Deploying RAXEM2: Planning Improvements in Daily Work Practice

Giulio Bernardi, Amedeo Cesta and Gabriella Cortellessa

Planning & Scheduling Team
ISTC-CNR – Rome, Italy
name.surname@istc.cnr.it

Abstract

This paper describes RAXEM2, an intelligent software system developed to support human mission planners in the daily task to plan uplink commands for the MARS EXPRESS mission at the European Space Agency. The system is operational at the ESA-ESOC mission control center since March 2009, replacing previous work practice. The paper describes how the system generates from previous experience, then details the components of RAXEM2, and presents an evaluation of the current performance.

Introduction

The space domain is one of those in which Planning & Scheduling (P&S) technology has demonstrated its maturity and effectiveness. The space environment offers a wide spectrum of possibilities from advanced platforms for space autonomy (Jonsson et al. 2000; Chien et al. 2005), to diversified uses on ground segment for supporting human decision making (Ai-Chang et al. 2004; Cesta et al. 2007). This domain has seen major investments in planning since the early stages (e.g., (Vere 1983)) and also the presence of research groups of significant size dedicated to the topic. Several aspects are to be addressed for a widespread use of our technology. Among these aspects, the issue of user trust is not trivial (see for example (Freed et al. 2004)).

In this paper we describe a recent effort devoted to supporting a particular aspect within the mission planning of the MARS EXPRESS mission: the command uplink problem. After the success of MEXAR2 (Cesta et al. 2007) we have been given a second sub-problem connected to MARS EXPRESS and have developed a first decision support system RAXEM (Cesta et al. 2008) in use since Summer 2007. Thanks to the effectiveness of our tool we have been given an additional set of requirements which led us to develop a new system, called RAXEM2, which now takes responsibility for the complete cycle of work practice involved in command uplink (a continuous process in spacecraft mission planning). RAXEM2 is in daily operational use since March 2009. The whole experience of injecting P&S solutions in the MARS EXPRESS mission planning practice tells a story of incremental development grounded on a trustful relationship between the daily users and the tool's developers. The work described here adds some slots to this whole picture. In particular we highlight how the enhanced version of the tool contributes to improve mission planners' work practice in many directions (e.g., time spent to create solutions, robustness and reliability of the plans, greater flexibility in producing different uplink plans, management of the whole plan life-cycle).

A key point of both RAXEM and RAXEM2 tools is to support the continuity of work of mission planners. They are in continuous contact with payload PIs and may receive commands to be uplinked distributed over time including the possibility of having to accommodate new activities in a short notice. As a consequence tools have been endowed with an interaction layer that supports incremental plan definition and management (see the specific discussion on plan management in (Cesta et al. 2008)).

As done in the rest of our work, in RAXEM2 we have used a core approach based on timeline-based modeling and solving. The core problem solver plans each command file for uplink, retains a backup window wherever possible, keeps the on-board timeline as full as feasible, and ensures the safety of the spacecraft at all times.

In RAXEM2 the "core P&S" module and the user interaction functionalities are integrated with an additional module that ensures complete and continuous management of the uplink problem. Previous work practice in fact turned out to be highly fragmented: RAXEM was part of a loop which entailed the use of an external semi-manual process in order to produce the *Spacon Instruction Form* (SIF) needed to the operator to execute the uplink plan and to maintain an updated detailed information of the on-board memory dedicated to commands status. This led to the need of having a tool to better embrace the whole work cycle in a more organic and rational way. The new RAXEM2 generates SIFs and populates a database which records the uplink history, maintaining also track of users' responsibilities and guaranteeing a continuous complete control on the uplink steps.

The paper shows how the end-to-end features and the overall management of the problem are contributing not only to support mission operations improving work practice but also to increasingly inject innovative ideas about more flexible ways of managing operations and data during mission.

The paper is organized as follows: after an introduction of the command uplink problem (formalized as MEX-UP) we underscore the additional requirements that motivated RAXEM2. The overall architecture of the system is then presented. The model of the problem is described and the new enhanced algorithm is then illustrated. A general overview of the system's use highlights the main interaction functionalities that have been designed. A final section summarizes the outcome of a preliminary experiment we performed to evaluate the system at work and reports on specific feedback received from users. Some conclusions end the paper.

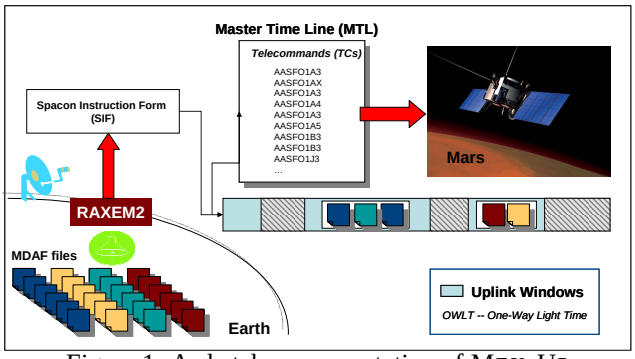

Figure 1: A sketchy representation of MEX-UP

The Problem

The MARS EXPRESS spacecraft receives instructions from Earth on a continuous basis to operate. The spacecraft is endowed with a Master Timeline (MTL), a memory buffer containing time-tagged telecommands (TCs) to be executed at specific time instants. On board, TCs are ordered by their execution time, and are removed from the MTL upon execution.

The spacecraft activities are determined in accordance with the Medium Term Plan (MTP), typically covering a period of 4 weeks. Based on the MTP various Operation Requests (OR) are generated. Finally, during the daily planning activities ORs are converted into *MTL Detailed Agenda Files* (MDAFs): each MDAF is a sequence of related TCs, typically containing between 100 and 600 of them and providing operations for up to one week. MDAFs can thus be seen as "programs", and it is fundamental to upload them completely, to avoid the risk of having the spacecraft left in an inconsistent state.

MDAFs are sent from ground stations through well defined temporal windows called *Uplink Windows* (UW) during which communication with the spacecraft is possible. It is important to note that uplink windows are half-duplex communication channels — that is, at a given instant it is only possible to send a TC or receive data from the spacecraft: they can be considered binary resources being free or available on a given instant. Moreover, each UW has an important property, named *One-Way Light Time* (OWLT), that specifies the time it takes for light to reach the spacecraft from Earth when data is sent through the uplink window.

It is also possible to upload a set of MDAFs in the same uplink window as if they were one, since this saves some time (see later for details). This set (possibly containing only one MDAF) is described in a *Spacon Instruction Form* (SIF). A SIF is a document, identified by a unique number, specifying which MDAFs should be uplinked, at what time instant this should happen, together with some other piece of information (comments, the name of the engineer that produced the SIF and so on). This document is then used by the SPACON operator to actually send data to the spacecraft.

The MEX-UP problem, summarized in Figure 1, thus consists in producing a plan to uplink as early as possible all requested MDAFs, grouped into SIFs, so that they are on board in time, given the finite capacity of the MTL and the limited bandwidth of the transmission channel, while trying to keep the MTL as full as possible.

Additional constraints are to be taken into account:

Confirmation scheme: normally, data is sent with *full confirmation*. This means that the spacecraft receives data, processes it (stores it in the MTL), and sends a confirmation message back to the ground station. In this case the uplink duration, defined as the duration of the interval that starts when the ground station starts to send data and ends when the same station receives the confirmation message from the spacecraft, is computed using the following formula:

$$2 * OWLT + num\_tc * (tc\_upl\_time + tc\_proc\_time) \quad (1)$$

where num_tc is the total number of TCs in the SIF, tc_upl_time is the upload time for TC, tc_proc_time is the processing time per TC, and OWLT is the One-Way Light Time of the Uplink Window. As already mentioned sending multiple MDAFs together saves time, since only two OWLT times are required for the whole set. To better exploit small uplink windows, it is also possible to employ *reduced confirmation*: this means that the spacecraft sends the confirmation message before processing data. Thus, uplink duration is computed as follows:

$$2 * OWLT + num\_tc * (tc\_upl\_time) \quad (2)$$

Uplink scheme: If possible, a backup window should be identified (this uplink scheme is called "with secondary window", the other being "without secondary window"). This is necessary in the case of uplink failure or to handle the eventuality that the chosen primary ground station is not available, e.g., because it has been reallocated for use by another mission with higher priority.

Caching: the MTL is endowed with a cache, containing the most immediate TCs that are to be executed (currently large enough to contain 300 TCs). If some TCs of the uplinked MDAFs have an execution time that causes them to be inserted in the cache (remember that the MTL orders TCs by their execution time), a cache reordering operation is required, resulting in the uplink duration to be enlarged. Moreover, in this case, TCs have to reach the spacecraft during a "gap" in the MTL, that is, the spacecraft must be idle during the uplink process.

In addition to the points listed above, it should be noted that planning uplink is a continuous and incremental activity, sometimes requiring a quick reaction to cope with emergency or unforeseen events. An automated solver must allow to deal with an otherwise prone to errors activity, ensuring the safety and quality of the solution at all times, with an high degree of flexibility needed to better fit user needs even in the case of extraordinary events.

Additional Requirements for RAXEM2

While RAXEM was able to produce a solution for the MEX-UP problem, it did not cover the entire management of the process, requiring the use of additional software to produce the SIFs needed to execute the plan. The success of using RAXEM in the operational environment encouraged mission planners to require an enhanced new system to further support the plan life cycle management. For this reason a set of additional requirements have been specified in order to produce a fully operational tool able to substitute the previous work practice.

SIF generator. In order to execute the produced uplink plan, mission planners had to generate Spacon Instruction Forms using semi-manual procedures involving external tools that parsed RAXEM's output. This also meant that every further customization (e.g., an arbitrary override of RAXEM's decisions in the SIF) happened outside of

RAXEM's control. There was the need for better integration and further awareness of user actions.

In particular the tool had to provide mission planners with the possibility to take decisions, while checking possible human errors. For example, the system had to allow planned MDAFs to be checked, and to generate SIFs for all those checked MDAFs. Moreover, the user had to be able to manually override RAXEM's planned primary and secondary intervals for each SIF, and to enter the time interval during which the uplink actually happened: the system had to accept all these values, providing hints and warnings about potential problems with user's decisions, and updating its internal model to reflect these changes.

Uplink database. While printed SIF forms constitute a sort of historical archive, there was the need for a more effective way of accessing uplink history. The system had to record the whole history of uplink operations, from the very first MDAF ever sent to the newest, still to be planned one. It had to allow to examine spacecraft status at arbitrary instants in the past, to reprint SIFs, to browse MDAFs, to add new ones and delete entries that were canceled. In few words, RAXEM had to reflect the current and past status of the spacecraft, allowing the user to commit its changes or rollback any unsaved operations at any time.

User management. RAXEM was installed on a single machine and used by a single person at a time (possibly over a remote X Window connection). This led to the need for a user authentication system, essentially for two purposes: (a) SIF *generation*: the name of the engineer who created the SIF had to be automatically inserted by the system in the newly generated SIF form. (b) *Tracking operations*: the system had to record user actions in per-session and per-MDAF log files, to easily review operations performed during a particular work session or on a certain MDAF. This is important for tracking responsibilities through the mission phases.

Graphical improvements. The interaction services provided with RAXEM were particularly appreciated by the mission planners. Using the interaction module, users started appreciating the new modality of work and were motivated to ask for additional functionalities of the interaction in order to improve the level of control and the what-if analysis capability. For this reason RAXEM2 had also to provide more advanced graphical features.

In addition to the work described above, the new version of the tool had to improve robustness of its solving capability.

The RAXEM2 solution

RAXEM2 is made up of various macro components, the most important ones being displayed in Figure 2: (1) the "AI module", which provides the domain modeling and problem solving capabilities, (2) the interaction module, allowing the user to interact with the system, (3) the module comprising plan life cycle services, encapsulating some key technical functionalities like the persistence layer, which deals with the database allowing problem data and historical records to be quickly retrieved and saved, the user management package, which provides user management and login services, and the SIF generator, responsible for creating the SIF forms used to control the uplink process. This last module has been instrumental for completing the plan life-cycle management which was not fully supported before.

RAXEM2 operates on the uplink database, which provides

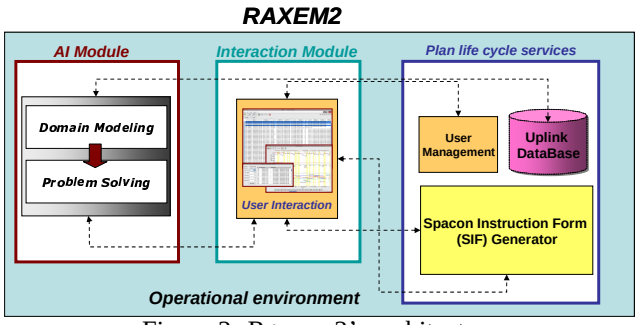

Figure 2: RAXEM2's architecture

the input to the solver and stores its output; the database can be populated importing new data in the form of MDAF files and uplink windows files, describing the MDAFs to be planned and the temporal availability of different ground stations, respectively. RAXEM2 can produce SIF forms that are directly used to drive the process of uplinking data to the spacecraft, eliminating the need of having to rely on other programs to complete the work cycle.

The rest of this section describes in more details the three modules.

Modeling with Timelines

As in any AI approach the basic step in solving the MEX-UP problem has been to build a representation (or *model*) of the domain which contains the relevant objects and constraints that influence the problem solving in the particular domain. We thus identified the following objects as being the most important to suitably represent the domain of interest:

– *MDAF* it represents an MDAF. Its relevant properties are its name, type, generation time, and execution times of its first and last TC.

– *Activity* a group of MDAFs (possibly containing only one element) that are uplinked together. Its relevant properties are the primary and secondary uplink intervals, the uplink status (to be uplinked, planned for uplink, uplink successful and so on), the confirmation scheme (full or reduced confirmation), and the uplink scheme (with or without secondary window).

– SIF it represents a SIF, which is always associated to an activity. Its relevant properties include the number, the generation time, the user who generated it, the actual uplink interval (the time interval during which the uplink was actually performed), and textual comments that can be entered by the user.

– *Window* it models a temporal "uplink window" during which it is possible to communicate with the spacecraft. Its relevant properties are the OWLT, its start and end instants, and the ID of the ground station it belongs to.

Additionally, we followed a timeline-based approach (Muscettola et al. 1992; Jonsson et al. 2000; Chien et al. 2005; Cesta et al. 2007) which focuses the attention on problem features evolving over time. Deciding on temporal evolution of the main timelines is the "meta-goal" of the problem solver. In RAXEM2 we consider the temporal evolution of two relevant system components:

– *Master Timeline (*MTL*)*. The MTL contains the set of telecommands. This can be represented as a *cumulative* resource characterized by a finite capacity and a finite cache capacity. It can report its status at arbitrary instants

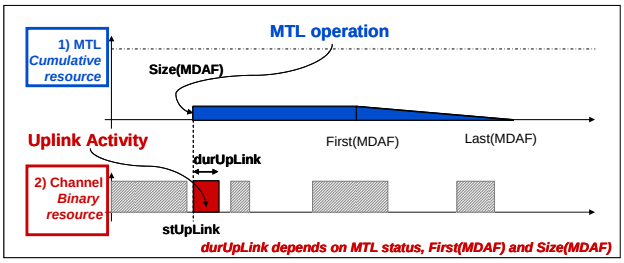

Figure 3: The MTL operation and the Uplink activity

telling what its size is, what MDAFs are on board, what TCs are going to be executed and so on.

– *Communication Channel*. The uplink connection to Earth for transmitting data. This resource, which is *binary* (either busy or free), is the union of all uplink windows, and keeps track of the availability of time intervals to be used for the uplink.

The core of the MEX-UP problem is to decide the *uploading plan*, that is, when each MDAF can be uploaded.

As shown in Figure 3, for each MDAF ready to be uplinked it is possible to identify two different activities to allocate on the previous component timelines:

– An *Uplink activity*: this is to represent the transmission over a free slot of the communication channel. This operation will require the whole bandwidth of the communication channel for the entire duration (due to the binary capacity). As highlighted in the figure the duration of the transmission depends on the execution time of the first telecommand in the file, the size of the MDAF, and on the MTL "status". Such a status is particularly relevant when the MTL is almost full to capacity, or some last-minute set of commands should be allocated directly in the on-board cache;

– An MTL *operation*: at its start time, each operation "instantaneously" stores in the Mission Timeline an amount of data equal to the number of telecommands in the MDAF, Size(MDAF). At the specified execution time, each TC will be released and removed from the MTL. In the figure a linearly decreasing behavior is given for a certain MDAF, even if this is not the general shape of the curve because TC execution time depends on the TCs distribution within a given MDAF.

The *Uplink activity* and the MTL *operation* represent together the act of uplinking data to the spacecraft, and this can be labeled with the more generic name "activity". It should be noted that, as described earlier, it is possible to uplink more MDAFs in an unique operation: an activity can thus be defined as the process of performing the uplink of one or more MDAFs coalesced together.

Solving the MEX-UP problem

The solver is the component responsible for planning a solution to the MEX-UP problem: given a set of MDAFs, whose status is set by the user as "to be uplinked", the solver must generate an appropriate uplink plan satisfying various constraints. In particular, it must determine which MDAFs can be uplinked together in an activity, specifying primary and (if appropriate) secondary uplink intervals. It can optionally relax the problem (choosing a less restrictive uplink/confirmation scheme) for some activities to maximize the number of MDAFs that are to be uplinked.

In RAXEM2 the solver has been re-engineered to improve performance and to implement some new features (see the section about experimental analysis later): when producing a plan, the software is now able to provide an estimate of how much the uplink can be delayed for each SIF to maintain the solution coherent with the constraints. This is a great help for SPACON operators in the case they cannot uplink some data in the exact instant that was planned, for a variety of practical reasons. RAXEM2 also does support the uplink/confirmation scheme "reduced confirmation with secondary window" that RAXEM didn't, improving the quality of the solution. There is also an option to reserve space for secondary windows without allowing other windows to be allocated in the same time interval; this feature is meant to avoid the need of having to replan subsequent MDAFs in the case of actual usage of the backup window instead of the primary one. While this approach is correct it should be noted that in practice this is not a case that happens very often, and may result in a less "user desired" solution since some MDAFs are planned later than they would have without this behavior: for this reason the feature is optional, and disabled by default. Moreover, the reliability has been improved to address some corner cases (though difficult to happen during the daily work) where RAXEM didn't behave as expected.

The solver employs a relatively simple *ad-hoc* algorithm providing a good trade-off between speed and optimality of the solution: in the general case, no MDAFs are discarded and the problem is not relaxed. The user can however adjust parameters to better suit his/her needs or to give hints to the solver. As the last option, it is also possible to override the solver's decisions.

Listing 1 The main algorithm of the solver

```
foreach mdaf to uplink do
    while not ALLOCATEMDAF(mdaf,currentTime) do
        if can relax then
            RELAX(mdaf)
        else
            break
    end while
    if ALLOCATED(mdaf) then
        currentTime ← PRIMARYUPLINKEND(mdaf)
end for

function ALLOCATEMDAF(mdaf, currentTime)
    start ← FIRSTAVAILABLEINSTANT(currentTime)
    while not ALLOCATED(mdaf) do
        try
            if not MULTIMDAFALLOCATION(mdaf,start) then
                SINGLEMDAFALLOCATION(mdaf,start)
        on RetryLaterError do
            start ← instantToRetry
        end try
    end while
end function
```

The algorithm can be described, at its higher level, by the pseudo-code shown in listing 1, and is shortly explained as follows: following time from left to right (initially time is set to a time origin chosen by the user) the solver tries to plan each MDAF, giving higher priority to the ones that need to be executed earlier. If the MDAF is successfully planned, the current time is set to the end time of its primary uplink interval (this ensures the monotonicity of the algorithm). If it is not possible to include the MDAF in the plan some constraints are relaxed and the process is done again, until the MDAF ends up in the plan or it is not possible to further relax the constraints.

Relaxing constraints means that the requirement of having to find a secondary window, if initially present, is discarded. If it is not enough, the confirmation scheme becomes "reduced" if it was initially "full" and so on.

When a check for a condition does fail the system is often able to retry the uplink at a later instant in which it is more likely that the condition might be satisfied, without having to trigger a constraint relax.

To include an MDAF in the plan the solver first tries to append it to an existing activity, and if this fails it tries to create a new single-MDAF activity, which will be possibly expanded later.

The single MDAF case. The solver determines the first usable time instant in which the channel is available and the MTL has sufficient room for the MDAF, and then computes the uplink duration as described by formulas 1 and 2; in this phase a lot of constraints are verified, and finally the latest instant in which the uplink can begin is determined. The solver then tries to find a secondary interval (if required to do so), preferring a window originating from a ground station different than the primary one, or ensuring that at least twelve hours have elapsed. The channel and MTL representations are then updated to reflect the addition of the MDAF to the uplink plan.

The multi-MDAF case. The solver performs a first check to determine if the last created activity and the MDAF to uplink are compatible. If this is the case, the durations of the uplink intervals are recomputed as if the MDAF was added to the activity. If all constraints are respected both for the to-be-added MDAF and for the ones that are already part of the activity, the latter is enlarged with the new MDAF and channel and MTL status are updated accordingly.

Such a relatively simple constructive algorithm serves well the need of the problem at hand. For this reason after having obtained the required new features we have focused on the complementary issues described in the rest of the paper.

Completing the uplink plan management

To better support the management of the plan life cycle, additional work has been done to build the last module of the general architecture shown in Figure 2, named plan life cycle services. This part comprises various sub-modules, shortly described here, providing persistence functionalities, user management and SIF generation capabilities, overall allowing to close the loop with the operational environment.

The persistence module. This module is responsible of implementing persistent storage of data, allowing information to be quickly retrieved and updated. Low level data access is obtained through SQLITE, a small library providing a self-contained SQL database engine backed up by a single data file.

Data which needs to be saved is constituted by those core objects that represent the past and current status of the spacecraft: more specifically, MDAFs, activities, SIFs, and uplink windows. To manage these objects life cycle, the well-known *DAO* (Data Access Object) design pattern is employed: each DAO implements *CRUD* operations (Create, Retrieve, Update and Delete) for the type of core object it handles, creating a mapping between the in-memory representation of the object and its corresponding entry in the SQL database.

The user management package. User information is contained in a file similar to the passwd file used in UNIX systems, and it is read at program start up. The user management package is instrumental for providing the ability of tracking user responsibilities, to support the addition and deletion of new users and to log in into the system.

The SIF generator. The ability of RAXEM2 to handle SIFs can be split in two distinct areas. The first involves the functionalities needed to actually produce a SIF form that can be printed and used to perform the uplink of data; this feature alone replaces the need of an external program to generate SIF forms. This is accomplished by generating HTML data that can be easily displayed on screen during the SIF editing session or printed to paper.

While this functionality is certainly valuable, another important aspect is the possibility that is given to users to completely override RAXEM2's decision by operating at SIF level, while performing checks and updating the internal representation of the problem. The implementation of this feature relies for a certain extent on the solver, which has been enhanced to provide some of its functionalities "in real time" outside of the standard planning phase.

An overview of RAXEM2 usage

In this section we present a quick tour of RAXEM2 showing some of its features and giving an intuition of the graphical improvements. However, most of the details of both aspects or advanced options are omitted for the sake of brevity.

When the tool is started the user is presented with a login window, allowing him/her to authenticate himself/herself into the system. Once the user performs the login, the main RAXEM2 window is shown (see Figure 4). The content area of the window is a view on the uplink database, showing the MDAFs contained therein in tabular format. During the work session, the user alters the spacecraft status, saving these changes via a *commit* operation or reverting to the latest saved situation with a *rollback*. These functionalities are easily accessed via the appropriate buttons in the tool bar.

Planning MDAFs for uplink. As mentioned before, RAXEM2's main task is to plan MDAFs for uplink. Typically, after having optionally added new *Uplink Windows Files* describing the available UWs, the first step is to import some new MDAFs from a directory into the database. It is then possible to alter the confirmation and uplink schemes for the MDAFs of interest to tune the solver's behavior, as highlighted in Figure 4. After the solver performs its work, the user receives visual feedback from the MDAF table, that reflects the changes that have occurred to the MDAFs involved in the plan.

Inspecting the solution graphically. In addition to the tabular view RAXEM2 provides a more advanced graphi-

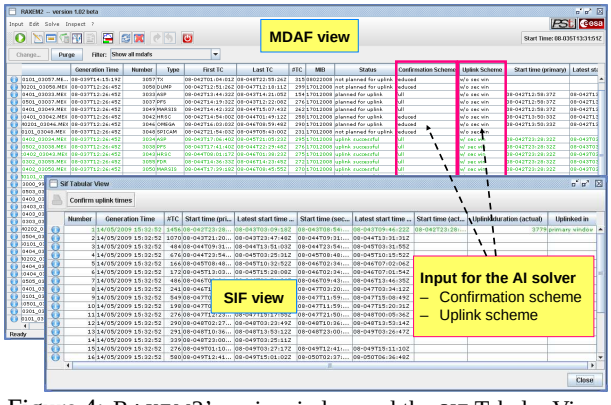

Figure 4: RAXEM2's main window and the SIF Tabular View.

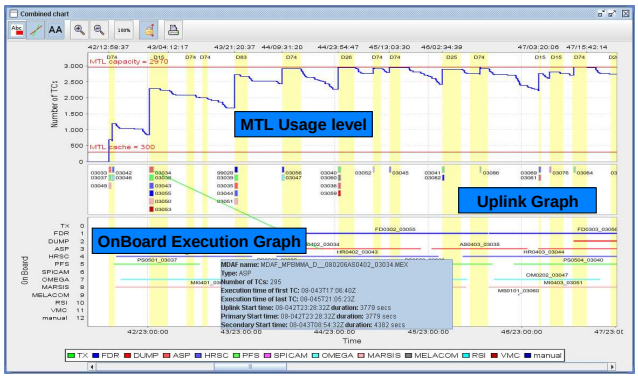

Figure 5: An aggregated view of the RAXEM2 solution which helps mission planners in making decision

cal representation of the solution. In particular an aggregated view of three graphs displays the relevant information needed to the mission planners to reason on the solution and on the spacecraft status.

In Figure 5, an aggregated view represents the following information: a) MTL graph, showing the MTL usage level during a specified time interval; b) Uplink graph, a Gantt chart displaying the MDAFs being uplinked; c) On board Execution graph, another Gantt chart showing the execution lifetime of the MDAFs while in the MTL.

This is an example of the graphical improvements that RAXEM2 is able to provide with respect to RAXEM.

Handling SIFs. Once the user asks RAXEM2 to generate SIFs for the planned MDAFs, he/she can handle them more effectively using a separate view, the SIF *Tabular View*, depicted in Figure 4. This window allows to examine all SIFs and to print multiple entries, obtaining hard-copies that can be delivered to SPACON operators to perform the actual uplink.

SIF entries can then be edited by the means of the SIF *Form View* (see Figure 6), another window that can be invoked from the SIF *Tabular View*. The SIF *Form View* displays the uplink intervals (highlighted in Figure 6) and allows to change them while providing a real time estimate of how much the uplink can be delayed. It is also possible to change the assigned request number, enter comments for the SPACON operator and print the SIF.

The final step in the management of the uplink work-cycle happens when a certain SIF has been actually uplinked to the

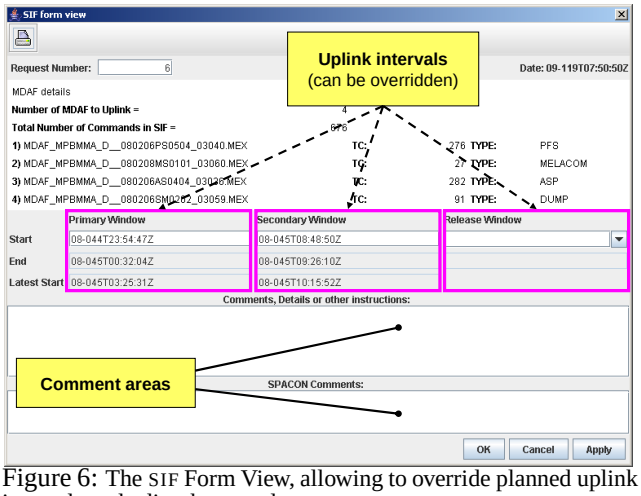

Figure 6: The SIF Form View, allowing to override planned uplink intervals and edit other SIF data.

spacecraft. When this happens, the user will typically acknowledge RAXEM2 entering the details of the time interval during which the uplink happened; the program updates its internal model and marks all MDAFs in the SIF as "uplink successful".

RAXEM2 Evaluation

This section reports results of an ongoing evaluation on the RAXEM2 tool. We present a twofold effort made to assess the tool's effectiveness. On one hand we set up a preliminary experimental analysis to compare aspects of the two versions of the delivered software RAXEM and RAXEM2. On the other hand, given the continuous feedback we are receiving by the mission planners, we describe the user's opinion of the new version of the software, highlighting the main advantages they obtained with the new work practice.

Experimental Analysis

We ran accuracy and performance tests on some sets of input data, both with RAXEM and RAXEM2.

Accuracy tests are meant as an estimate of the degree of adherence of the solution to user's expectations, that is, how much the solution complies with user's desire of having all MDAFs planned with a certain uplink and confirmation scheme combination.

Performance tests measure the time it takes for RAXEM2 to produce a plan, given an uplink and confirmation scheme combination.

We chose two particularly interesting sets of input data.

The first, *Set 1*, is made of 59 MDAFs whose lifetime spans from February 10 to 24, 2008, and of a set of uplink windows ranging from February 11 to March 10. There are four MDAFs that cannot be planned, because the execution time of their first TC is earlier than the first available uplink window. The challenge here is that, excluding the 4 MDAFs just mentioned, there are other 6 of them which need to be on board very early, and for this reason must be compulsorily sent through the first uplink window.

The second set, *Set 2*, is made of 58 MDAFs ranging from October 8 to 21, 2007 with uplink windows starting from September 24 and ending in October 21. In this case, there is one MDAF that cannot be uplinked because there is not enough space in the MTL, no matter which plan is generated: there are simply too many telecommands in the MTL at a certain instant, and this problem can only be solved using a different strategy at mission planning time (e.g. splitting one or more MDAFs so that less TCs are uplinked, or delaying some other operation).

Each test was run with different uplink and confirmation scheme combinations, abbreviated as follows:

- *full w/* Full Confirmation With Secondary Window
- *full w/o* Full Confirmation Without Secondary Window
- *reduced w/* Reduced Confirmation With Secondary Window
- *reduced w/* Reduced Confirmation Without Secondary Window

Please note that RAXEM did not support *Reduced Confirmation With Secondary Window*.

Usually test results are not influenced by the option of allowing secondary intervals to overlap: when it is the case, however, it is reported in the results.

Note that it does not make sense to compare absolute values coming from different sets, since each one represents a problem with different constraints and because of this sets made of a similar number of MDAFs can lead to different behaviors.

Accuracy Tests. For each test, a score has been computed using a weighted mean. The weights are defined as follows:

– 4 for Full Confirmation With Secondary Window
– 3 for Full Confirmation Without Secondary Window
– 2 for Reduced Confirmation With Secondary Window
– 1 for Reduced Confirmation Without Secondary Window

Each uplinked MDAF is multiplied with the weight corresponding to the uplink and confirmation mode combination used for that MDAF. The sum of these values is then divided by the total number of MDAFs multiplied by the weight of the desired uplink and confirmation mode. This way, if all MDAFs have been uplinked, and always using the desired scheme, the score will be 100%.

As mentioned earlier, there are situations where it is impossible to uplink certain MDAFs: in these cases, the score has been normalized (marked with *norm* in the tables) without taking into account those MDAFs.

Accuracy Results. Both RAXEM2 and RAXEM scored very high accuracy values (see Table 1 and Table 2). In particular they have shown to produce the same results when allowing secondary intervals to overlap, and excluding the *Reduced Confirmation With Secondary Window* combination that RAXEM did not support. For this reason, RAXEM results were omitted from the tables.

It is also worth noting the ability of the tool to support a kind of mixed-initiative problem solving, through the tuning of the solver parameters, as mentioned in the previous section. Indeed, when asked to produce a solution for *Set 1*, using Full Confirmation With Secondary Window, RAXEM2 initially planned the MDAFs as follows: 5 Not Planned; 2 Reduced Confirmation Without Secondary Window; 5 Full Confirmation Without Secondary Window; 47 Full Confirmation With Secondary Window.

Four of the five not uplinked MDAFs are the ones that need to be on board before the first available uplink window: RAXEM2 cannot do anything for them. For what regards the 6 MDAFs that must be sent through the primary window, RAXEM2 plans 3 of them with full confirmation, 2 with reduced, and fails to uplink the last one. In this case the user can tune the solution, forcing the program to uplink the first three with reduced confirmation. As a result, RAXEM2 successfully plans the 6 MDAFs with reduced confirmation (and without secondary window of course, since there is not another window available before the execution time of their first TC).

Table 1: Accuracy results for Set 1

	full w/	full w/o	reduced w/	reduced w/o
Raxem2				
std	86.86%	89.27%	86.44%	93.22%
norm	93.18%	95.76%	92.73%	100.00%

In this case, not allowing secondary intervals to overlap does not change anything: the reason is that most of the time the uplink of other MDAFs is delayed because the MTL is nearly full, so windows used by secondary intervals are not used by other MDAFs because there is not enough room to uplink them at that time.

For *Set 2*, besides the MDAF that cannot be uplinked, RAXEM2 manages to plan the others with the desired confirmation scheme, allocating backup windows (if required) for all but 6 MDAFs.

It is worth noting that not allowing secondary intervals to overlap produces slightly different results. In fact in this case there are two more MDAFs that cannot have a backup window: this is due to the fact that the secondary interval of some item could not be reused for some other MDAF.

Table 2: Accuracy results for Set 2

	full w/	full w/o	reduced w/	reduced w/o
Raxem2 - allow secondary intervals to overlap				
std	95.69%	98.28%	93.10%	98.28%
norm	97.37%	100.00%	94.74%	100.00%
Raxem2 - do not allow secondary intervals to overlap				
std	94.83%	98.28%	91.38%	98.28%
norm	96.49%	100.00%	92.98%	100.00%

Performance Tests. The tests were run on a machine equipped with a 1.8GHz AMD Athlon64 processor, single core, under Windows XP (32 bit) and using two different Java Virtual Machines: 1.5.0_18 and 1.6.0_13.

Each test was repeated 12 times, discarding the first two runs to raise the probability that the Just In Time Compiler (JIT) of the Java Virtual Machine could optimize the parts of the code involved in the test and to mitigate cache-related issues: the value assigned to each test is the mean of the remaining 10 runs, expressed in seconds.

Table 3: Performance test for Set 1

	full w/	full w/o	reduced w/	reduced w/o
Raxem2				
jvm5	0.919	0.505	0.917	0.389
jvm6	0.755	0.387	0.731	0.316
Raxem				
jvm5	1.264	1.103	n/a	1.192
jvm6	1.008	0.878	n/a	0.948
Improvement of Raxem2 over Raxem				
jvm5	37.54%	118.42%	n/a	206.42%
jvm6	33.51%	126.87%	n/a	200.00%

Table 4: Performance test for Set 2

	full w/	full w/o	reduced w/	reduced w/o
Raxem2				
jvm5	1.123	0.453	1.064	0.434
jvm6	0.814	0.360	0.784	0.352
Raxem				
jvm5	1.630	1.473	n/a	1.456
jvm6	1.236	1.200	n/a	1.138
Improvement of Raxem2 over Raxem				
jvm5	45.15%	225.17%	n/a	235.48%
jvm6	51.84%	233.33%	n/a	223.30%

The tables evidence that the time needed to find a solution is quite low even in the worst case.

Secondly, it must be noted that RAXEM2 always outperforms RAXEM by a great extent; this is because RAXEM2's solver has been redesigned with a particular attention on speed.

The results also show that when RAXEM2 is required to find a backup window time requirements are about doubled, which is due to the need for the solver to try to find two intervals instead of one. This is not true for RAXEM, because its solver was less optimized than the RAXEM2's one.

It can also be seen that the time required to find a solution is very similar for different confirmation schemes having the same uplink scheme. The reason the process is slightly slower with Full Confirmation is that with this scheme the problem can be relaxed more than with Reduced Confirmation. However, this actually happened very few times, hence the similar results.

The users' perspective

Feedback from the Mars Express mission planners has been very positive: overall, users reported that RAXEM2 has reduced the amount of engineer work load from about 5 hours to below 1 hour.

RAXEM2 integrated smoothly in the well-established work practice, also because the previous version of the tool has been used daily since the late summer of 2007. For this reason, the learning curve needed to master RAXEM2's features has been very short. In the following the main outcomes of an extensive testing phase of RAXEM2 performed by users are reported which compare RAXEM2 and RAXEM in terms of performance, continuous support to plan management, SIF handling and ability to perform what-if analysis.

Performance. For a small period, a "parallel planning" was performed using RAXEM and RAXEM2 together on the same set of input data, for testing purposes. The testing phase shown the latter to be faster than its predecessor due to improvements to the solver, confirming the results of the experimental analysis. Users also reported positive evaluations regarding database performance, both while saving and retrieving data.

Support to plan management. RAXEM2 has demonstrated to be up to expectations with regards to emergency re-planning of MDAFs: it has actually happened that users needed to quickly generate an alternative uplink plan due to the loss of an uplink window, and the tool always handled the situation seamlessly.

SIF handling. SIF generation also proved to be a very valuable feature. With RAXEM2's, the dependency on the external tool that was used to create SIF forms has been removed, transforming a two-stage process in a single-stage effort, thus avoiding the need to interrupt the work flow.

Users particularly appreciated the consistency checking ability and the respective feedback that the tool gives when editing SIF data. In some cases, this prevented the operators to make mistakes that would have resulted in a failed uplink, avoiding the consequent waste of work hours and the need for an emergency replan of failed data.

What-if analysis. Enhancements in the graphical capabilities have shown to be useful for "what-if" analysis: at mission planning level, the charts generated by RAXEM2 are used to provide a first level of feasibility analysis, i.e. to evaluate if the plan is feasible with the chosen number of MDAFs, or if the station windows are used optimally.

Finally, in terms of reliability and stability, users reported that so far RAXEM2 has always generated correct plans, without discarding a single MDAF, and neither crashes nor loss of data have been experienced.

Conclusions

In this paper we reported our experience in the development of an end-to-end system able to support mission planners in the uplink problem which entails the synthesis of plans to uplink telecommands needed to operate the MARS EXPRESS probe. The tool is the result of a two-step effort which first produced an initial version, named RAXEM, then an enhanced and optimized system which overcomes the previous one in terms of performance, reliability, flexibility and users' satisfaction.

Our experience shows that Planning & Scheduling is one aspect of the problem, but this alone is not enough to achieve success. The key ability is that of integrating different technologies in a coherent way to provide the users with a complete system to effectively perform their work (see also (Fernández-Olivares et al. 2006)).

The paper focused on aspects related both to the enhancement of P&S module and to the synthesis of new functionalities introduced for user-interaction and plan management. The synergy of the three components contributed to create a comprehensive and intelligent support environment which offers a valid help to mission planners for the entire lifecycle of the MEX-UP problem. Time for the generation of uplink plans decreased considerably, plans are more robust and reliable, and the tool guarantees a level of flexibility which enables mission planners to cope with the incremental and in some case unpredictable nature of the problem. All these aspects made RAXEM2 a valuable substitute for the previous work practice at ESA since March 2009.

Acknowledgments

This activity is supported by the European Space Agency under contract 21522/08/D/HK ("MARS EXPRESS Mission Planning Support Tools"). Authors are indebted to Erhard Rabenau for his continuous effort in pushing us to improve RAXEM2 through different intermediate steps until its actual operational use. Thanks to Nicola Policella, Simone Fratini, Angelo Oddi and Jonathan Schulster for previous work on RAXEM and for their assistance on various issues. Thanks to Michel Denis and Alessandro Donati for providing support to the project.

References

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. 2004. MAPGEN: Mixed-Initiative Planning and Scheduling for the Mars Exploration Rover Mission. *IEEE Intelligent Systems* 19(1):8–12.

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

Cesta, A.; Cortellessa, G.; Denis, M.; Donati, A.; Fratini, S.; Oddi, A.; Policella, N.; Rabenau, E.; and Schulster, J. 2008. Continuous Plan Management Support for Space Missions: the RAXEM Case. In *PAIS/ECAI-08. Prestigious Applications on Intelligent Systems in Proceedings of the 18^{th} European Conference on Artificial Intelligence*, 703–707.

Chien, S.; Sherwood, R.; Tran, D.; Cichy, B.; Rabideau, G.; Castano, R.; Davies, A.; Mandl, D.; Frye, S.; Trout, B.; Shulman, S.; and Boyer, D. 2005. Using autonomy flight software to improve science return on earth observing one. *Journal of Aerospace Computing, Information, and Communication* 2(4):196–216.

Fernández-Olivares, J.; Castillo, L. A.; García-Pérez, Ó.; and Palao, F. 2006. Bringing Users and Planning Technology Together. Experiences in SIADEX. In *ICAPS-06. 16^{th} International Conference on Automated Planning and Scheduling*, 11–20.

Freed, M.; Bonasso, P.; Ingham, M.; Kortenkamp, D.; Pell, B.; and Penix, J. 2004. Trusted Autonomy for Spaceflight Systems. In *Proceedings 1^{st} Space Exploration Conference: Continuing the Voyage of Discovery (available from American Institute of Aeronautics and Astronautics)*.

Jonsson, A.; Morris, P.; Muscettola, N.; Rajan, K.; and Smith, B. 2000. Planning in Interplanetary Space: Theory and Practice. In *AIPS-00. Proceedings of the Fifth Int. Conf. on Artificial Intelligence Planning and Scheduling*.

Muscettola, N.; Smith, S.; Cesta, A.; and D'Aloisi, D. 1992. Coordinating Space Telescope Operations in an Integrated Planning and Scheduling Architecture. *IEEE Control Systems* 12(1):28–37.

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