

Timeline-based Mission Operations Architecture: An Overview

Seung H. Chung* Duane L. Bindschadler†

Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109, U.S.A.

Some of the challenges in developing a mission operations system and operating a mission can be traced back to the challenge of integrating a mission operations system from its many components and to the challenge of maintaining consistent and accountable information throughout the operations processes. An important contributing factor to both of these challenges is the file-centric nature of today’s systems. In this paper, we provide an overview of these challenges and argue the need to move toward an information-centric mission operations system. We propose an information representation called *Timeline* as an approach to enable such a move, and we provide an overview of a Timeline-based Mission Operations System architecture.

I. From File-Centric to Information-Centric Mission Operations

ONE of the challenges of mission operations is in maintaining consistent and accountable information throughout the process. To maintain consistency, a mission operations system (MOS) must be able to readily compare and analyze the operations information. One example of consistency maintenance may involve comparing temperature predictions from a finite element analysis and the gathered telemetry from a variety of onboard temperature sensors. Another example of consistency maintenance may involve assuring that all MOS processes are based on a consistent set of information.

To maintain accountability, the MOS must be able to analyze the relationships between different types of operations information. One example of accountability maintenance may require accounting for all planned observations in the to be uplinked command sequences. Similarly, the MOS must account for all planned observations in the downlinked telemetry. This involves the ability of the MOS to compare and analyze the planned observations against the command sequences and the telemetry.

Unfortunately, in today’s MOSs, even a simple information comparison task can be very difficult due to many variations in information representation. These variations in information representation are due to today’s *file-centric* mission operations. In this section, we motivate the need for *information-centric* mission operations and Timelines as an effective representation for time-varying information.

A. File-centric Mission Operations Issues

Today’s MOSs are characterized by large numbers of files that are used to carry information from one component of the system to another. An example of such a file-based MOS is depicted in Figure 1.¹

In this section, we will discuss three concerns for such a file-centric MOS: MOS integration, information consistency and information accountability concerns. These are all well-known issues for ground systems. They help explain why MOS inheritance reviews (i.e. inheritance from other missions) for ground software had to become more rigorous over time.

1. MOS Integration Concerns

One of the concerns is that software interfaces are designed at the discretion of software developers and thus within conceptual “stovepipes.” The tendency is for the developers of a new or updated software tool

*Systems Engineer, System Architecture and Behaviors Group, 4800 Oak Grove Dr. M/S 301-270, AIAA Senior Member

†Asst. Program Manager for Operations, MGSS, 4800 Oak Grove Dr. M/S 264-255, AIAA Senior Member

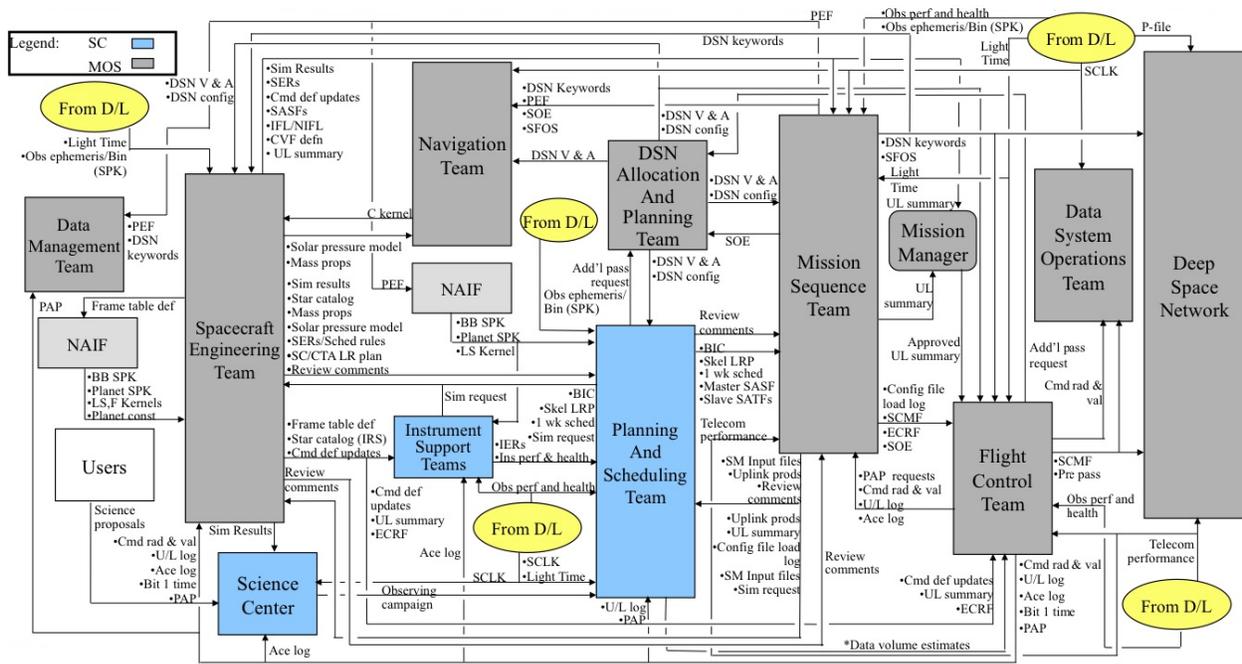


Figure 1. File-based Mission Operations System (MOS).¹ The files that are exchanged between the components of the MOS are labeled on the connectors with arrows, where each arrow depicts the direction of the information flow

to create new file formats or improve on existing ones. When generalized across the system, this tendency approaches an N^2 integration problem where each change to a given software element must also account for changing the $N - 1$ other software elements that consume or produce a particular file format*.

A second and more subtle concern is a focus on file format at the expense of information content. File formats and their three- and four-letter acronyms become the representation for a particular type of information from a particular software tool. For example, engineers speak and write about the “SPK” (i.e. Spacecraft and Planet Kernel) as opposed to the “trajectory” of a given spacecraft. This focus on format at the expense of information content results in creation of additional tools to translate various file formats, frequently implemented (and re-implemented) on a mission by mission basis.

They help explain why interfaces consume much of the system engineering effort and review board attention for any particular mission. And this integration issue surrounding file-based architectures also explain why addition of new components to a ground system generally requires significant and costly re-testing at the subsystem and system level before they can be accepted into a flight operations environment.

2. Information Consistency Concerns

In today’s file-centric systems, the mission operations processes require configuration management to keep track of multiple versions and version histories of the files in order to establish which set of files is the authoritative instance for a given step of a process. The size of these files may be large or small. Their formats are generally dictated strictly by the software used to read, edit, transform, and forward that information. Thus, even simple changes to a particular file format can prevent a particular piece of software from functioning – effectively “breaking” the MOS until either the file-producing or file-consuming software elements are “fixed”.

Our work^{1,2} to re-architect ground systems for NASA deep-space missions indicates other issues that arise from a file-centric approach. One is a tendency for information to be duplicated in several places, as shown in Figure 1, e.g. “Planet SPK” from the two uses of NAIF. This creates risk of inconsistencies and confusion as to which is the “right” or authoritative source.

*The N^2 integration is a worst case problem, but this integration problem is also evident in the MOS example depicted in Figure 1.

3. Accountability Concerns

Generally, ground systems must provide accountability for their information products. We refer to this as *reconciliation* to reflect that this accountability is provided by analyses that reconcile plans or predictions with some form of observed data. This observed data is our best insight to the actual state of the system. A number of software development efforts have been made to address accountability by developing software that “knows” the formats of files containing planned or predicted information and of files containing observed information, and that is able to perform simple or sophisticated analyses. Both the Spitzer Space Telescope and Mars Reconnaissance Orbiter missions have partially-automated software systems that help provide narrow-focus accountability information for particularly important kinds of information. However, efforts conducted over the past 5–10 years to generalize these examples and provide ground-system-wide accountability have not been able to move beyond early conceptualization, in no small part because of the lack of any unifying information framework in which to integrate all the diverse file types needed to provide meaningful, system-wide accountability.

The result is that today’s systems include a host of specialized tools to perform reconciliation, each with a narrow context. There are tools used for individual spacecraft subsystems, for individual science instruments, or parts thereof, for trajectories, for DSN tracking data, for telemetry volumes. In general, each tool must read multiple file formats and must be run with knowledge of which version of each input is the correct, authoritative version. This can be challenging because planning is intrinsically iterative and generally results in multiple versions of any particular predicted or planned information.

B. Timeline as the Unifying Information Representation

To address all these concerns, a unified information representation is needed. With a unified information representation, we no longer have an N^2 integration problem, but instead a one-to-one integration. This also allows us to focus on the information and their analyses, including information consistency and accountability, rather than on the tools and their files.

As discussed above, and depicted in Figure 1, today’s MOS operates by storing and passing files. Based on our analysis,^{1,2} we have identified many of the files to contain what we call a *time-varying information*. Figure 2 depicts examples of common mission operations related time-varying information. What is common among these examples is that they are all functions of time, or what we call *Timelines*.

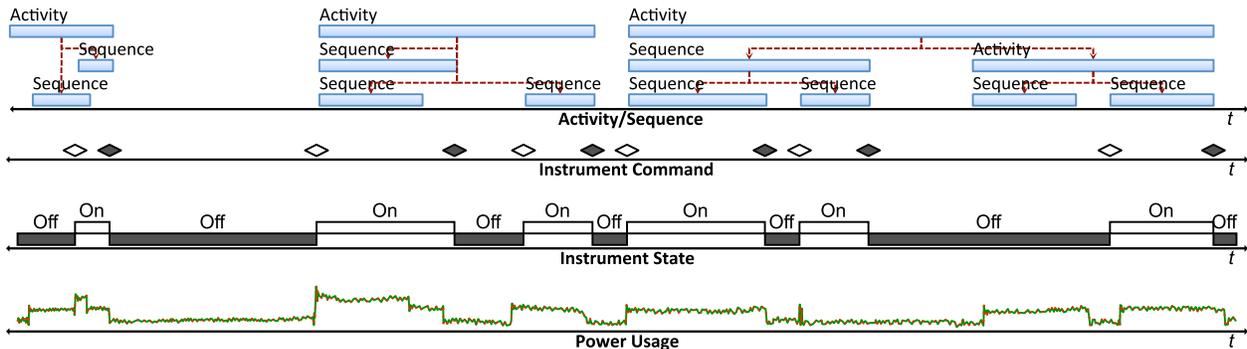


Figure 2. Examples of timelines. The first example at the top of the figure represents activities and sequences that specify our intent for an instrument on the spacecraft. The second example is instrument commands, “turn-on” and “turn-off”, that is intended to be uplinked to the spacecraft. The intended command timeline is an elaboration of the specified activities/sequences. The third example is the instrument state, “on” or “off”, predicted based on the instrument commands. Finally, the fourth example is the observed/estimated instrument power usage, extracted from the downlinked telemetry.

In mission operations, the word “timeline” has been used broadly and only notionally. From the formal representation perspective, the concept of timeline has a long lineage in the automated planning and scheduling community,³ and has a formal mathematical basis in the fields of constraint programming and predicate logic. The formalisms used in these fields are unfortunately inadequate for mission operations. For example, the formalism used in automated planning and scheduling community focuses on information representation for planning and scheduling, which is only one aspect of mission operations. Furthermore,

their notion of plans and schedules are quite different than activities and sequences used in mission planning and sequencing.

On the other hand, we could define any arbitrary timeline using very expressive mathematical expressions. However, defining a formalism that can store and correctly relay the semantics of an arbitrary mathematical expression is a nontrivial task, accomplished by only few tools such as Mathematica⁴ and Maple.⁵ But, such sophisticated analysis tools are inadequate for mission operations use in many ways. For example, their languages are too complex for most engineers and a mission operations environment. Nevertheless, defining a specific formalism that represents timelines and their semantics is achievable.

In the subsequent sections, we provide an overview of our information representation formalism called “Timeline” that we believe is sufficient as a unified time-varying information representation. We also provide an overview of an MOS architecture that is based on our Timeline concept. While this paper only provides an overview, our ongoing work² leverages a formal specification of Timeline and will fully analyze the benefits of the Timeline-based MOS architecture.

II. Timeline

Timeline is a representation for time-varying information. A simple and common example of a time-varying information is a position $x(t)$ of some object. While the codomain of a time-varying position is a three dimensional vector, i.e. $D(x) \in \mathbb{R}^3$, a timeline may be associated with more complex and less conventional codomain, e.g. a set of discrete data, such as camera images.

Generally speaking, we define *Timeline* as a representation of time-varying information; specifically, a representation of *events* in time, where events are occurrences (notable things that happen) at a given time or over a time period. While some only associate the word “event” with an occurrence at a specific point in time, e.g. Theory of Relativity, we use more general and broader definition of *event* that includes a notion of an occurrence over a time period, e.g. an event that refers to a ceremony that occurs over hours. Thus, to disambiguate the difference, we denote an event at a given time as an *instant event* and an event over a time period as a *durative event*.

While we have a formal definition of *Timeline*, we defer the discussions of the formalism and the technical details to another paper. Instead, in this section we describe Timeline semi-formally through a simple example.[†]

A. Timeline

As a simple example, consider a Timeline for a position x in one dimension. Let us assume that the position is initially zero[‡], with zero velocity, at time $t_0 = 0$ from some reference point associated with x and a reference time t . Then, at time $t_1 = t_0$, the position accelerates at a rate of 2.0 for the time duration of $t_2 - t_1 = 5$, where t_2 marks the time at which the acceleration ends. Subsequently, at t_2 , the position decelerates at a rate of 2 for the duration of $t_4 - t_3 = 8$, where t_3 and t_4 respectively mark the beginning and the end of the deceleration. Immediately following the deceleration, at time t_4 , the position accelerates once more at a rate of 0.5 until the time $t_6 = 25.0$, where t_5 and t_6 respectively mark the beginning and the end of the acceleration. Finally, at t_6 , the position maintains 0.0 acceleration for the duration of $t_8 - t_7 = 5$, where t_7 and t_8 respectively mark the beginning and the end of the zero acceleration. Furthermore, let us also constrain that the position is never less than zero. Finally, we also constrain that the position is only defined for time t greater than 0. The solution to the motion of the equation as described by this specified sequence of events and these constraints can be represented explicitly as a set of constraints specifying a parameterized piecewise continuous function, shown in Equation 1 and illustrated in Figure 3.

[†]JPL is in the process of implementing an infrastructure for storing timelines. For a discussion of timeline implementation, refer to the work by Reinholtz.⁶

[‡]While we recognize the criticality of specifying units, we have omitted the units here to simplify the discussion. Our Timeline formalism readily accommodates units on values.

$$x(t) = \begin{cases} 0, & \text{if } t = t_0 \\ (t - t_1)^2, & \text{if } t_1 < t \leq t_2 \\ -(t - t_3)^2 + 10(t - t_3) + 25, & \text{if } t_3 < t \leq t_4 \\ 0.25(t - t_5)^2 - 6(t - t_3) + 25, & \text{if } t_5 < t \leq t_6 \\ 5, & \text{if } t_7 < t \leq t_8 \end{cases},$$

$$\begin{aligned}
t_0 &= 0, \\
t_1 &= t_0, \\
t_2 - t_1 &= 5, \\
t_3 &= t_2, \\
t_4 - t_3 &= 8, \\
t_5 &= t_4, \\
t_6 &= 25, \\
t_7 &= t_6, \\
t_8 - t_7 &= 5, \\
t_\infty &= +\infty, \\
\forall t_0 \leq t \leq t_\infty \quad x(t) &\geq 0
\end{aligned} \tag{1}$$

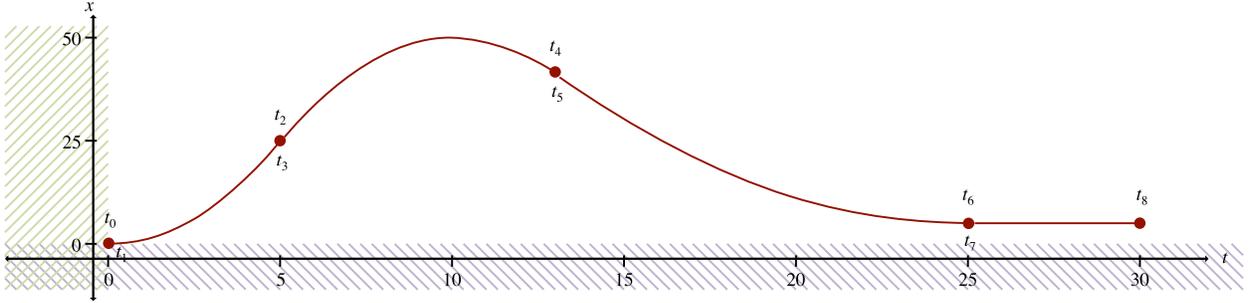


Figure 3. Position as a piecewise function of time.

Equation 1 has multiple parts: a function x that varies over time, a reference time t , time variables $t_0, t_1, \dots, t_\infty$, pairs of a value and its time range (e.g. $x(t) = (t - t_1)^2$, if $t_1 < t \leq t_2$) and temporal constraints (e.g. $t_1 = t_0$). Similarly, a Timeline $\mathcal{TL} \equiv \langle X, C, E \rangle$ is composed of a set of *variables* X (e.g. x , t and t_0), a set of *constraints* C (e.g. $t_1 = t_0$), and a set of *events* E (e.g. $x(t) = (t - t_1)^2$, if $t_1 < t \leq t_2$). Each of these is described below for our example.

1. Timeline Variables

The variables of the Timeline specified in Equation 1 are listed in Equation 2. Given a Timeline \mathcal{TL} , we denote the Timeline Variables as $X(\mathcal{TL})$, but also represented simply as X when there is no ambiguity.

$$X(\mathcal{TL}) = \{x, t, t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_\infty\} \tag{2}$$

In the set of Timeline Variables, X , there exists exactly one reference time variable t , i.e. reference clock for the Timeline. A subset of the Timeline Variables are time variables X^t that specify the times of the events $E(\mathcal{TL})$ (see Section 3 below), $X^t = \{t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_\infty\}$. In a Timeline, t_0 and t_∞ are special time variables that respectively mark the beginning and the end time points of the Timeline. The remaining variables, e.g. $X^d = \{x\}$, that are not time variables or reference time, are the *time-dependent variables*[§] X^d . Note that X^d may be multi-dimensional.

[§]Technically, x is a *function* rather than a variable. In fact, X denotes a set of mathematical terms that are not *constants*. For the simplicity of the discussion, we refer to them simply as *variables*.

2. Timeline Constraints

Timeline Constraints $C(\mathcal{TL})$ is a set of all constraints over the Timeline Variables $X(\mathcal{TL})$. The constraints define the range of values for the Timeline Variables $X(\mathcal{TL})$. A subset of the constraints are *temporal constraints* C^t that constrain only the time variables X^t . Equivalently, given a temporal constraint c^t , the scope of the temporal constraint $Scope(c^t)$, i.e. the variables referred to within the constraint, is a subset of the time variables X^t . The temporal constraints of the Timeline represented by Equation 1 are shown in Equation 3.

$$C^t = \left\{ \begin{array}{l} t_0 = 0, \\ t_1 = t_0, \\ t_2 - t_1 = 5, \\ t_3 = t_2, \\ t_4 - t_3 = 8, \\ t_5 = t_4, \\ t_6 = 25, \\ t_7 = t_6, \\ t_8 - t_7 = 5, \\ t_\infty = +\infty, \end{array} \right\} \quad (3)$$

The remaining subset of the constraints is a set of *dependent variable constraints* C^d that also constrains the dependent variables X^d over the entire duration of the Timeline, i.e $t_0 \leq t \leq t_\infty$. For example, the dependent variable constraints of the Timeline in Equation 1 is shown in Equation 4.

$$C^d = \{x(t) \geq 0\} \quad (4)$$

A temporal constraint can be broadly defined as a constraint over a set of time variables. In this paper, however, we restrict a temporal constraint as either a unary or binary *interval constraint*. For example, a *time constraint* $t_0 = 0$ is equivalent to a unary interval constraint $t_0 \in [0, 0]$. A *duration constraint* $t_2 - t_1 = 5$ is equivalent to a binary interval constraint $t_2 - t_1 \in [5, 5]$. The readers should refer to the work by Dechter⁷ for detailed descriptions and discussions of temporal constraints.

3. Timeline Events

Unlike the dependent-variable constraints of a Timeline, $C^d(\mathcal{TL})$, the Timeline events $E(\mathcal{TL})$ constrain the time-dependent variables over specific ranges of time in a piecewise manner. A Timeline has at least two *instant events*, e^I_0 and e^I_∞ , that respectively constrain the absolute beginning and the absolute end of the Timeline (see Section B for further discussions of events, including *instant events*).

For compactness, when e^I_0 is not defined, we assume that none of the variables in $X(e^I_0)$ are constrained, i.e. $C(e^I_0) = \emptyset$. Similarly, when e^I_∞ is not defined, we assume that none of the variables in $X(e^I_\infty)$ are constrained, i.e. $C(e^I_\infty) = \emptyset$.

B. Event

A Timeline is composed of two types of events, *instant events* and *durative events*. Conceptually, an instant event defines a point in a Timeline. Equation 5 is an example of an instant event e^I of the Timeline represented by Equation 1. This instant event defines the Timeline's absolute beginning $e^I_0 \in E(\mathcal{TL})$.

$$x(t) = 0, \quad \text{if } t = t_0 \quad (5)$$

An instant event $e^I \equiv \langle X, C \rangle$ is composed of a set of variables $X(e^I)$ (e.g. x , t and t_0) and a set of constraints $C(e^I)$ (e.g. $x(t) = 0$, if $t = t_0$ and $t_0 = 0$).

Conceptually, a durative event defines a line (or a hyperplane for higher dimensions) on a Timeline. Equation 6 is an example of a durative event e^D of the Timeline represented by Equation 1.

$$x(t) = (t - t_1)^2, \quad \text{if } t_1 < t \leq t_2 \quad (6)$$

Similar to a Timeline, a durative event $e^D \equiv \langle X, C, E^I \rangle$ is composed of a set of variables $X(e^D)$ (e.g. x , t , t_1 and t_2), a set of constraints $C(e^D)$ (e.g. $x(t) = (t - t_1)^2$, if $t_1 < t \leq t_2$ and $t_2 - t_1 = 5$) and a pair of instant events $E^I(e^D) = \{e^I_s, e^I_e\}$ that constrain the timeline at the beginning and the end time of the event (e.g. t_1 and t_2).

Note that the main difference between a durative event and a Timeline is that a Timeline also contains other events in addition to the instant events that mark the beginning and end. Also similar to Timelines, when e^I_s is not defined, we assume that none of the variables in $X(e^I_s)$ are constrained, i.e. $C(e^I_s) = \emptyset$. Accordingly, when e^I_e is not defined, we assume that none of the variables in $X(e^I_e)$ are constrained, i.e. $C(e^I_e) = \emptyset$.

1. Event Variables

Just like a Timeline, the set of variables of an event is partitioned into a set of time-dependent variables X^d , a singleton set of reference time variable X^r and a set of time variables X^t . The set of time-dependent variables and reference time variable of an event is identical to that of the Timeline that the event is a part of. The set of time variables of an event is a subset of the Timeline's time variables. If an event is an instant event, the event has exactly one time variable that represents the time of the instant event, i.e. a point in time (e.g. t_0 of the event represented by Equation 5). If an event is a durative event, the event has exactly two time variables that represent the beginning and end times of the durative event (e.g. t_1 and t_2 of the event represented by Equation 6).

2. Event Constraints

Similar to Timeline Constraints, a set of event constraints $C(e)$ also consists of temporal constraints $C^t(e)$ and dependent variable constraints $C^d(e)$. For example, a temporal constraint for the instant event e^I_0 is $t_0 = 0$ and for the instant event e^I_∞ is $t_\infty = +\infty$. Note that if the temporal constraints $t_0 = 0$ and $t_\infty < +\infty$ are specified in $C(e^I_0)$ and $C(e^I_\infty)$, respectively, specifying them also in $C(\mathcal{TL})$ is redundant, although allowed. A dependent variable constraint of the instant event represented by Equation 5 is the constraint specified in Equation 5. For the durative event represented by Equation 6, a temporal constraint is $t_2 - t_1 = 5$ and a dependent variable constraint is the constraint specified in Equation 6. The formal definition of event constraints follows directly from the definition of Timeline constraints of Definition ??.

C. Semantics of Timeline

The details of the formal semantics of Timeline is out of scope for this paper. Intuitively, the semantics of a Timeline is equivalent to the mathematical semantics of the set of constraints and piecewise functions it represents. For example, the semantics of the Timeline in Equation 7 is identical to the mathematics semantics of Equation 1.

$$\begin{aligned}
\mathcal{TL} &= \langle X, C, E \rangle \\
X &= \{x, t, t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_\infty\} \\
C &= \{t_1 = t_0, t_3 = t_2, t_5 = t_4, t_7 = t_6, x(t) \geq 0\} \\
E &= \{e^I_0, e^D_1, e^D_2, e^D_3, e^D_4\} \\
e^I_0 &= \langle \{x, t, t_0\}, \{t_0 = 0, x(t) = 0\} \rangle \\
e^D_1 &= \langle \{x, t, t_1, t_2\}, \{t_2 - t_1 = 5, x(t) = (t - t_1)^2\}, \emptyset \rangle \\
e^D_2 &= \langle \{x, t, t_3, t_4\}, \{t_4 - t_3 = 8, x(t) = -(t - t_3)^2 + 10(t - t_3) + 25\}, \emptyset \rangle \\
e^D_3 &= \langle \{x, t, t_5, t_6\}, \{x(t) = 0.25(t - t_5)^2 - 6(t - t_3) + 25\}, \{e^I_{3e}\} \rangle \\
e^I_{3e} &= \langle \{x, t, t_6\}, \{t_6 = 25\} \rangle \\
e^D_4 &= \langle \{x, t, t_7, t_8\}, \{t_8 - t_7, x(t) = 5\}, \emptyset \rangle
\end{aligned} \tag{7}$$

While the timeline example has the property of a mathematical function, a Timeline as described is not limited to only expressing a function. As defined, a Timeline constrains the range of the dependent variables over time, and in the most general case, it defines a set of regions within the space defined by domain and codomain.

D. Related Work

1. Relationship to TCSP

The temporal constraint satisfaction problem (TCSP) defined by Dechter⁷ is the foundational work for the Timeline described here. As such, the mapping from Timeline to a temporal constraint network is trivial. Collecting all time variables of a Timeline X^t and all temporal constraints in the Timeline and its events maps directly to a TCSP. For example, the timeline in Equation 7 maps to the TCSP shown in Equation 8. The domains of the variables D is assumed to be the same time domain, i.e. the real numbers \mathbb{R} , for all time variables.

$$\begin{aligned}
 TCSP &\equiv \langle X, D, C \rangle \\
 X &= \{t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_\infty\} \\
 D &= \{\mathbb{R}, \mathbb{R}, \mathbb{R}, \mathbb{R}, \mathbb{R}, \mathbb{R}, \mathbb{R}, \mathbb{R}, \mathbb{R}, \mathbb{R}\} \\
 C &= \left\{ \begin{array}{l} t_0 = 0, \\ t_1 = t_0, \\ t_2 - t_1 = 5, \\ t_3 = t_2, \\ t_4 - t_3 = 8, \\ t_5 = t_4, \\ t_6 = 25, \\ t_7 = t_6, \\ t_8 - t_7 = 5, \\ t_\infty = +\infty, \end{array} \right. \tag{8}
 \end{aligned}$$

The benefit of using TCSP as the foundation is that the well-established algorithms based on TCSP are applicable to Timelines as well. For example, a Timeline can be scheduled or checked for temporal consistency by using TCSP solvers.^{7,8,9} Another example is algorithms used to dispatch or execute Timelines.^{10,11,12}

2. Timeline Constraints

The Timeline formalism presented in this paper is based on the work by Williams et al.¹³ and other similar work on formalizing Timelines.¹⁴ This previous work was centered around planning, scheduling and execution applications. The Timeline presented here, however, was formalized for the purpose of representing a general time-varying information, and the formalism is indifferent to its applications. Nevertheless, because the Timeline formalism is based on *constraints*, many constraint based algorithms can be used to process and analyze a Timeline. For example, the Timeline formalism can be used to exchange information to and from automated planners^{15,16,17} and automated estimators and controllers.¹³

III. Timeline-based Mission Operations System Architecture

As discussed above, and depicted in Figure 1, today’s MOS operates by storing and passing files. The word “timeline” is used in mission operations, but only informally and implicitly. Within the Timeline-based MOS Architecture, Timeline becomes an explicit and a central part of mission operations. A simplified view of a Timeline-based MOS is depicted in Figure 4. As shown, all mission operations processes and their associated components interchange their time-varying information via the unified Timeline representation. Furthermore, the time-varying information is stored in one *Timeline Storage*[¶] (depicted in the middle of Figure 4) that is version and configuration controlled (not called out explicitly in Figure 4). Note that the Timeline Storage may be centralized or distributed depending on the need.

Such a Timeline-based MOS architecture has several benefits. While we could dedicate a paper to describing all the benefits, in this overview paper we discuss the benefits associated with the integration, consistency and accountability challenges.

[¶]Note that an implementation of the *Timeline Storage* called “Timeline Management Service”⁶ is currently being implemented.

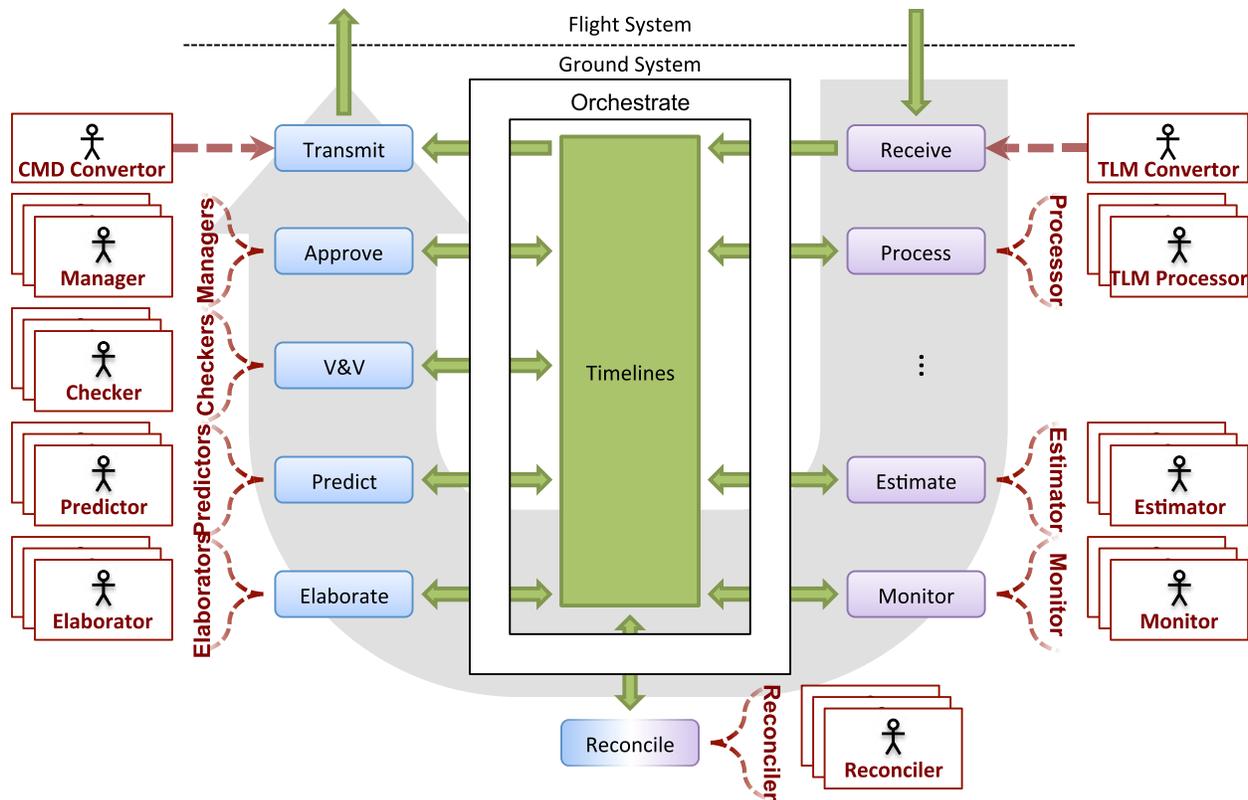


Figure 4. Timeline-based Mission Operations System Architecture. The dotted line at the top represents the boundary between the Flight System and the Ground System. The arrow going into the Flight System represents the flow of commands. The arrow coming out of the Flight System represents the flow of telemetry. The purple and blue ovals represent the downlink and uplink processes, respectively. These processes exchange information via the *Timeline Storage* in the middle.

A. Integration Benefits

With the Timeline-based architecture depicted in Figure 4, all components now only need to understand one representation for time-varying information. This eliminates the potential N^2 integration complexity and effort. Thus, rather than wasting the integration effort on managing the effects of file format changes that potentially propagate throughout the system, the effort is focused on simply integrating a component of MOS with the Timeline Storage, and that integration effort does not propagate additional effort on other component integrations.

This simplified integration complexity and effort, then leads to reduced effort in the integration verification and validation, and ultimately reduces the cost of building MOS for a mission.

B. Consistency Benefits

With a single Timeline Storage for the time-varying information, we no longer need to worry about duplicate information. This benefit is mainly due to the change from file-centric MOS to information-centric MOS. Rather than designing an MOS around the software components and the associated file formats, we are now designing MOS processes and software around the Timelines, i.e. the information, they require and must produce. With this approach, we are able to incorporate a single source of authoritative time-varying information into an MOS architecture. Therefore, the Timeline-based MOS architecture improves the consistency of time-varying information throughout the mission operations processes.

C. Accountability Benefits

Enforcing the representation of information as Timelines allows for analysis to be performed with greater rigor. Knowing that all information has identical representation makes the comparison of information more streamlined, which enables some analysis to become automated. Examining a single Timeline reveals the events that occurred in the past, the events that are occurring presently, as well as the events that will occur in the future. Comparing that Timeline to the Timeline that records the outcomes of the events provides the system with an introspection capability that is often hard to implement currently. The system is then able to see how well it performed tasks and is able to ask questions of itself such as: “Did all events occur as expected?”; “did all scheduled events execute correctly (i.e., does the Timeline of event outcomes match the Timeline of expected event outcomes)?” These types of analysis allow for greater insight into what is going on in the MOS.

IV. AMMOS Definitions for Timeline Concept

At JPL, the concept of Timeline in the context of mission operations was conceived within the Sequence Revitalization (SEQR) task led by the Mission Planning and Sequencing (MPS) element of the Multimission Ground Systems and Services (MGSS) Office. Under SEQR, the operational concept for Timeline-based planning and sequencing was developed. Currently, MPS is in the process of implementing the Timeline Management Service⁶ that is capable of storing and managing Timelines. Through socialization and reviews of the Timeline concept, the Operation Revitalization² (OpsRev) initiative also identified its applicability and criticality to overall mission operations. The Timeline concept is important to OpsRev since its charter is to improve the efficiency and the effectiveness of Mission Operations Systems (MOS) for deep-space missions. Within this initiative, Timeline has been formally defined and incorporated into the overall MOS 2.0 architecture.

As the Timeline concept became an overarching element of various efforts under MGSS, the MGSS Chief Architect organized an effort to define Timeline for Advanced Multi-Mission Operations Systems (AMMOS) program of MGSS. This effort involved the relevant members from MGSS, including OpsRev and SEQR. For architecture and implementation, we must formally define the Timeline concept, as we described in a previous section. Due to overarching applicability of the Timeline concept and the need to convey the concept to a general public, however, AMMOS required simple and informal definitions that is consistent with the formalism we presented. In this section we present the AMMOS definitions for Timeline and its concepts.

A. General Timeline Concepts

Timeline is a representation for time-varying information. Figure 2 depicts examples of common mission operations related Timelines. The first Timeline example at the top of the figure represents activities and sequences that specify our intent for an instrument on the spacecraft. Activities and sequences lie on a Timelines in which their ordering and durations are specified. The second example is a Timeline of the instrument commands, “turn-on” and “turn-off”, as specified by the activity/sequence Timeline. The third example is a discrete state Timeline that specifies the state of an instrument, either “on” or “off” as a function of time. The final example is a power Timeline that specifies what the power usage of the instrument is at a given time. What is common among these examples is that they all represent values (the codomain) over time (the domain). In the AMMOS, we define Timeline informally as follows:

Definition 1 (Timeline). A Timeline is a representation of time-varying information; more specifically, a representation of a set of *values* with associated times. The time domain of a Timeline may be discrete or continuous. The times of the *values* can be assigned or be a variable. Additionally, the range of the variable times may be restricted by *temporal constraints*.

1. Timeline Value

As described in Definition 1, a Timeline represents a set of values associated with times. The values on a timeline are the time-varying information. In the AMMOS, we define a Timeline Value as follows:

Definition 2 (Value). A Timeline value is a quantity (not necessarily numeric) belonging to the codomain specified for that Timeline.

The set of all values that Timeline may specify is the codomain of the Timeline. The Timelines in Figure 2 have varying codomains. The first Timeline example, at the top of Figure 2, has a codomain that includes actives and sequences^{||}. The second Timeline has a codomain that is a finite discrete set, including the values “turn-on” and “turn-off”. Similarly, the third Timeline has a codomain that is a finite discrete set, including the values “on” and “off”. Finally, the fourth Timeline has a Real codomain that represents power in Watts.

2. Temporal Constraint

Under certain circumstances the time associated with a value may be specified in an absolute sense**, e.g. the time-stamp on a telemetry item. Under certain circumstances, the time associated with a value may be relative, e.g. activity 2 starts within 3 seconds after activity 1 ends. We specify relative timing of the values using *temporal constraints*. Under AMMOS, we define a temporal constraint as follows:

Definition 3 (Temporal Constraint). A temporal constraint is a restriction on the possible times for particular variable times.

Figure 5 depicts various temporal relationships between activities and sequences. A time interval represents the feasible times between the activities and sequences. For example, the time interval $[0, 3]$ between Activity 1 and Activity 2 specifies that Activity 2 start between 0 and 3 seconds after Activity 1 ends. In the case of the time interval $(2, 5]$ specifies that Sequence 8 starts between 2 and 5 seconds before Sequence 6 ends.

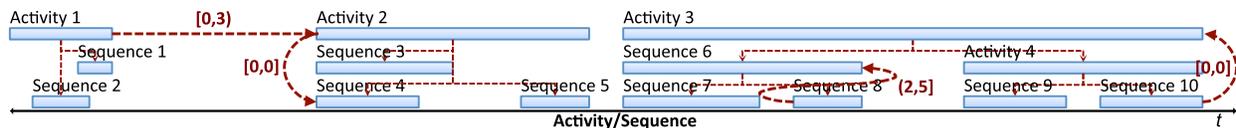


Figure 5. Examples of temporal constraints on a Timeline.

^{||}The codomain in an activity/sequence Timeline is a power set of a set of the available activities and sequences since the activities and sequence may overlap at any given time.

**For simplicity, we consider a time value to be absolute, with respect to its reference time frame.

Note that while our example uses interval algebra to specify the timing constraints, our intent for and our definition of temporal constraint is not limited to only interval algebra, and instead may include any mathematical constraints, e.g. a time constrained to a Poisson distribution.

3. Discrete and Continuous Timelines

Given the above definition of Timeline, we can categorize a Timeline by its time-domain. For time-varying information of any system, the information, i.e. a Timeline, is defined over either a discrete or continuous time-domain. For example, consider a periodic or aperiodic sampling of the power used by an instrument, depicted at the top of Figure 6.

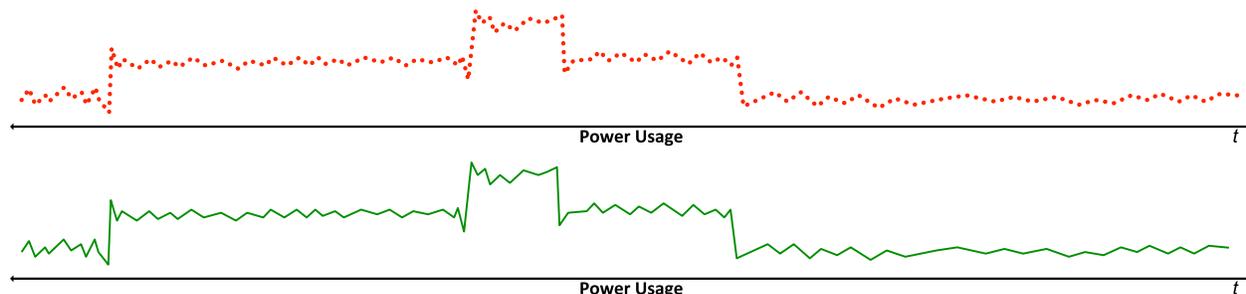


Figure 6. Examples of Discrete and Continuous Timelines.

In such a Timeline, the values are expected to be associated with a discrete, but not necessarily finite, set of time points, i.e. a discrete time-domain. Another example of such a Timeline is the instrument command timeline depicted in Figure 2. In the AMMOS, we classify such a Timeline as a Discrete Timeline and have defined it as follows:

Definition 4 (Discrete Timeline). A Discrete Timeline is a *Timeline* that represents a set of *values* at finite discrete times within a time domain.

As a different example, consider the prediction or the estimate of the power used by an instrument, depicted at the bottom of Figure 6. In such a case, based on physics, we know that the power usage of an instrument can be specified at all times, i.e. over a continuous time-domain. Typically, we expect the state or the behavior of a system to be specified over a continuous time-domain. In the AMMOS, we classify such Timeline as a Continuous Timeline and have defined it as follows:

Definition 5 (Continuous Timeline). A Continuous Timeline is a *Timeline* that represents a set of *values* for all times within a time domain.

B. Timeline Categories for Mission Operations

In this section, we identify various timeline categories for Mission Operations within the context of control systems. A control system exists to control another system that we will refer to as a system being controlled. A control system and a system being controlled exchange information. The control system sends commands to the system being controlled for the purpose of changing the behavior of the system being controlled. The system being controlled sends measurements to the control system for the purpose of providing observation into its behavior.

These categories were first conceived under SEQR, and further refined under OpsRev, and unified under AMMOS. The control system perspective is important and appropriate for mission operations since MOS is the control system for the spacecraft of a mission.

1. Timeline Categories for the System Being Controlled

Following are the Timeline categories that are relevant to the system being controlled:

Definition 6 (State Timeline). A state timeline is a *continuous timeline* that represents a state (a particular aspect or property) of the system being controlled over time. The codomain of a state (the values over which

it varies) may be a continuous set of *values* (e.g., voltage in watts) or a discrete set of *values* (e.g., on/off, discrete mode of the system being controlled) or a set of even more complex functional forms. For example, the state may be constrained to a specific *value* (i.e., assignment) or to a probability distribution over the state *values*.

Definition 7 (Command Timeline). A command timeline is, practically speaking, a *discrete timeline* that represents the information received (or may be received) by the system being controlled for the purpose of changing its behavior. Conversely, a command timeline represents the information that is sent (or may be sent) by the control system for the purpose of changing the behavior of the system being controlled. Similar to a state, the codomain of commands may be a continuous range or a discrete range or something more elaborate.

Definition 8 (Measurement Timeline). A measurement timeline is, practically speaking, a *discrete timeline* that represents the information sent (or may be sent) by the system being controlled for the purpose of providing observation into its behavior. Conversely, a measurement timeline represents the information received (or may be received) by the control system for the purpose of observing the behavior of the system being controlled. A measurement may be sensor or instrument readings or any other data received from the system. Similar to a state, the codomain of measurements may be a continuous range or a discrete range or something more elaborate.

Since commands and measurements are information exchanged between the control system and the system being controlled, both Command and Measurement Timelines are categories that are important to both the system being controlled as well as the control system.

2. Timeline Categories for the Control System

As described above, both Command and Measurement Timelines are categories that are also relevant to the Control System. Instead of re-listing them here, we refer the readers to their definitions, Definition 7 and Definition 8. Following are the additional Timeline categories relevant to the control system:

Definition 9 (Actual Timeline). An actual timeline represents data at control system interfaces of a system being controlled. A control system has access to only two actual timeline categories regarding the system being controlled: Measurement timeline and command timeline. The state of the real system being controlled is generally unobservable. Nonetheless, a simulation of the system being controlled can produce actual state timelines that are observable by testers.

Definition 10 (Intention Timeline). An intention timeline represents what the control system is trying to achieve regarding the system being controlled. An intention timeline may be elaborated into more detailed intention timelines.

Definition 11 (Estimation Timeline). An estimation timeline represents a control systems computed or analyzed information over time regarding the system being controlled based on the available actual timelines, intent timelines, and/or other estimation timelines. An estimation timeline may be further classified as prediction timeline and trend timeline.

Note that the Estimation Timeline can be further categorized into:

Definition 12 (Prediction Timeline). A prediction timeline is a type of estimation timeline that contains values for times in the future (relative to when predictions are made) based on information from actual timelines, intent timelines, and/or other estimation timelines, including prediction timelines, available at the time. In estimation theory, this represents an estimate at a time point greater than available actual data, hence, prediction.

Definition 13 (Trend Timeline). A trend timeline is a type of estimation timeline that contains values for times in the past (relative to when trends are evaluated) based on information from actual timelines, and/or other trend timelines available at the time. In estimation theory, this is equivalent to smoothing (an estimate at a time point less than available measurement data), but may also include filtering (an estimate at a time point at the end of available measurement data).

3. *Timeline Categories for the Operations System as a Control System (e.g. Mission Operations System)*

Given the Timeline categories for the system being controlled and the control system, we can combine the Timelines of the two categories to create the Timeline category we call “Operations System as a Control System”. These are the timeline categories we are finding useful, and potentially sufficient, when describing a mission operations system.⁷ We will not define the Timelines in this category since their definitions are simply derived from the aforementioned Timeline definitions. Instead, we simply list them here:

- Estimated State Timeline
- Intended State Timeline
- Actual Measurement Timeline
- Predicted Measurement Timeline
- Actual Command Timeline
- Intended Command Timeline

V. Conclusion

In this paper, we have described some of the challenges in developing a mission operations system and operating a mission. Namely, we have described the challenge of integrating a mission operations system from its components and the challenge of maintaining consistent and accountable information throughout the operations processes. We provided an overview of these challenges and the need to move toward an information-centric mission operations system. We proposed an information representation called “Timeline” as an approach to move toward information-centric mission operations system, and provided an overview of a Timeline-based Mission Operations System architecture.

Acknowledgments

We would like to acknowledge Christopher Delp who, as the Systems Architect for OpsRev, saw the benefits of Timelines and enabled further work on formalizing Timeline; Carlos Carrion who has provided the insight into and concerns of today’s mission operations system. We would also like to acknowledge Benjamin Smith, the task lead for SEQR, who conceived the discussed concept of Timeline for mission operations. Finally, we would like to thank Jeff Estefan, the MGSS Chief Architect who organized the effort to define AMMOS definitions for the Timeline concepts, and additionally, Christopher Delp, Elyse Fosse, Robert Rasmussen, William Reinholtz and Dave Santo who contributed to the AMMOS Timeline definitions.

This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

- ¹Delp, C. L., Bindschadler, D., Wollaeger, R., Carrion, C., McCullar, M., Jackson, M., Sarrel, M., Anderson, L., and Lam, D., “MOS 2.0—Modeling the Next Revolutionary Mission Operations System,” *IEEE Aerospace Conference*, Big Sky, Montana, March 5–12 2011.
- ²Bindschadler, D. L. and Delp, C. L., “Principles to Products: Toward Realizing MOS 2.0,” *Proceedings of the 12th International Conference on Space Operations (SpaceOps 2012)*, Stockholm, Sweden, June 11–15 2012.
- ³“International Conference on Automated Planning and Scheduling,” <http://www.icaps-conference.org/>, May 2012.
- ⁴*Core Language*, Wolfram Research, Inc., 2008.
- ⁵Bernardin, L., Chin, P., DeMarco, P., Geddes, K. O., Hare, D. E. G., Heal, K. M., Labahn, G., May, J. P., McCarron, J., Monagan, M. B., Ohashi, D., and Vorkoetter, S. M., *Maple Programming Guide*, Maplesoft, 2012.
- ⁶Reinholtz, W. K., “Timeline as Unifying Concept for Spacecraft Operations,” *Proceedings of the 12th International Conference on Space Operations (SpaceOps 2012)*, Stockholm, Sweden, June 11–15 2012.
- ⁷Dechter, R., Meiri, I., and Pearl, J., “Temporal Constraint Networks,” *Artificial Intelligence*, Vol. 49, No. 1, September 1991, pp. 61–95.
- ⁸Tsamardinos, I., Muscettola, N., and Morris, P., “Fast Transformation of Temporal Plans for Efficient Execution,” *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, 254–261, Madison, Wisconsin, July 26–30 1998.

⁹Moffitt, M. D. and Pollack, M. E., “Applying Local Search to Disjunctive Temporal Problems,” *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, Edinburgh, Scotland, UK, July 30–August 5 2005, pp. 242–247.

¹⁰Morris, P., Muscettola, N., and Vidal, T., “Dynamic Control of Plans with Temporal Uncertainty,” *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, Seattle, Washington, August 4–10 2001, pp. 494–502.

¹¹Shu, I.-h., Effinger, R., and Williams, B. C., “Enabling Fast Flexible Planning through Incremental Temporal Reasoning with Conflict Extraction,” *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling*, Monterey, California, June 5–10 2005, pp. 252–.

¹²Shah, J. A., Conrad, P. R., and Williams, B. C., “Fast Distributed Multi-agent Plan Execution with Dynamic Task Assignment and Scheduling,” *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS’09)*, Thessaloniki, Greece, September 19–23 2009, pp. 289–296.

¹³Williams, B. C., Ingham, M. D., Chung, S. H., and Elliott, P. H., “Model-based Programming of Intelligent Embedded Systems and Robotic Space Explorers,” *Proceedings of the IEEE: Special Issue on Modeling and Design of Embedded Software*, Vol. 91, No. 1, January 2003, pp. 212–237.

¹⁴Knight, R. L., Rabideau, G., and Chien, S., “Extending the Representational Power of Model-Based Systems using Generalized Timelines,” *Proceedings of Sixth International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, Montreal, Canada, June 18–22 2001.

¹⁵Ai-Chang, M., Bresina, J., Charest, L., Chase, A., Hsu, J. C.-j., Jönsson, A., Kanefsky, B., Morris, P., Rajan, K., Yglesias, J., Chafin, B. G., Dias, W. C., and Maldague, P. F., “MAPGEN: Mixed-Initiative planning and scheduling for the Mars Exploration Rover Mission,” *IEEE Intelligent Systems*, Vol. 19, No. 1, January/February 2004, pp. 8–12.

¹⁶Chien, S., Sherwood, R., Tran, D., Cichy, B., Rabideau, G., Castano, R., Davies, A., Lee, R., Mandl, D., Frye, S., Trout, B., Hengemihle, J., D’Agostino, J., Shulman, S., Ungar, S., Brakke, T., Boyer, D., VanGaasbeck, J., Greeley, R., Doggett, T., Baker, V., Dohm, J., and Ip, F., “The EO-1 Autonomous Science Agent,” *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004)*, New York, New York, July 19–23 2004, pp. 420–427.

¹⁷Chung, S. H., *Model-based Planning through Constraint and Causal Order Decomposition*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, August 2008.