platform usage in aerial, rover, marine surface and submersible, ground communication station automation, and fixed sensor network control.

ASPEN supports a wide range of timeline types including the ability to naturally interface more complex timeline types via "generalized timelines" [Knight et al. 2001]. The basic timeline and constraint types are supported by modeling in a text and XML based "ASPEN modeling language" which is an activity-based modeling language. However, more expressive modeling often requires utilization of specialized code linked in to ASPEN. ASPEN also includes a range of native search capabilities [Rabideau et al. 1999] as well as a native optimization/preference capability [Rabideau et al. 2000]. However to achieve top performance often domain specific search control (e.g. heuristics) or domain-specific search strategies are sometimes used.

C. Extensible Universal Remote Operations Planning Architecture (EUROPA)

As a complete Planning & Scheduling platform, EUROPA offers capabilities in 3 key areas of problem solving:

1. Representation: EUROPA allows a rich representation for actions, states, resources and constraints that allows concise declarative descriptions of problem domains and powerful expressions of plan structure. This representation is supported with a high-level object-oriented modeling language for describing problem domains and data structures for instantiating and manipulating problem instances.
2. Reasoning: Algorithms are provided which exploit the formal structure of problem representation to enforce domain rules and propagate consequences as updates are made to the problem state. These algorithms are based on logical inference and constraint-processing. Specialized techniques for reasoning about temporal quantities and relations included in EUROPA are particularly useful to deal with real-life problem domains. Other specialized algorithms handle timelines and resources of different types, which are the focus of this paper.
3. Search: Problem solving in EUROPA requires search. Effective problem solving typically requires heuristics to make search tractable and to find good solutions. EUROPA provides a framework for integrating heuristics into a basic search algorithm and for developing new search algorithms.

EUROPA's main input modeling language is New Domain Definition Language (NDDL) (pronounced 'noodle'), a domain description language for constraint-based planning and scheduling problems. NDDL can describe a number of concepts based on Variables and Constraints. The NDDL representation includes state and activity descriptions, as is common in planners using traditional modeling languages like the Planning Domain Definition Language (PDDL) [Gerevini et al. 2009; Hoffmann and Edelkamp 2005]. EUROPA state variables are called timelines, and the values of timelines are sequences of states. States are temporally extended predicates, and consist of a proposition and a list of parameters, which by default includes the start, end and duration times. Timelines are totally ordered sequences of states; hence, a timeline can be in only one state at any instant.

EUROPA also permits the declaration of resources, including reusable (e.g. a machine used for a task, then freed when the task is over) and renewable resources (e.g. energy or memory, wherein separate tasks impact resource availability differently). EUROPA can also use this resource mechanism to facilitate both passive constraint violation checking for 'classical' planning problems, and 'active solving' by using the resource mechanism to detect flaws in a plan.

The final component of NDDL model is a set of compatibilities that govern the legal arrangements of states on, and across, timelines. These compatibilities are logical implications asserting that if a time- line is in a state, then other timelines must be in one of a set of compatible states. Compatibilities can incorporate explicit constraints on the parameters of the states. EUROPA provides a library of such constraints, and this library can be extended if new constraints are needed.

An overview of the EUROPA framework is provided in [Frank and Jonsson, 2003]. Modern versions of EUROPA integrate the grounded and flexible time perspectives at a basic level [Morris et al., 2011a,b].

MAPGEN, which utilizes EUROPA, was (and is still today) used by the Tactical Activity Planners to perform science planning for the Mars Exploraiton Rovers [Bresina et al., 2005]. For the Mars Exploration Rover mission, an early version of EUROPA was interfaced to plan viewing and editing using the APGEN system to form MAPGEN. This combined a grounded time representation with the flexible time of EUROPA, supporting passive constraint checking as well as active fixing of flaws and insertion of activities such as CPU usage. EUROPA was integrated with external reasoning systems to calculate energy usage.

For the International Space Station, EUROPA was employed to build a tool called the Solar Array Constraints Engine (SACE) [Reddy et al. 2011] to assist flight controllers in planning Solar Array activities in the presence of several different types of constraints. Unlike the MAPGEN application, SACE planning is heavily slanted towards optimization; as a result, EUROPA was used to build an optimizing planner. EUROPA was customized to handle a novel type of constraint based on tables describing the quality of a proposed orientation and mode of the solar arrays in a given configuration. Another interesting feature of the SACE application of EUROPA is the reuse of a generic timeline for a single solar array in the model; there are eight arrays onboard ISS whose ideal behavior is solely a function of the tables that apply to that specific array.

Finally, it is worth noting that the EUROPA framework was also the inspiration for a robotic control architecture employing planning and scheduling in the inner control loop. The T-REX architecture employed the NDDL language and a planner based on the EUROPA infrastructure to control an autonomous underwater vehicle [McGann et al. 2008].

D. flexplan

Flexplan is a space mission planning system developed by GMV. The *flexplan* system is structured upon four major elements that are used for the planning purposes. These are:

- Resources: Model mission resources, including space resource (spacecraft, sensors, antenna), ground resources (ground stations, antenna), and human resources (manpower, operator shifts); offering both "integer capacity" and analytical modeling.
- Events: Model input activities, can be external/input based (flight dynamics events, geometry events, instrument service requests) or internal/planning system based (generate a list of imaging events based on a defined trigger)
- Tasks: Model timeline events, are organized in hierarchical sequences for decomposition from a high level sequence to the lowest executable command level ("Task decomposition")
- Rules: Model schedule based upon logical soft algorithms, applied not only to scheduling (Rules applied to the Events and Resources to generate the schedule Tasks) but also to plan optimization, constraint detection and constraint resolution. Rules generation is accomplished by mission specialists without the need for complete software recompile. Rules also model the constraints of the system, including resources, scheduling parameters out of limit, timeline, and other mission specific constraints, as defined by mission specialist to ensure the cheduling process output is an optimized mission schedule.

flexplan's framework high configurability to model Resources, Events, Tasks and Rules allow to setup the system according to the mission specific requirements and constraints. This activity grants the operations team direct control to implement the mission plan. This allows preparing both the team and the scheduling software to support the mission, as well as minimizes the need to modify the system code in order to create, maintain or extend the mission based upon new mission goals or events.

The *flexplan* architecture is highly modular, based on a three-tier, client-server architecture that can be deployed in a physical or virtualized environment. Each module has a purpose in a simple stepped approach to schedule generation. It allows for a well organized process to configure the mission, review the inputting events, generate the baseline schedule and detect and resolve the conflicts in the schedule to generate the final conflict free executable schedule. The *flexplan* server is highly scalable and is capable of supporting a large number of clients simultaneously, with the only limiting factor being the deployment hardware sizing.

The *flexplan* system also allows embedding add-on modules in the schedule generation process for specialized calculations or operations. For example, a Radio Frequency Interference calculation module can be integrated to generate constraints based upon RF interference between missions; a Load Builder module can be integrated to generate and export the list of executable CCSDS commands to be sent to the spacecraft via the real time system.

In terms of timeline representation, the following aspects are supported in ***flexplan*** as follows:

- Platforms: Each spacecraft, sensor suite, ground site, or operations center in the mission is depicted in the timeline as a "platform" with associated command sets, and is displayed and color coded separately in the timeline. The timeline can be filtered by platform as needed. It is possible to implement Task or "Activity Overlap" in the same Platform or across Platforms when the combined overall resource usage exceeds its capacity. When Tasks overlap a schedule conflict is raised.
- Linear representations: Each Task in the schedule is depicted with a start time and a stop time, as well as an exclusion time windows (interval before and after the Task where no other task can be scheduled, for example if a certain time is required to move the ground antenna to a default step location), if applicable. Tasks can also be instantaneous tasks of no duration (for example turning on an on-board system).
- Resources: Permanent and temporary ("Integer capacity") depletable resources can be defined. Resource values (including "discrete states") are stored in integer format. Resource consumption versus availability timelines can be selected and visualized in synchronized time plots underneath the schedule timeline. Resources can be associated with Tasks (when task A is executed, resource B is affected).
- Constraints: Constraints are defined by either of three categories: Temporal constraints, resource constraints, and limit violations.
    - Temporal constraints (equivalent to variable relative time constraints): Constraints based upon activity overlap execution times. For example Task A cannot be executed until Task B is executed; or Task

A cannot be executed during time interval [B, C], where time interval [B, C] is determined based upon an initial variable condition determined by an input Event.
 - Resource constraints (equivalent to functional parametric dependencies among usages and activity parameters): For example Resource A cannot exceed availability usage.
 - Limit violations: Constraints that specify command parameter values beyond which a task cannot be scheduled. For example if commanded temperature value is not between an operationally defined minimum-maximum interval.
- Task (equivalent to Activity) Insertion: Tasks can be manually inserted in *flexplan* anytime during the schedule generation process. After a Task is inserted, the schedule conflict detection step can be run to detect if the new task violates the schedule constraints, and if this is the case, run conflict resolution to obtain an alternative time interval to generate a new conflict free schedule.
- Task (equivalent to Activity) Move or Deletion: Activities can be moved or deleted from the schedule in *flexplan* anytime based upon input Events or manual insertion. When a task is moved, the conflict detection and resolution steps will be run to generate a conflict free schedule.
- Conflict Detection: One of the steps in *flexplan* schedule generation process is conflict detection. In this step, the conflict detection rules are applied to the baseline schedule, and all conflicts detected are flagged for operator review.
- Conflict Resolution: Typically an operator led process. However, *flexplan* offers the capability of generating a conflict free schedule in a fully automated process. Conflict resolution is performed immediately after conflict detection to generate, based upon conflict resolution rules, a conflict free schedule.

In certain missions, advanced, schedule-optimization algorithms are integrated into *flexplan* to meet more complex scheduling requirements. In these cases, constraint rules are combined with scored rules. As before, constraints are schedule necessities that must be resolved prior to execution (for example Task B cannot be executed unless Task A is complete). Scoring rules allow for an automated process of resolving constraints by defined moves to achieve the highest score (also known as soft constraints). The schedules generated will be conflict free and best support of the mission according to the scored rules.

For certain missions, *flexplan* is in the process of implementing position based scheduling into the timeline framework. When the mission reaches a certain geometrical condition, as determined by the GPS or other attitude determination system on-board, a set of tasks will be executed. The exact execution time for each task is fed back into the scheduling timeline based upon flight dynamics transformation algorithms.

An extensive amount of development has been devoted to enhancing the operability of *flexplan* based upon a human centered design. A pictorial timeline representation of the schedule is available, which allows the user to decompose the scheduled items to its lowest level. Additionally, *flexplan*'s interface allows the user to perform near-real-time procedure scheduling based upon last minute events and multiple visual cue functions, as well as to receive feedback on the schedule execution state.

E. MUSE

MUSE (Multi-User Scheduling Environment; [Johnston and Giuliano 2009] is a system focused on multi-objective optimization using evolutionary algorithms, and as such employs timelines in a different way from most of the other systems discussed here. Although adapted to a number of different domains including James Webb Space Telescope [Giuliano and Johnston 2008], it has also been applied to the Cluster II constellation of four spacecraft [Johnston and Giuliano 2011], in which schedules are sought which optimize science return while considering robustness, priority, flexibility, and other factors. Within the Cluster II adaptation of MUSE, timelines are used to represent grounded activity placements and their constraints, as well as resource availability, limits, and consumption. The main point of a multi-objective approach is to develop a representation of the Pareto or trade-off surface for two or more objectives of interest. Evolutionary algorithms represent one powerful technique for this purpose, and maintain a population of candidate solutions that are evolved towards an approximation of the Pareto frontier.

In MUSE, timelines play a central role in the definition of the different objective functions that need to be evaluated for each candidate schedule. Candidate schedules are represented as set of timelines, based on decision variable assignments. These timelines can be accessed by objective function calculation methods to score each candidate schedule. The scores are used in subsequent evolution steps to approach and sample the Pareto frontier. Examples of the kinds of timeline information supported by the query interface for objectives would be:

- expected resource consumption by high-priority or inflexible non-Cluster activities, for which Cluster would be unlikely to compete successfully should it attempt to schedule them (schedule robustness objective)

- timelines for the different categories and priorities of science activities, queried for degree of similarity to ideal distributions
- activity timelines for time separation metrics, to avoid schedules with too many large gaps or activities clumped too closely together

In addition to their role in fast evaluation of objective functions, timelines also used to post-process and optimize schedules at the conclusion of the evolutionary phase, exploiting additional flexibilities to further improve schedules as finally presented to users.

F. Pinta/Plato (core of TerraSAR-X/TanDEM-X mission planning)

The Pinta/Plato scheduling software has been developed by the German Space Operations Center for many years [GSOC-DLR 2012a]. It has been used for the missions SRTM (DEM generation using radar on the shuttle), Bird (infrared earth observation), Champ (earth's gravity and magnetic fields observation), Grace 1&2 (continuous earth gravitation measurement), TerraSAR-X/TanDEM-X (high resolution X-Band radar earth observation and DEM generation). It is based on a flexible modeling language, whose main intention is to allow easy mapping of the real world into the model. For this purpose, we support

- a hierarchical structure, where tasks may be given timeline entries and groups may contain other groups and tasks
- resources, each resource defining a profile over time, which represents the temporal evolution of a certain aspect of the system
- resource limits, which specify a calculation rule for the resource profile, e.g. the capacity of a battery: when the battery is full, charging will no longer take place.
  Remark: this is not a constraint, since it can't imply a conflict
- constraints:
    - duration limits of a task
    - time-dependencies in between tasks A and B:
      Minimum distance in between start/end of A before start/end of B
      Remark: a negative minimum distance yields a maximum distance in between B and A
    - time-dependencies of task A with respect to absolute times
      Best Practice: define a timeline horizon task with a timeline entry covering the whole timeline horizon. You can define a time dependency on this task and task A, which represents a time-dependency of A with respect to absolute times.
    - minimum/maximum number of child elements (tasks and groups) to schedule for a group
    - resource comparison
        - define an interval relative to the start- and end-time of the task's timeline entry (infinite offsets are allowed)
        - checks a resource's value during the specified interval
    - resource modification
        - define an interval relative to the start- and end-time of the task's timeline entry (infinite offsets are allowed)
        - modifies a resource's value during the specified interval
    - resource bounds
      Best Practice: define a timeline horizon task with a timeline entry covering the whole timeline horizon. On this task you can define a resource comparison, which defines a 'global' bound on the resource's profile
    - suitability
      roughly: a timeline entry's value depends on a resource's value during the timeline entry

Note that no distinction in between depletable, non-depletable or renewable resources needs to be defined. The ways the tasks' timeline entries interact with the resources are defined via resource comparisons and resource modifications. A detailed description of the modeling language can be found at [GSOC-DLR, 2012b].

The Plato library is the scheduling engine, which supports

- configuring heuristic algorithms from predefined generic algorithm-snippets
- a plug in mechanism for specialized algorithms, which may use all features of the Plato library
- combining generic algorithm snippets with project specific algorithms

- a plug in mechanism for further special code, which may break off a sub-path of the algorithm (-> backtracking on outer algorithm), e.g. in case a thermal model shows a conflict, which can't be modeled by Pinta/Plato's modeling language
- coming soon: a pseudo rational data type, which removes rounding issues
- search for non-conflicting timeline entries
- calculating conflict traces, i.e. for a given timeline entry, all conflicting timeline entries are calculated, together with the constraints, which imply the conflict
- search for best timeline entry according to soft constraints (-> suitability)
- …

The graphical front-end of the scheduling suite is called Pinta, which is a windows application. It allows manual interaction with the model and the timeline in a very convenient and illustrative manner. Constraints may also be defined in a semi-automated manner, which means that you can specify inside an XML file, what constraints to define for which tasks, task-pairs resp. task/resource-pairs. The matching tasks, task-pairs resp. task/resource-pairs are determined depending on parameter values and object names. Additionally you have full access to the Plato library; in particular you can execute pre-configured algorithms from Pinta. The plugin mechanism of Pinta allows including project specific code, such as a timeline exporter, which generates the commands from the timeline or a file importer which allows reading files which obey a project specific file format.

GSOC's mission planning tool-suite is completed by two further applications:

TimOnWeb: it allows distributing the timeline via web browser

SCOTA: Current generation of mission analysis and event calculation software with advanced visualization support. It supports the most commonly used orbit propagator models (SGP4, SGP8, etc) for different mission requirements. Depending on the trajectory and attitude of the considered spacecraft, a number of events can be calculated and visualized in 2D and/or 3D graphics. Examples for these events are day/night periods, satellite visibility and target opportunities. Furthermore, the software also supports spacecraft groundtrack real-time plotting.

G. Scheduling and Planning Interface for Exploration (SPIFe)

The SPIFe user interface is designed to be a highly adaptable and user-customizable framework for viewing and manipulating plan and schedule data [Aghevli et al. 2007, 2011, Marquez et al. 2010]. In order to achieve this, SPIFe employs a composable, plug-in architecture based on the open source Eclipse Rich Client Platform (RCP). Eclipse provides a robust plug-in framework, and the RCP provides many fundamental user interface components, such a tabbed "workbench" that allows users to manipulate views and editors to display the information most relevant to the task at hand. The following sections describe a number of SPIFe views and editors that can be combined (or omitted) depending on the needs of a particular planning application.

One of the central components of the SPIFe framework, the timeline provides a traditional time-based representation of a plan. Activities appear as bars that vary in width according to when they're scheduled. Timeline rows are highly configurable: which rows are displayed, their ordering, bar figure look and feel, and row criteria (which determine whether a given activity appears on a given row) can all be modified in a descriptor file for a given application. All activities can be edited directly via drag-and-drop, and the timeline also provides several feature such as multiple selection, feedback during editing operations, and full support for multiple levels of Undo and Redo to allow users to freely explore multiple solutions. The goal of the SPIFe timeline is to capitalize on user familiarity with common visual editing paradigms where possible (e.g. manipulating figures in drawing tools like Visio or Powerpoint) in order to remain approachable by non-experts.

In addition to the timeline editor, SPIFe provides a tabular representation of the activities and groups in the plan. The Table Editor is useful for displaying a large number of activities, and is especially useful for plans that are sparsely populated (few events over long periods of time) where a timeline display would be mostly empty for a given time range. The Table Editor can be configured with columns representing each piece of activity metadata, including basic start time and duration information as well as details of resource requirements or per-activity resource consumption predicts.

One of the fundamental design principles of the SPIFe toolkit is that the user's hand should not be forced by any integrated automated planning system. As a result, much of the feedback from the native constraint and resource engine as well as feedback from external systems is presented in a view called the Plan Advisor. The concept behind the Plan Advisor is that the human is in control of the plan, but he or she may selectively invoke help from automated systems. In most cases feedback is presented to the user in realtime after each plan edit. If a violation is determined to be fix- able, either by native code or an external engine, users are presented with a context menu containing common fixes. If more extensive reasoning or search is required, users can invoke the capabilities of external systems via "Fix Violations" commands which are also invoked from the Advisor.

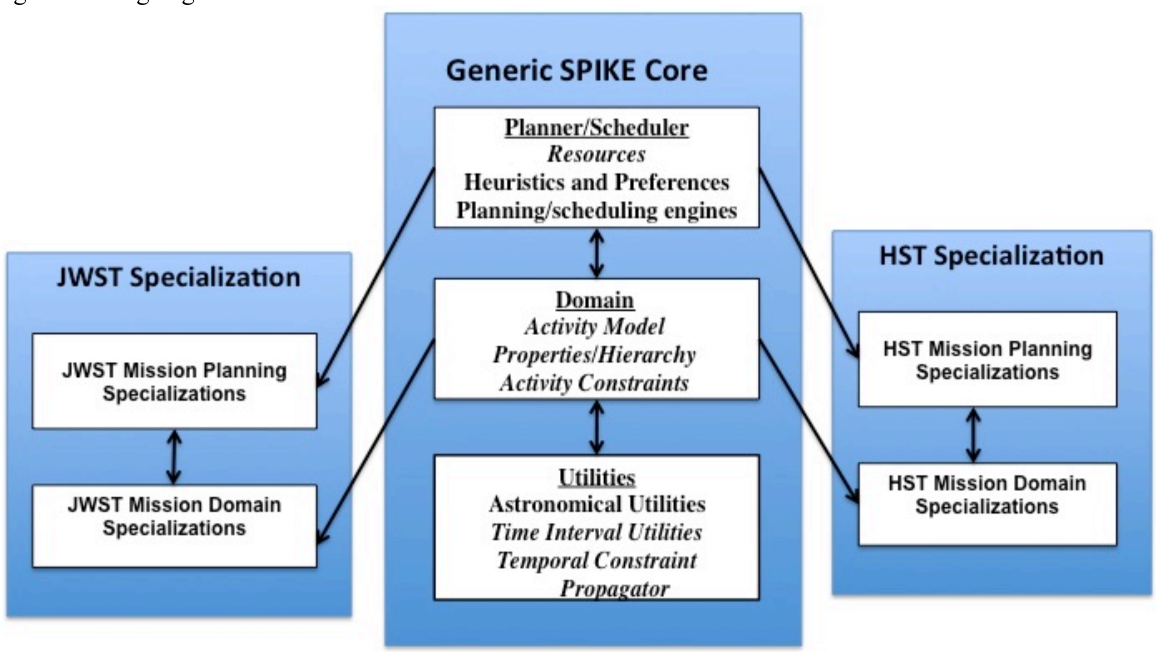
In many cases violations are deemed acceptable, either due to a one-time exception, or more commonly an error or omission with the model or constraints themselves. In these cases, users can waive the violation and provide rationale. These waivers and rationale are persisted with plan data so an audit history is always preserved.

SPIFe has the capability to display resource usage effects that are derived from the schedule and visualize resource modeling information of varying kinds from coarse approximations, encoded as Javascript formulas, to extremely high resolution simulation data. It also supports a multitude of higher fidelity simulation engines to display things like power, geometry (e.g. position of sun relative to spacecraft), or data usage. The results of external modeling tools are transferred seamlessly to the SPIFe toolkit for display in the context of the planning session: alongside the timeline, in columns in the table editor, in fields in an inspector pane associated with each activity, and in the Plan Advisor if necessary. This allows users to immediately see the effect of plan changes within the same context and debug issues that potentially result from them.

H. SPIKE

SPIKE [Johnston and Miller 1994] is a planning and scheduling tool kit that was created for use on the Hubble Space Telescope and has been used for multiple orbital and ground based astronomical missions including FUSE [Calvani et al. 2004], Chandra, Subaru [Sasaki et al. 2004], and Spitzer [Kramer 2000]. SPIKE is used operationally for HST long range planning and has been refactored for JWST long range planning [Giuliano et al, 2011]. Multi-objective aspects of the MUSE system (see elsewhere in this paper) have been integrated with the generic SPIKE core and are being integrated with HST and JWST long range planning capabilities.

As shown in below Spike has a three-layer architecture including utilities, domain model, and planning/scheduling. Not shown on the figure is a GUI controller that integrates with the planner/scheduler. Functionalities associated with timelines are shown in *italics* below. Although timeline functionality is distributed across all three layers of the architecture, most functionality is supported by the combination of the domain model and the temporal constraint propagator. To create a SPIKE application protocol methods in the domain and the planner/scheduler modules are specialized to implement mission specific functionality. The design pattern in the domain model engages the lower level temporal constraint propagator. The architecture supports placing different planning/scheduling engines over a domain model for a mission.



The utilities temporal constraint propagator implements most of the timeline functionality for SPIKE. The propagator allows the creation of *chronikas*, which propagate constraints over an interval of integer-based time. Piecewise Constant Functions (PCFs) are used to represent suitability of tasks over the chronika interval. The standard encoding of PCFs has values over the domain 0-1 (inclusive) where a value of zero indicates unsuitable intervals and a value greater than zero represents the quality of the time. Chronikas support the following actions:

- The addition/removal of tasks with a fixed or variable duration

- The addition/removal of absolute constraints that impact the suitability of single tasks.
  - Absolute constraints can have multiple disjunctive intervals.
    - For example, a sun avoidance constraint may be suitable from January-April, unsuitable from May-June, and again suitable from July-December.
- The addition/removal of relative constraints that link multiple observations.
  - Tasks to task overhead for slews and or instrument reconfigurations
  - Relative constraints can support unary or disjunctive intervals linking multiple tasks.
- Alternative constraint implementations are available and can be coded that trade off run time constraint propagation with the time to add/remove constraints.
  - For example, path consistency for constraints is desirable if the constraint structure is static but may not be worthwhile if the constraint structure is dynamic.
- Observations with space telescopes often allow users to specify the roll of the telescope along the bore sight. Users may prefer certain rolls to take advantage of instrument asymmetries and/or bad pixels in an instrument.
  - Users can specify a range of rolls for a given task and can link pairs of observations to have the same or offset rolls.
  - Typically thermal constraints limit the legal roll for a telescope at any given time to a small subset of the full 360-degree roll.
  - The ability to reason about roll links has been integrated with the domain model and the temporal constraint propagator.
- The propagator computes the pair wise transitive closure of all constraints and provides interfaces for the current suitability for tasks
  - The propagator tracks input and will automatically recalculate suitability windows on demand.
  - The propagation system can compute false positives if disjunctive constraints are present.
- The propagator supports the interface e-j in Section III.
- The propagator has been used by other HST observation preparation systems and is also used to compute individual constraints for HST SPIKE.

The constraint propagator deals with temporal constraints and only has a weak model of astronomical observations. The SPIKE domain model module provides support for more detailed knowledge about astronomical observations and provides built-in support for the following hierarchy of objects and object properties:

- *Observations:* An observation represents a group of one or more telescope exposures that are to be executed in a contiguous manner
  - Exposures are grouped in observations based on science and engineering needs.
  - Observations have a flag indicating whether or not they are ready for planning according to some mission specific query.
- *Proposals*: A proposal is a set of observations from an astronomer that implements a specific science goal.
- *Targets*: An observation has a set of targets.
- *Constraints*: A visit has a set of absolute and relative temporal constraints
  - The system provides support for computing constraints, caching constraints, and/or reading constraints from disk (e.g. DB).
  - Domain constraints map flexibly to constraints in the temporal constraint propagator.
  - The system supports execution time constraints
  - Constraints can be marked to override other constraints.
    - For example, execution time constraints override all others.
- *Errors:* Processing errors and anomalies are tracked in all objects.

Although SPIKE does not have built in support for parametric constraint and states the system that provides HST SPIKE input does support these features. For HST the TRANS system [Curtis et al, 1998] models details of observations and provides SPIKE with a high level model of observations suitable for long range planning. TRANS supports simulation states, constraints between properties, and uses the same temporal constraint propagator as SPIKE to reason about activity placements [Curtis and Giuliano, 1999].

Due to the evolution of the SPIKE system it supports resources at the level of the planner/scheduler module. As SPIKE evolved different operations concepts emerged each requiring a different planning/scheduling implementation. An implementation based on a Constraint Satisfaction Problem (CSP) supports traditional scheduling to precise times. An implementation based on heuristic repair supports Long Range Planning to least commitment windows that are typically weeks in duration [Giuliano 1998]. These alternate implementations have very different needs with respect to resource models. The CSP implementation handles typical renewable and non-renewable resources found in planning applications. See [Giuliano et al, 2007] for a case where tasks can both consume and restore a resource depending on when it is scheduled. In contrast the LRP implementation distributes resource usage across the entire plan window for an observation [Ferdous et al, 2006, Giuliano et al, 2011]. The two planning engines also treat resource violations in a different manner. The LRP engine treats resources as a preference while the CSP engine treats resource violations on par with other constraint violations. A final observation is that while planning preferences are modeled at the scheduling/planning layer the domain model and timeline need to support the computation of preference data.

V. Looking to the future

This paper was organized with the intention of (1) documenting the tremendous progress that has been made in the use of automated and semi-automated techniques for scheduling of space mission operations and (2) highlighting the significant shared approach in timeline-based modeling of spacecraft state and resources in these systems. As the above descriptions indicate, while there are significant differentiating factors in the scheduling systems represented, there is even greater commonality in modeling and search interfaces.

Why should we care about this commonality? Because where there is commonality and shared interfaces there is significant potential for re-use, leveraging, and synergy at the algorithms and software level. The next step might be to more formally define the search and modeling services and to investigate whether it might be possible to share: display and editing software, file formats (XML), analysis software, and even search algorithms and modeling capabilities. Well-defined interfaces might enable a search technique (i.e. software) developed for one application and timeline reasoning package to be applied to another timeline reasoning package with minimal re-work. Such interfaces might also enable easier comparison of approaches and results on benchmark problems.

VI. Conclusions

We have described a common class of mission planning systems designed for space mission operations. Key to all of these systems is the concept of a “timeline” in which the evolving state and resources of the target system is modeled in order to track the effects of past executed and future planned activities on the system. All of these systems share a common core of modeling capabilities including some ability to model finite and infinite domain states, depletable and non-depletable resources, temporal constraints, and varying elements of dependency functions. All of these systems also share a common core of timeline analysis capabilities to enable detection of constraint violations and to potentially make changes to the schedule by addition, removal, or moving activities in the schedule. However, this set of systems while sharing a common set of general capabilities, implements these capabilities using a wide range of technologies (constraint programming, integer programming, direct code implementation, temporal constraint networks, rules, and others) and using a wide range of implementation options (e.g. rules, external libraries, custom coded algorithms).

Acknowledgments

Portions of this work were performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

Portions of this work were performed by GMV under contract with the National Aeronautics and Space Administration at the Goddard Space Flight Center and the European Space Agency.

References

A. Aghevli, A. Bachmann, J. Bresina, K. Greene, B. Kanefsky, J. Kurien, M. McCurdy, G. Pyrzak, C. Ratterman, A. Vera, S. Wragg. Planning Applications for Three Mars Missions with Ensemble *5th International Workshop on Planning and Scheduling for Space.* Baltimore, MD, 2007.

A. Agveli and A. Bencomo and M. McCurdy. Scheduling and Planning Interface for Exploration (SPIFe). System Demonstration, International Conference on Automated Planning and Scheduling, 2011.

A. Barnoy, G. Garcia, “Ouroboros: Self-Driven Planning System for the Automation of Satellite Scheduling Operations”, Space Operations & Support Technical Committee (SOSTC), 2010.

A. Barnoy, J. Caplinger, F. Colmenero, J. Tejo, G. Garcia, “High- Impact Performance Processing Capabilities in Operational Mission Planning and Scheduling Systems”, SpaceOps 2010 - AIAA-2010-2238 4/25/2010

A. Barnoy, A. Kavelaars, F. Colmenero, J.A. Tejo, M. Pereda, "Evolution of a Flexible Mission Planning and Scheduling System for Complex Missions: *flexplan*" 6th Int. Workshop on Planning and Scheduling for Space (IWPSS), 2009.

A. Barnoy, G. Garcia, T. Beech, R. Saylor, J. Sager Cosgrove, S. Ritter, "Mission Planning and Scheduling System for NASA's Lunar Reconnaissance Mission", 13th Ground System Architectures Workshop (GSAW), 2009.

A. Barnoy, S. Gregory , G. Garcia, G. Greer, R. Shendock, J . Williams, V. Dulski, "Centralized Mission Planning and Scheduling System for the Landsat Data Continuity Mission" - SpaceOps 2012, pending NASA approval for publication.

J. Bresina, A. Jonsson, P. Morris, K. Rajan, "Activity Planning for the Mars Exploration Rovers," Proceedings International Conference on Automated Planning and Scheduling, 2005, Monterey, CA.

H. M. Calvani, A. F. Berman, W. P. Blair, J.R. Caplingera, M.N. Englanda, B.A. Roberts, R. Hawkins, N. Ferdous, T. Krueger, 2004. The Evolution of the FUSE Spike Long Range Planning System. In *Proceedings of the 4th International Workshop on Planning and Scheduling for Space*, Darmstadt, Germany, 25-31.

A. Cesta, Cortellessa, G., Fratini, S., Oddi, A. and Policella. N. An innovative product for space mission planning: an a posteriori evaluation. In Proceedings of International Conference on Automated Planning and Scheduling, 2007, Providence, RI.

A. Cesta, Cortellessa, G., Denis, M., Donati, A., Fratini, S., Oddi, A., Policella, N., Rabenau, E. and Schulster, J. RAXEM - Supporting Command Uplink in Mars Express. In i-SAIRAS-08, Proceedings of the 9th International Symposium in Artificial Intelligence, Robotics and Automation in Space, Studio City, CA, February, 2008.

A. Cesta, G. Cortellessa, S. Fratini, A. Oddi, "Developing an End-to-End Planning Application from a Timeline Representation Framework." In IAAI-09. Proceedings of the 21st Innovative Applications of Artificial Intelligence, 2009.

A. Cesta, G. Cortellessa, S. Fratini, A. Oddi, MrSPOCK: Steps in Developing an End-to-End Space Application. In Computational Intelligence, 27 (1) 83-102, 2011.

J. Chamoun, J. Kim, T. Beech, R. Saylor, "Mission Planning and Scheduling for the Lunar Reconnaissance Orbiter" American Institute of Aeronautics and Astronautics 2007.

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

S. Chien, R. Sherwood, D. Tran, B. Cichy, G. Rabideau, R. Castano, A. Davies, D. Mandl, S. Frye, B. Trout, S. Shulman, D. Boyer, "Using Autonomy Flight Software to Improve Science Return on Earth Observing One," Journal of Aerospace Computing, Information, and Communication, Apr. 2005, pp. 196–216.

S. Chien B. Cichy, A. Davies, D. Tran, G. Rabideau, R. Castano, R. Sherwood, D. Mandl, S. Frye, S. Shulman, J. Jones, S. Grosvenor, "An Autonomous Earth- Observing Sensorweb," IEEE Intelligent Systems, vol.20, no. 3, 2005, pp. 16–24.

S. Chien, D. Tran, G. Rabideau, S. Schaffer, D. Mandl, S. Frye, "Timeline based Space Operations Scheduling with External Constraints, Proceedings of the International Conference on Automated Planning and Scheduling, Toronto, Canada, May 2010.

S. Chien, G. Rabideau, J. Willis, T. Mann, "Automating Planning and Scheduling of Shuttle Payload Operations," Artificial Intelligence Journal 114 (1999) 239-255.

S. Chien, R. Sherwood, D. Tran, B. Cichy, G. Rabideau, R. Castano, A. Davies, D. Mandl, S. Frye, B. Trout, S. Shulman, D. Boyer, "Using Autonomy Flight Software to Improve Science Return on Earth Observing One, Journal of Aerospace Computing, Information, & Communication, April 2005, AIAA.

S. Chien, B. Cichy, A. Davies, D. Tran, G. Rabideau, R. Castano, R. Sherwood, D. Mandl, S. Frye, S. Shulman, J. Jones, S. Grosvenor, "An Autonomous Earth Observing Sensorweb," IEEE Intelligent Systems, May-June 2005, pp. 16-24.

S. Chien, D. Tran, G. Rabideau, S. Schaffer, D. Mandl, S. Frye, "Timeline-Based Space Operations Scheduling with External Constraints," Proceedings International Conference on Automated Planning and Scheduling, Toronto, Canada, May 2010.

C. Chouinard, R. Knight, G. Jones, D. Tran, D. Koblick, Automated and Adaptive Mission Planning for Orbital Express, Space Operations 2008, Heidelberg, Germany, 2008.

G. Curtis, T. Donaldson, R. Douglas, A. Gerb, M. Giuliano, N. Nigro, 1998. "TransVERSE an Architecture for Configuring Astronomical Observations," In *Proceedings of Astronomical Data Analysis Software and Systems VIII*, 73-76, Urbana Illinois.

G. Curtis, M. Giuliano, 1999. "COSI: Adding Constraints to the Object-Oriented Paradigm," In the *Proceedings of the LISP User Group Meeting*. San Francisco, California.

N. Ferdous, and M.E. Giuliano, 2006. Validating Resource Usage in Least Commitment Planning. *International Workshop on Planning And Scheduling for Space (IWPSS)*. Baltimore, Maryland.

J. Frank, and A. K. Jonsson, 2003. Constraint-based attribute and interval planning. Journal of Constraints Special Issue on Constraints and Planning 8(4).

A. Fukunaga, G. Rabideau, S. Chien, D. Yan, "Toward an Application Framework for Automated Planning and Scheduling," *Proceedings of the 1997 IEEE Aerospace Conference,* Aspen, CO, February 1997, v. 1, pp. 375-386.

M.P. Geyer, F. Mrowka, C. Lenzen, "TerraSAR-X/TanDEM-X Mission Planning – Handling Satellites in Close Formation", SpaceOps 2010 Book: 'Space Operations: Exploration, Scientific Utilization & Technology Development', Summer 2011.

M. E. Giuliano, 1998. Achieving Stable Observing Schedules in an Unstable World. In *Astronomical Data Analysis Software and Systems VII*. 271-274.

M. Giuliano, R. Rager, N. Ferdous, 2007. Towards a Heuristic for Scheduling the James Webb Space Telescope. In *Proceedings of the ICAPS*, Providence, RI2007, 160-167.

M. Giuliano and M. D. Johnston. Multi-Objective Evolutionary Algorithms for Scheduling the James Webb Space Telescope. International Conference on Automated Planning and Scheduling (ICAPS), Sydney, Australia, 2008.

M. E. Giuliano, M. D. Johnston, 2011. Developer Tools for Evaluating Multi-Objective Algorithms. *International Workshop on Planning and Scheduling for Space (IWPSS).* Darmstadt, Germany.

GSOC-DLR, Software applications for mission planning, http://www.dlr.de/rb/en/desktopdefault.aspx/tabid-6816/4256_read-6303/, retrieved 24 April 2012.

GSOC-DLR, GSOC Modelling Language, http://www.dlr.de/rb/Portaldata/38/Resources/dokumente/gsoc_dokumente/mb/GSOC_Modelling_Language.pdf, retrieved 24 April 2012.

L. Gutiérrez, J. Tejo, M. Pereda, I. Veiga, "***flexplan***: Deployment of powerful comprehensive Mission Planning Systems", SpaceOps 2006. Rome (Italy) 2006.

M. Johnston, R. Henry, A. Gerb, M. Giuliano, B. Ross, N. Sanidas, S. Wissler, J. Mainard, "Improving the observing efficiency of the Hubble Space Telescope," AIAA Computing 9, October 1993.

M. D. Johnston, and G. E. Miller, 1994. Spike: Intelligent Scheduling of Hubble Space Telescope Observations. In Intelligent Scheduling, M. Zweben and M. Fox Eds. Morgan Kaufmann, San Mateo, 391-422.

M.D. Johnston, M. Giuliano, MUSE: The Multi-User Scheduling Environment for Multi-Objective Scheduling of Space Science Missions. IJCAI Workshop on Space Applications of AI, Pasadena, CA, 2009.

M. D. Johnston and M. Giuliano, "Multi-Objective Scheduling for the Cluster II Constellation," 6th International Workshop on Planning and Scheduling in Space (IWPSS), Darmstadt, Germany, 2011.

R. Knight, G. Rabideau, S. Chien, "Extending the representational power of model-based systems using generalized timelines," Proc 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), Montreal, Canada, 2001.

L. A. Kramer, Generating a Long Range Plan for a New Class of Astronomical Observatories. In *2nd NASA Conference on Planning and Scheduling for Space*.

J. Marquez, M. Ludowise, M. McCurdy, J. Li, "Evolving from Planning and Scheduling to Real-Time Operations Support: Design Challenges," 40th International Conference on Environmental Systems; 11-15 Jul. 2010; Barcelona, Spain; Spain.

E. Maurer, F. Mrowka, A. Braun, M.P. Geyer, C. Lenzen, Y. Wasser, M. Wickler, "TerraSAR-X Mission Planning System: Automated Command Generation for Spacecraft Operations", IEEE Transactions on Geoscience and Remote Sensing, Vol.48, Number 2. pp. 642 – 648, February 2010.

McGann, C.; Py, F.; Rajan, K.; Thomas, H.; Henthorn, R.; and McEwen, R. 2008. A deliberative architecture for AUV control. In Proc. of Intnl. Conf. on Robotics and Automation (ICRA).

P. Morris, J. Bresina, J. Barreiro, M. Iatauro, T., Smith, "State-based scheduling via active resource solving," Proceedings of the Space Mission Challenges for Information Technology, 2011

P. Morris, J. Bresina, J. Barreiro, "Stable Grounded Inference in Flexible Resource Scheduling," Proceedings of the Workshop on Generalized Planning, National Conference on Artificial Intelligence, 2011.

N. Policella, H. Oliveira, and T. Siili, "Managing SOHO's Keyhole Periods: Problem Definition and Solving Model," Proceedings of the International Workshop on Planning and Scheduling for Space, IWPSS-09, 2009, Pasadena, California, USA.

G. Rabideau, R. Knight, S. Chien, A. Fukunaga, A. Govindjee, "Iterative Repair Planning for Spacecraft Operations in the ASPEN System," *International Symposium on Artificial Intelligence Robotics and Automation in Space*, Noordwijk, The Netherlands, June 1999.

G. Rabideau, B. Engelhardt, S. Chien, "Using Generic Preferences to Incrementally Improve Plan Quality," *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling*, Breckenridge, CO, April 2000.

S. Y. Reddy and J. D. Frank, M. J. Iatauro, M. E. Boyce, E. Kurklu, M. Ai Chang and A. K. Jonsson. *Planning Solar Array Operations on the International Space Station*. Special Issue on AI in Space, ACM Transactions on Intelligent Systems and Technology vol. 2 no. 4, July 2011.

T. Saski, G. Kosugi, R. Hawkins, J.A. Kawai, T. Kusumoto, 2004. Observation scheduling tools for Subaru Telescope. In *Optimizing Scientific Return for Astronomy through Information Technologie (SPIE Vol. 5493)*, P.J. Quinn and A. Bridger Eds. SPIE, 367-372.

B.D. Smith, B.E. Engelhardt, and D.H. Mutz,"The RADARSAT-MAMM Automated Mission Planner," AI Magazine, vol. 23, no. 2, 2002. pp. 25–36.

E. Smith and D. Korsmeyer, Utilization of Intelligent Systems Technologies for Manned Mission Operations Support. Proceedings of AIAA Space Operations Conference, 2010.

J.A. Tejo, M. Pereda, I. Veiga, J.P. Chamoun, G. Garcia and T. Beech; "***flexplan***: An Operational Mission Planning & Scheduling COTS Used Internationally", 5th International Workshop on Planning and Scheduling for Space. Baltimore (USA) 2006.

J. Tejo, F. Colmenero, "*flexplan* – Deployment of the Mission Planning System for the SMOS mission", 7th International Symposium Reducing the Costs of Spacecraft Ground Systems and Operations (RCSGSO). Moscow (Russia) 2007.