MrSPOCK: Generating a Planning System through a Timeline Representation Framework

Amedeo Cesta, Gabriella Cortellessa, Simone Fratini and Angelo Oddi

ISTC-CNR, National Research Council of Italy
Institute for Cognitive Science and Technology
Via S. Martino della Battaglia, 44 Rome, Italy.
name.surname@istc.cnr.it

Abstract

This paper describes aspects of a project, named APSI (Advanced Planning and Scheduling Initiative), aiming at creating a general, flexible and reusable software architecture to address planning problems in space missions. It introduces our recent work that realized a general software framework for supporting development of planning and scheduling applications for space. The framework which is named TRF (Timeline-based Representation Framework) aims at supporting application development within different ESA missions and is currently being tested on three problem examples all solved on top of the TRF functionalities. The paper describes the TRF three-layerd software architecture and shows how it has been used to deploy a complete application named MrSPOCK an interactive system for MARS EXPRESS Long Term Planning.

1 Introduction

Automation of complex procedures in space mission is a need that has always represented a challenge for AI planning and scheduling (P&S) techniques. In fact planning systems research has been deeply influenced by challenges offered by space applications. Innovations have concerned initial works on temporal planning [13], real time control of the space shuttle [10], planning and execution loop, e.g., [11], the broad concept of autonomy [12], the allocation of Earth Observations on a satellite [2], negotiation tools for on-ground decision making of Mars missions [1] and so on.

A specific line of work have involved our research group[1] in an attempt of injecting P&S to support MARS EXPRESS mission planning. In particular two main efforts have been instrumental to pave the way to the current work. The first project ended up in the development of MEXAR2 [3], an AI decision support tool that helps human mission planners in managing the MARS EXPRESS dumping problem. The second project produced a tool, named RAXEM [4], that solves the complementary problem of synthesizing the uplink plans for uploading the *tele-commands* on board the satellite. Both these projects are examples of great success in introducing AI techniques within ESA mission planning contexts, and have shown clear advantages in term of performances and users' satisfaction. However, it is worth highlighting how a great effort and amount of time has been necessary to both understand the problems, capturing all the specificities, and to create a model of the relevant aspects of the domains and the problems themselves. The work done for MEXAR2 has been in some way useful for the RAXEM tool, but in general the development process has been time-consuming and extremely demanding. In addition both MEXAR2 and RAXEM were devoted to solve very specific and isolated problems, while the space missions offer many opportunities for relying on AI P&S to solve problems. This experience suggested us to operate in a more systematic way trying to identifying commonalities and similarities among the different domains and problems within the space contexts, with the aim of developing a more general and flexible approach that can be applied to different cases.

The software framework we introduce in this paper is the synthesis of our long experience as both researchers in P&S and developers of working tools for the European Space Agency (ESA). The opportunity to investigate in this direction has been provided by the APSI (Advanced Planning and Scheduling Initiative) project, an ESA

[1] http://pst.istc.cnr.it/

initiative to develop an open framework for the flexible support of mission planning systems. The result of this project has been a quite general software framework, named TRF (Timeline-based Representation Framework), which provides the basic elements for modeling the relevant entities in the space contexts. In these contexts the relevant aspects are represented by the ability to deal with *time*, *resources*, description of *operational modes*, and *synchronizations* among events. The TRF offers a structured library for managing effectively and efficiently these elements and provides the flexibilities to model different domains and problems. It is centered on the concept of *timelines* which evolve over time, a concept that is particularly suitable and close to the way of working of human mission planners, thus offering also a good metaphor for managing the interaction with the users.

This paper introduces the TRF software framework, describing the main three layers it is composed of. The TRF is then shown at work, for the modeling and resolution of a problem identified within the MARS EXPRESS program. In particular the TRF has been used to deploy an application, named MrSPOCK, also described in this paper, that is devoted to the support of Long Term Plan synthesis. The lower effort put in the application development with respect to MEXAR2 and RAXEM provides a clear proof of both the effectiveness of the architecture and the support to fast prototyping it provides. The paper briefly describes both the TRF and the MrSPOCK application and describe how the TRF has also been applied to two additional case studies, confirming its flexibility and reusability. We end the paper discussing shortly the lines for improving the current status of the TRF architecture also highlighting directions to pursue in order to extend its use to ESA robotic missions.

2 Timeline-based Representation Framework

The TRF software architecture consists of layers organized in a hierarchy. Each layer is responsible for dealing with a particular aspect of the problem, and each layer uses the services provided by the underlying layers to implement its functionalities. The constraint-based nature of the approach is extremely visible in the way the different layers exchange information: constraints are posted on the underlying levels as a consequence of decisions taken on higher levels, and decisions are taken on higher levels by analyzing the domains of the variables in the underlying levels. The architecture has been conceived to be easily extensible by adding components. This capability is very important to achieve a good balance between general, domain independent planning (easily customizable to various domains) and specialized, efficient reasoning (often needed in real world domains for efficiency reasons).

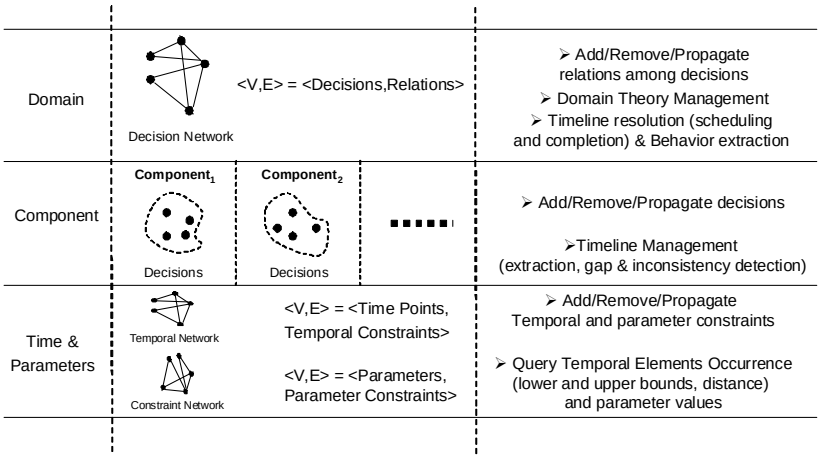

Figure 1: *The layered implementation of the* TRF.

TRF's levels are: a *Time/Parameters* layer, a *Component* layer and a *Domain* layer. Layers are organized according to the hierarchy shown in Figure 1. 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. Hence the TRF structure, where a common lower level represents the information shared among the timelines, temporal information and parameter information, a middle level that represents the extension point where the modeler plugs the components, and an upper level that provides a unified, shared representation of the plan.

Time and Parameters Layer. This is the lowest layer in the TRF's architecture. Temporal and parameters' information is managed at this level. The interface provided by this level is simple and straightforward. Higher levels create temporal elements and parameters, impose constraints on them and query the database to access the information on events temporal positions and parameters values. The temporal information is managed in shape of Temporal Constraint Networks (TCNs) [7]. TCNs allow representing *events*, also called *time points*,

and *temporal constraints* that represent distances, separation constraints, etc. This layer is endowed with propagation algorithms to maintain the consistency of the possible value assignments to time points. The current implementation is based on the Simple Temporal Problem [7]. Two propagation algorithms are defined: the first one implements an *All Pair Shortest Path* algorithm, which provides the temporal distance between each pair of time points; the second one implements a *Single Source Shortest Path* algorithm, more efficient, but provides only the temporal distances with respect to the common *reference* time point.

Parameters are managed through an external CSP solver, CHOCO [9].

Component Layer. The component layer is the point of expansion of the TRF architecture. In this architecture a component is a software module that encapsulates the logic for (1) computing a timeline resulting from decisions, temporally tagged functions of parameters; (2) evaluating the consistency of the computed timeline with respect to a set of given rules and (3) computing a set of temporal and/or parameter constraints and further decisions to solve (if possible) any threat to the consistency of the computed timeline. A couple of practical examples might help in understanding the three points.

In the current implementation the TRF provides two types of standard components: *state variables* and *reusable resources*. State variables have behaviors that are piecewise constant functions over a finite, discrete set of symbols which represent the *values* that can be taken by the state variable. Each behavior represents a different sequence of values taken by the component. The consistency notion is stated as a set of sequence constraints, i.e., a set of rules that specify which transitions between allowed values are legal, represented as a timed automaton. Resources behaviors are real functions over time. Each behavior represents a different profile of resource consumption.

Referring to the state variable and to the reusable resource components, a state variable component encapsulates the logic for computing, given a set of value choices, the resulting timeline. Temporally intersecting decisions must require the same values, otherwise the resulting timeline will be inconsistent. If two decisions that require $P(x)$ and $P(y)$ happen to overlap, the state variable component must be able to deduce $x = y$ to ensure the consistency. In the case of a reusable resource component, it encapsulates the logic for computing resource profiles given a set of allocated activities. An inconsistency is detected when n overlapping activities requires a total amount of resource greater than the maximum availability. In this case the resource component must be able to post temporal constraints between them to solve the conflict.

A component provides to higher levels basic timeline-management primitives (like timeline extraction and inconsistencies detection). It is a point of expansion because the components make the architecture independent from the actual implementation of the functionalities they provide, encapsulating component-specific algorithms and hiding differences about behaviors, inconsistency detection and resolution behind a common interface.

Domain Layer. The Domain layer manages *relations* among decisions maintaining the *decision network* updated. This is the level where concurrent threads represented by each component in the underlying level are put together to constitute the component-based *domain*: this level is in fact responsible for providing domain theory management functions (e.g., sub-goaling and/or unification possibilities) and to generate synchronizations among components. The decision network provides a unified vision of the current solution, while the synchronizations that constitute the domain theory provide a unified means for expressing the constraints that the decisions must satisfy. In the current TRF implementation an extension of the DDL.3 language [8] is used for specifying the domain theory.

It is worth pointing that the decision network and the domain theory are flexible enough for representing a wide range of different problems. It is possible to model a timeline-based planning problem and represent a plan. In fact, a timeline-based domain independent planner as OMPS [8] can be easily refactored as a solver on top of the TRF with state variables. The solver implements search procedure and heuristics, while the TRF maintain the planner search space, also providing powerful functionalities for helping in building such a search space. But also a pure scheduling problem (and its solution) can be represented with a decision network. At present we have used the TRF not only in the space project but also as a support for our research on scheduling algorithms implementing a scheduler for RCPSP/max problems which is built on top of the framework.

How to use the TRF. The TRF architecture provides the primitives to capture the specificity of an application domain and a given problem. In order to deploy a complete application it is necessary to complete the representation aspect by adding (a) a *solver engine* and (b) *user interaction services*.

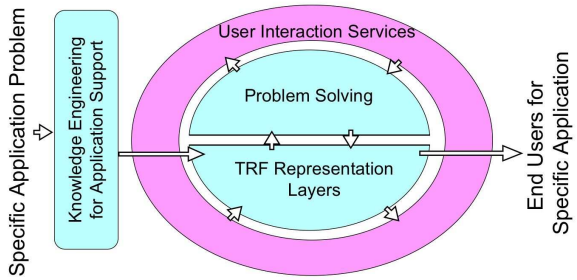

Figure 2: *Using the* TRF *to develop applications.*

Figure 2 gives an intuition on how use the TRF to develop applications. The idea is that once the representation of the domain and the problem is completed (using the TRF primitives), it is possible to use specialized solvers to efficiently solve the problem at hand, and complete the whole application by adding the interaction functionalities to support end users. It is worth highlighting how to increase the level of support for end users the modeling phase can be enhanced by knowledge engineering services that help accessing and using the TRF libraries. Currently the knowledge engineering support tool is simply the Eclipse environment which has been used to develop the TRF. Clearly this entails that the access to TRF is mainly reserved to expert users. An additional investment is desirable in order to make the TRF functionalities more useable and accessible also to non expert people. For this reason a future direction of the TRF development could be to further work in the enhancement of knowledge engineering services in order to speed up and ease the application development.

The next section provides an example of how the TRF software architecture has been used to tackle a specific application problem, showing how the idea depicted in Figure 2 has been instantiated to build a complete application for end users. The result is MrSPOCK a software tool that has been delivered to ESA to support mission planners in the synthesis of the Long Term Planning within the MARS EXPRESS mission.

3 MrSPOCK

MrSPOCK, the "MARS EXPRESS Science Plan Opportunities Coordination Kit", is a new tool which combines together diversified research aspects from the planning and scheduling area. MrSPOCK solves an interesting multi-objective optimization problem that requires the satisfaction of a number of temporal and causal constraints to produce long term plans for the MARS EXPRESS spacecraft activities. An interesting aspect of the system is the hybrid combination of a constraint-based representation that supports timeline-based planning and scheduling, an optimization algorithm that exploits such representation and an interaction front end which has multiple features. The system has been first deployed to end users during May 2008 and is currently being refined to perfectly match the details of the daily use. Apart the fielded application it is worth highlighting here the interesting leverage we obtained with respect to our previous experience in ESA projects, e.g., [3], due to the use of the TRF. This general framework has allowed us to capture an amount of constraints with a basic domain description language. Additionally the use of the timeline-based representation as a central concept for the user interaction front-end demonstrates again its particular suitability to capture the way of working of human planners in space domains.

The MEX-LTP Problem. The open problem we addressed at ESA with the MrSPOCK application was to support the collaborative problem solving process between the science team and the operation team of the space mission. 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. Goal of MrSPOCK has been to develop a pre-planning optimization tool for spacecraft operations planning and specifically we have focused on the generation of a *pre-optimized skeleton LTP* which will then be subject to cooperative science team and/operation team refinement (see [5] for a more detailed description of the whole work).

A critical point in developing an application to produce the MARS EXPRESS skeleton LTP is the consideration to be given to a great number of operational constraints that cannot be removed after four years of daily mission operation practice. In order to capture the work practice we had to cope with very specific constraints that are difficult for the general purpose solving framework but more easily to be taken into account in a domain specific solver, hence the choice of creating such solver on top of the TRF. In general it is worth underscoring that in developing application of planning and scheduling in real context the trade-off generality/specificity is a relevant one even if it is usually not mentioned in official literature. In our previous experience described in the MEXAR2 tool [3] we have used a model-based representation based on timelines and several principles of

mixed-initiative planning that are research products of our area, the whole implementation was done on-purpose for the application. In MrSPOCK the amount of the general purpose modules used in the implemented system is quite high with respect to our previous work. It is also worth mentioning that the development of a solver entirely based on domain independent solver would require the customization of an amount of specific knowledge in the domain description with a consequent production of a rather cumbersome domain model. Our choice has been to use TRF for clean modeling purposes while relying on a specific module for driving an efficient problem solving.

Modeling and Solving the Problem. MrSPOCK uses the TRF domain modeling capabilities to capture the main entities of the Long Term Plan domain within the MARS EXPRESS mission. In order to describe the components we used to model the problem it is important to introduce two different types of them (1) *Controllable Components*, whose temporal behavior is decided by the solver. They define the search space for the problem, and their timelines ultimately represent the problem solution; (2) *Uncontrollable Components* the evolution of which is exogenous to the solver. They represent values imposed over time which can only be observed; they can be seen as additional/external data and constraints for the problem.

Figure 3 shows how the MARS EXPRESS LTP domain is captured in the current release of MrSPOCK. In particular in this case we only use the state variable component type. A single *controllable* state variable models the spacecraft's pointing mode (*Pointing*), which specifies the temporal occurrence of *Science* and *Maintenance* operations as well as the spacecraft's *Communication* to Earth. The values that can be taken by this state variable, their durations (represented as a pair $[min, max]$) and the allowed transitions among the possible states are synthesized by the automaton shown in the right side of Figure 3.

As uncontrollable variables we represent ground stations (GS) availability and the occurrence of the key orbit events (Apocentre and Pericentre). The temporal occurrences of pericentres and apocentres are shown in Figure 3 ("Apo" and "Peri" values on the timeline, left/top part of the picture) and are defined in time according to an orbit event file decided by the flight dynamics team. The other state variable maintains the visibility information of three ground stations ("MAD","CEB" and "NNO" timelines left/bottom part of the figure). The allowed values of these state variables are: {`Available(?rate,?ul_dl,?antennas)`, `Unavailable()`}, where the `?rates` parameter indicates the bitrate at which communication can occur, `?ul_dl` indicates whether the station is available for upload, download or both, and the `?antennas` parameter indicates which dish is available for transmission.

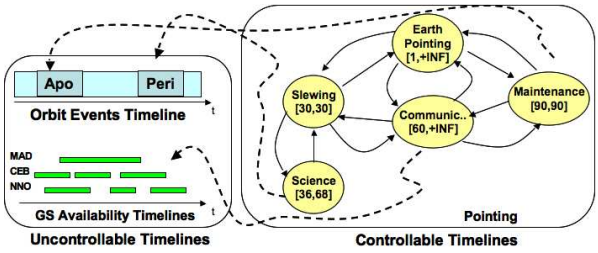

Figure 3: *MrSPOCK domain model.*

Any valid plan needs temporal synchronizations among the pointing timeline and the uncontrollable variables. These synchronization constraints are represented as dotted arrows in the figure: *Science* operations must occur during Pericentres, *Maintenance* operations must occur during Apocentres and *Communication* must occur during ground station visibility windows. As mentioned, in addition to those synchronization constraints, the Pointing timeline must respect the transitions among values specified by the automaton and the minimal and maximal duration specified for each value (in the automaton as well).

A solution is obtained when a set of consistent timelines for the controllable component are defined and all the operational constraints are satisfied. A distinctive aspect of MrSPOCK is the direction we have taken to build a problem solver once obtained the timeline representation: instead of using a generic search engine (for example the planning and scheduling integrated search of OMPS) we have built a specialized solver that dialogues directly with the problem representation in the TRF. In this way we exploits the TRF constraint engines for propagating several types of constraints, while using specialized search engines partly general partly tailored to the problem. In particular, MrSPOCK integrates a greedy one pass constructive search procedure with a generic optimization cycle that uses a genetic algorithm approach as discussed in [5]. One of the interesting achievements in our current work is the hybridization of a timeline based general purpose approach with a wrapping module that implements a genetic optimization search. It is worth underscoring again how the TRF is endowed with propagation algorithms hence it is not just a bookkeeping data structure rather it has an active role as is current practice of constraint satisfaction engines. In creating a complete architecture we situate MrSPOCK at an intermediate stage between generic timeline-based planners and the domain specific timeline-based solver described for example in [3].

MrSPOCK User-Interaction Front-End. In designing the interaction services for MrSPOCK we initially had available some basic services used as visualization functionalities for the OMPS system. They were mostly dedicated to support system developers in inspecting how part of the internal model are manipulated by the solving algorithm. Indeed a system developer is mostly interested in low level and internal details quite far away from the point of view of end-users. For this reason the best choice would be the one of designing an interaction from scratch dedicated to mission planners. Indeed the current version of the MrSPOCK interaction module somehow uses features from both these perspectives. Some of the features are completely new and dedicated to the problem, other features are adapted or evolved from those for a timeline based system. This is for two reasons: (a) because the temporal representation for the timeline is quite close to the way of taking decisions in the space domain; (b) because we had the additional goal of gaining the users' trust on the underlying approach as being not only general but also re-usable in other ESA missions. In this light we have dedicated attention to make the approach based on defining a domain model and in solving the problem by reasoning on this model more transparent and visible to the user. Somehow even if the user interface of MrSPOCK it is not also a suitable interface for the application developer, nevertheless it contains features that bring to forefront aspect of the underlying domain modeling and in general of the timeline based approach.

Main Interaction Features. The basic layout for MrSPOCK is shown in Figure 4. It is composed of a toolbar with the main commands to build instances of problem and to call and configure the solver, a message bar for the main dialogues, while the rest of the interface is mainly reserved to the timeline view. This central part describes both the uncontrollables (GS availability and Orbit Events) and the controllable (Pointing mode) components. In particular the Figure 4 shows the interface after a run of the solver and the pointing mode component presents a possible allocation of the main activities of the spacecraft (Science, Maintenance, Communication). The choice of centering the interaction on the concept of components which evolve over time allowed us taking advantage of the users' ability on reasoning over timelines to be completed and refined. Showing timelines, even in a preliminary version of the interface, resulted very useful to set up a context for the users and to facilitate our dialog with them since the early stages of the project.

The preliminary version of the interface allowed us to easily check the validity of our model for the problem, bridging the gap between us and the users.

Our second step in the development of the interaction has been to select few focal concepts to meet users' expectations on the open problem, in particular we focused on: (a) the need to explore alternative solutions, (b) the ability to control some parameters to favor an optimization criteria or another, (c) the easy visualization of the solution.

Figure 4: *Basic layout.*

Figure 5 presents a sketch of aspects that directly cope with these requirements. The main outcome of the genetic algorithm run is gathered in a solution table that gives an immediate view of the fitness values specified according to the different metrics like Science and Downlink efficiency and Uplink Tardiness. We have given the user the possibility to act on the parameters that influence the different fitness and to inspect the effects of this manipulation on the single fitness component (same table). Additionally a graphical version of the optimization values offer an alternative and cumulative view (left bottom of the figure) that allows to easily see the comparisons of alternative solutions. The connection with the existing legacy of the mission planning at ESA has been preserved by providing the users with the possibility to generate the files containing all the activities for the spacecraft in the format required (MEFs file in figure) directly from the MrSPOCK environment.

Exploiting the central concept of the timeline shared between users and system developers, an additional graphic service has been built for the users which consists in the comparison of the pointing mode timelines corresponding to alternative solutions (see bottom right of the figure).

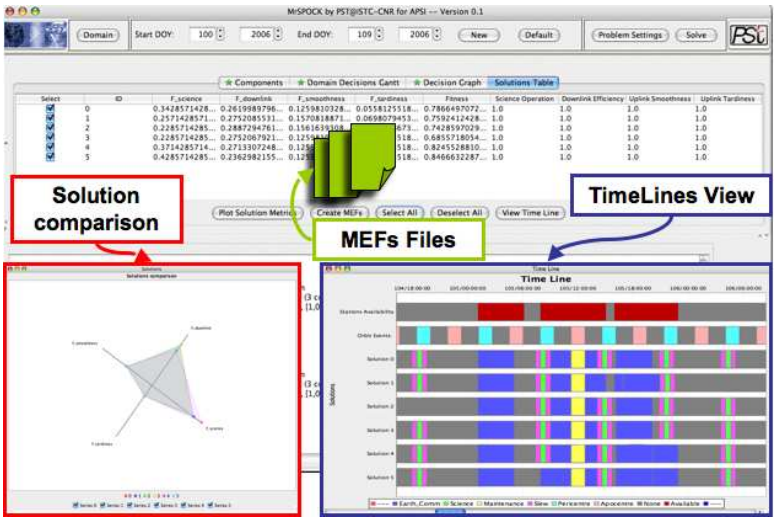

Figure 5: *Examples of interaction for end users.*

This additional graphical view guaranteed a twofold beneficial effect. On the system developer side we were able to quickly check the validity of our solving approach since the overall view highlights features of the different solutions and consequently the solving choices. On the users' side they were able to compare and reason on their choices using this environment as a means to perform "what -if" analysis. Also in this case the choice of centering the interaction on the timeline comparison, appeared particularly successful. It is possible to speculate that in space domain the idea of taking decision over time is a quite "natural concept" which facilitate the choice of the main shared concept in term of what to show to the user at first glance.

A further aspect in the user interface is dedicated to the work done to show to the users the underlying domain model. This effort is motivated by the goal of showing the user aspects connected to the reusability of this technology within different contexts and space missions. Examples of this interaction are the high level textual form domain description, an inspection of the single state variables, a graphical view of the automaton regulating the internal state transitions of the pointing mode component. This is somehow both irrelevant for the core application and also very simple but, together with other representations not shown here for lack of space, have obtained the effect of making explicit the generality of the underlying TRF representation module.

4 Discussion and Conclusions

This paper offers an overview of our recent effort within the APSI project. We have described the path that brought us to first develop the timeline representation framework (TRF) as a software development environment able to capture many of the contraints that are native of the space applications. In so doing we have created a software infrastructure which enables the rapid prototyping of solution for specific applications. In Figure 6 a general description of the APSI effort is given: the TRF plays the role of the basic infrastructure that enables three specific test cases to be developed.

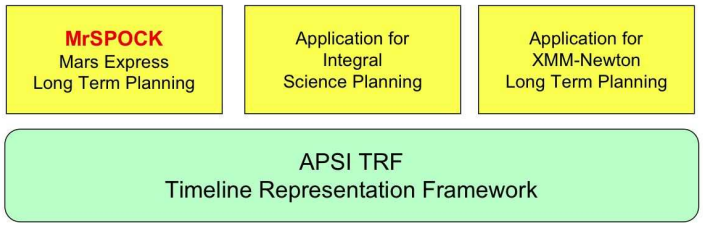

Figure 6: *The APSI use of* TRF *to deploy different applications.*

During the project after the TRF has been set up three separate groups of scientists have developed specific innovative applications: (1) first the problem of the Long Term Planning in MARS EXPRESS has been addressed as described in this paper and the MrSPOCK produced by the same group of the TRF; (2) second the problem of science planning in the INTEGRAL mission has been addressed and a solution proposed by other colleagues [14] has been delivered that provides a very effective metaheuristic for such a problem built on top of the TRF services. The proposed solution consists in an automated approach that improves significantly over the current practice; (3) the third case, currently under advanced development, focuses on the Long Term Planning of XMM-Newton.

Although the TRF can be subject to further improvements given the limited temporal extent of the APSI project it is worth underscoring the possibility that the TRF ensures to approach new application starting from a robust set of functionalities that shorten the return time for software development and allow to cope well with specification refinements in different phases of a mission. In this light a possible investment to increase the potentiality of the TRF use could be the enhancement of the knowledge engineering services, shown in Figure 2 in order to improve the process of fast modeling and to ease the access to the TRF libraries also for non expert people. It is worth underscoring the potential of a tool like TRF and the whole MrSPOCK experience also for robotic missions. Indeed we can exploit both the software development environment and the experience in the deployment of different interactive problem solvers to create innovative tools for the management of ground

segment activities. In fact it is worth noting the similarities among the various tools developed for Mars Express and the functionalities of systems used to support Mars rovers like the one descrived in [1]. Indeed we are interested in extending our approach to other critical task like the systhesis of controller and the robust loop between planning and execution. We are currently already working in two directions: (a) developing an integrated environment in which validation and verification techniques are used on timeline representations from planning systems, see preliminary results in [6], with the aim of developing planning procedures that guarantee certain formal properties; (b) exploring the loop between planning and execution in uncertain environments adding functionalities to the Trf.

Acknowledgments. Authors are partially supported by European Space Agency (ESA) within the Advanced Planning and Scheduling Initiative (APSI). APSI partners are VEGA GmbH, Onera, University of Milan and ISTC-CNR.

References

[1] Ai-Chang, M., Bresina, J., Charest, L., Chase, A., Hsu, J. C., Jonsson, 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 19*, 1 (2004), 8–12.

[2] Bensana, E., Lemaitre, M., and Verfaillie, G. Benchmark problems: Earth observation satellite management. *Constraints 4*, 3 (1999), 293–299.

[3] Cesta, A., Cortellessa, G., Denis, M., Donati, A., Fratini, S., Oddi, A., Policella, N., Rabenau, E., and Schulster, J. MEXAR2: AI Solves Mission Planner Problems. *IEEE Intelligent Systems 22*, 4 (2007), 12–19.

[4] Cesta, A., Cortellessa, G., Denis, M., Donati, A., Fratini, S., Oddi, A., Policella, N., Rabenau, E., and Schulster, J. Continuous Plan Management Support for Space Missions: the RAXEM Case. In *PAIS/ECAI-08. Proceedings of the 18th European Conference on Artificial Intelligence, pp.703-707* (2008).

[5] Cesta, A., Cortellessa, G., Fratini, S., and Oddi, A. Looking for MrSPOCK: Issues in Deploying a Space Application. In *SPARK-08, Scheduling and Planning Applications woRKshop at ICAPS* (2008).

[6] Cesta, A., Finzi, A., Fratini, S., Orlandini, A., and Tronci, E. Merging planning, scheduling & verification - a preliminary analysis. In *ASTRA-08. Proceedings of the 10th Workshop on Advanced Space Technologies for Robotics and Automation* (2008), ESA-ESTEC.

[7] Dechter, R., Meiri, I., and Pearl, J. Temporal constraint networks. *Artificial Intelligence 49*, 1-3 (Jan. 1991), 61–95.

[8] Fratini, S., Pecora, F., and Cesta, A. Unifying Planning and Scheduling as Timelines in a Component-Based Perspective. *Archives of Control Sciences 18*, 2 (2008), 231–271.

[9] http://choco.sourceforge.net/. Choco Project Web Site. `http://choco.sourceforge.net/`.

[10] Ingrand, F. F., Georgeff, M. P., and Rao, A. S. An architecture for real-time reasoning and system control. *IEEE Expert 7*, 6 (1992), 33–44.

[11] Knight, R., Rabideau, G., Chien, S., Engelhardt, B., and Sherwood, R. Casper: Space Exploration through Continuous Planning. *IEEE Intelligent Systems 16*, 5 (2001), 70–75.

[12] Muscettola, N., Nayak, P., Pell, B., and Williams, B. Remote Agent: To Boldly Go Where No AI System Has Gone Before. *Artificial Intelligence 103*, 1-2 (1998), 5–48.

[13] Vere, S. A. Planning in Time: Windows and Durations for Activities and Goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence 5*, 3 (1983), 246–276.

[14] Verfaillie, G., and Pralet, C. Modeling and solving the INTEGRAL mission long-term planning problem. Project presentation at ESA-ESOC, July 23, 2008, 2008.