

# MEXAR2 – An Operational Tool for Continuous Support to Mission Planning

Amedeo Cesta, Gabriella Cortellessa, Simone Fratini, Angelo Oddi, Nicola Policella

ISTC-CNR, National Research Council of Italy  
Institute for Cognitive Science and Technology  
Via S. Martino della Battaglia 44, I-00185 Rome, Italy  
{name.surname}@istc.cnr.it

## Abstract

This paper describes MEXAR2, a software tool that is currently used to synthesize the operational commands for data downlink from the on-board memory of the MARS EXPRESS spacecraft to the ground stations.

The system generates from a two steps effort. A first study, performed before the launch of MARS EXPRESS, has produced a formalization of the memory dumping problem and developed the MEXAR prototype. This first result shows the feasibility of using an artificial intelligence constraint-based approach to solve the problem.

A second effort has studied the real data flow during actual Mars operations, integrated more sophisticated solving algorithms and produced MEXAR2 which is successfully inserted in the overall mission planning process. The tool has been in daily use by the Mission Planning Team of MARS EXPRESS at the European Space Agency since February 2005.

Goal of this paper is to present a complete overview of how the planning and scheduling problem has been addressed, a complete application customized and put into context in a challenging application environment.

## The story: in search of real data

In the period November 2000 - July 2002, authors research group has worked at the European Space Agency (ESA) study “Efficient Planning Algorithms for an Interplanetary Mission”. This study was aimed at demonstrating Artificial Intelligence techniques for Planning and Scheduling applied to a mission planning problem within the MARS EXPRESS space program. An open problem, jointly identified by the research group and the MARS EXPRESS Mission Planning experts, has been studied and formalized as the Mars Express Memory Dump Problem (MEX-MDP). In May 2002, the group delivered to the ESA-ESOC center an advanced prototype called MEXAR, able to automate the generation of spacecraft operations for efficient on-board mass memory dumping. The software was composed of two main parts: (a) a Problem Solver able to address the MEX-MDP with different algorithms, and (b) and Interaction Module devoted to both facilitate user access to the solver, and provide a set of inspection functionalities over the solution. The various results have been presented in a series of official publica-

tions (see for example (Oddi *et al.* 2003; Cesta *et al.* 2003; Oddi & Policella 2004; Cortellessa *et al.* 2004)).

While developing MEXAR the group has acquired a substantial amount of background information on mission planning, difficult to obtain unless tightly cooperating with an operational space environment. Additionally, MEXAR has provided the mission planning team a unique example of intelligent software technologies integration to address a problem in their daily activities. Nonetheless, a very important limitation plagued the first project: the unavailability of real data from the spacecraft or even from a realistic simulation of the spacecraft’s nominal work. As a consequence, although the study had a very positive outcome, it remained at the level of “nice demo”. Its results were not used at mission time, as the tool was never validated against real mission data. In June 2004, contacts with the Mission Planning Team of MARS EXPRESS started again. It clearly emerged that during the first six months of spacecraft activities around Mars, the mission planners had faced serious manpower overload in addressing the Spacecraft Memory Dumping Problem. The downlink activities were synthesized mostly manually by a team of people continuously dedicated to this task. Authors proposed to start a specific study aimed at understanding the problem on the basis of real data analysis. Regular contacts with senior members of the ESOC Mission Planning Team allowed them to get acquainted with both those data and an amount of specific constraints that were introduced to cope with operational problems. After three months, authors have been able to deliver to ESA an increasingly accurate operational version of a software able to cope with real problem instances, and real data files. During a subsequent period of three months the tool has been tested back to back against the previous semi-manual procedure developed within the team. Since February 2005 the new operational system called MEXAR2 is in continuous use at ESA-ESOC as the main tool to solve MEX-MDP instances. It directly synthesizes commands that implement the data downlink policy from on board memory to Earth. With further work authors have robustified the tool with additional functionalities and a user interface that facilitates its management. This paper is a short report on the “problem life cycle” MEXAR2 is inserted in, the features of

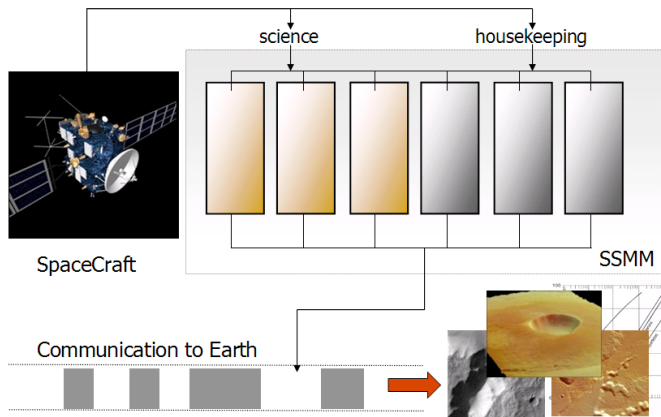


Figure 1: The on-board vs. ground segment data flow

the current tool, and an evaluation of its behavior.

### Analyzing the problem

In a deep-space mission like MARS EXPRESS data transmission to Earth represents a fundamental aspect. In this domain, a space-probe continuously produces a large amount of data resulting from the activities of its payloads and from on-board device monitoring and verification tasks (the so-called *housekeeping* data). All these data are to be transferred to Earth during a number of downlink sessions. Moreover, in the case of MARS EXPRESS a single pointing system is present. This implies that, during regular operations, the space-probe either points to Mars, to performs payload operations, or points to Earth, to download the produced data. As a consequence, on-board data generally require to be first stored in a Solid State Mass Memory (SSMM) and then transferred to Earth. Therefore, the main problem to be solved consists in synthesizing sequences of spacecraft operations (*dump plans*) necessary to deliver the content of the on-board memory during the available downlink windows. This allows to save upcoming pieces of information without losing previously stored data and to optimize given objective functions.

In Fig. 1 the main data flow addressed in our work is shown: the spacecraft instruments produce data (both science and housekeeping) stored in the on-board memory subdivided into slots called *packet stores*. The problem goal is to create commands for downlinking such data to Earth during the interval in which the communication channel with the ground station is active. In general, synthesizing the plan involves managing the constraints due to the bounded on-board memory and the different data transmission rates of the communication windows. The planned activities of the satellite are incrementally refined as soon as a specific operational period approaches. Same activities are not predicted in advance and a *short-term plan* is generated each one or two days. This two days plan is part of the set of commands uplinked to the satellite. Additionally due to dif-

ferences in compression algorithms the amount of data produced by some of the activities cannot be exactly predicted in advance creating once in a while the need to recompute the short term plan very quickly before the commands up-link.

**Domain entities.** To model the problem we have subdivided the basic entities that are relevant to the MEX-MDP domain into *resources* and *activities*: resources represent domain subsystems able to give services, whereas activities model tasks to be executed using resources over time. We model two different types of resources: (a) the SSMM that is used to store both science and housekeeping data. The SSMM is subdivided into *packet stores*,  $\{pk_1, pk_2, \dots, pk_{np}\}$ , each one with a fixed capacity,  $c_i$ , and priority,  $pr_i$ . Each packet store can be seen as a file of given maximum size that is managed cyclically, that is, previous pieces of information will be overwritten, and then lost, if the amount of stored data overflows the packet store capacity; (b) the *Communication Channel* that represents the downlink connections to Earth for transmitting data. This resource is characterized by a set of separated communication windows  $CW = \{cw_1, cw_2, \dots, cw_{nw}\}$  that identify intervals of time in which downlink connections can be established. Each element  $cw_j$  is a 3-tuple  $\langle r_j, s_j, e_j \rangle$ , where  $r_j$  is the available data rate during the time window  $cw_j$  and  $s_j$  and  $e_j$  are respectively the start and the end-time of this window.

Activities describe *how* resources can be used. Each activity  $a_i$  is characterized by a fixed duration,  $d_i$ , and two variables,  $s_i$  and  $e_i$ , which respectively represent its start-time and end-time. Two basic types of activity are relevant to MEX-MDP, store operations  $st_i$  and memory dumps  $md_i$ . Each store operation,  $st_i$ , “instantaneously” stores, at its end-time  $e_i$ , an amount of data  $q_i$  in a defined packet store,  $pk_i$ . Through a memory dump,  $md_i = \langle pk_i, s_i, e_i, q_i \rangle$ , an amount  $q_i$  of stored data is transmitted from the packet store  $pk_i$  to the ground station, during the interval  $[s_i, e_i)$ .

Two different data types are modeled using store operations: the *Payload Operation Request* (POR) and the housekeeping activities. *PORs* are scientific observations which generate a set of data distributed over the available packet stores. Housekeeping activities generate a flow of data with constant rate which is to be stored in the dedicated slots of the SSMM.

**Problem definition.** A single MEX-MDP instance is composed of a set of scientific observations, a set of housekeeping productions, and a time horizon  $\mathcal{H} = [0, H]$ . The *solution* to a MEX-MDP is a set of dump commands  $S = \{md_1, md_2, \dots, md_{nd}\}$  such that:

- At each instant of  $t \in \mathcal{H}$ , the amount of data stored in each packet store  $pk_i$  does not exceed the packet store capacity  $c_i$ , i.e., overwriting is not allowed;

- The whole set of data is “available” on ground within the considered temporal horizon  $\mathcal{H} = [0, H]$ , except an amount of residual data for each packet store  $pk_i$  lower or equal to the capacity  $c_i$ ;
- Each dump activity,  $md_i$ , is executed within an assigned time window  $cw_j$  which has a constant data rate  $r_j$ . Additionally, dump commands cannot mutually overlap.

Several additional constraints are considered by the solver. One of these constraints concerns the mentioned distinction between *housekeeping* and *science data* that have different relevance for mission planners and, for this reason, are stored in separate packet stores. The science devices allow to maintain on-board residual data, while housekeeping data have to be emptied by the end of each mission day because the status of the spacecraft should be checked continuously. A further constraint for the housekeeping packet stores requires to download them in a single dump, without preemption.

In general a solution should satisfy all the imposed constraints. A further goal is to find *high quality* solutions with respect to some specified metrics like *plan size*, *robustness*, etc. Informally, *a high quality plan delivers all the stored data (no overwriting is allowed), contains the smallest number of dump activities, satisfies the priorities preferences imposed on the set of packet stores and is able to “absorb” external modifications that might arise in a dynamic execution environment.*

It is worth noting that the MEX-MDP is mostly a planning problem because the key aspect consists in the synthesis of the sequence of dump commands  $md_j$ . To solve MEX-MDPs, we have introduced two levels of abstraction: (a) a first level, *Data Dump level*, assesses the amount of data to dump from each packet store during each time window; (b) a second level, *Packet level*, that, starting from the results of the previous level, generates the final dump commands. The abstraction allows us to focus on the *dominant* aspects of the problem, i.e. data quantities, packet stores capacities, and dump capability over the communication windows, without considering the problem of actually generating the dump commands. It is worth remarking that the packet level step can be automatically accomplished once a solution for the Data Dump level is computed (no failure is possible) so, in rest of the paper, we focused on producing solutions for the Data Dump level.

### A bimodular architecture

Given the goal of developing a decision aid for supporting the human mission planner in solving MEX-MDP problems we have chosen to design a software architecture that captures the entire problem *life-cycle*. In particular the tool supports a user in all the steps which go from the definition of a MEX-MDP instance to the generation of solutions and refinement. In the current work practice at ESA the user (mission planner) and the space system as a whole interact through certain modalities. One of the goals of our study has

been to contribute an additional means to this interaction offering a tool that preserves completely the “traditional” real world practice and potentially provides new aids. Indeed the assumption was made that all the interfaces as defined for the “semi-manual” dump generation tools previously used at ESA, should be maintained in order to guarantee a smooth transition to the new system. Nevertheless, a goal was pursued to provide a more advanced interactive system that relieve the human planner from boring and repetitive tasks while allowing her to concentrate on more strategic decisions.

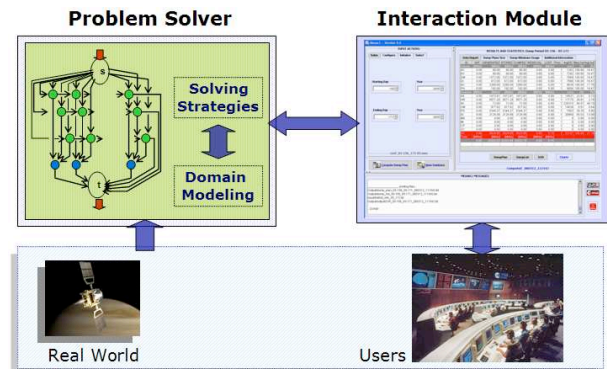


Figure 2: The interactive problem solving architecture

A software architecture for interactive problem solving that copes with this problem is sketched in Figure 2. The figure shows our general approach that consists in adding a path from the user to the controlled spacecraft. This enhanced path is created through a bipartite architecture composed of a Problem Solver and an Interaction Module. The two modules have distinct roles:

- *Problem Solver*. This module is responsible for modeling the problem and the domain. Based on a CSP (Constraint Satisfaction Problem (Tsang 1993)) approach for the modeling phase, it provides a hybrid solving procedure that combines backtracking with a max flow algorithm to solve MEX-MDP instances.
- *Interaction Module*. This component is responsible for the dialogue with the user. It supports a users in understanding what the solver is doing, improving her trust in the automated solving activity and providing various levels of intervention for strategic decisions during problem solving.

A key aspect of the authors’ approach is to build an AI-based representation (or *model*) of the domain that contains the relevant objects to describe the MEX-MDP problem features. This representation phase is fundamental because a good modeling choice not only supports the solving algorithm but creates a basis for the interaction with the user (so it is the core of the architecture in Figure 2).

It is worth reminding that the approach based on developing a model of the domain is widely used in knowledge

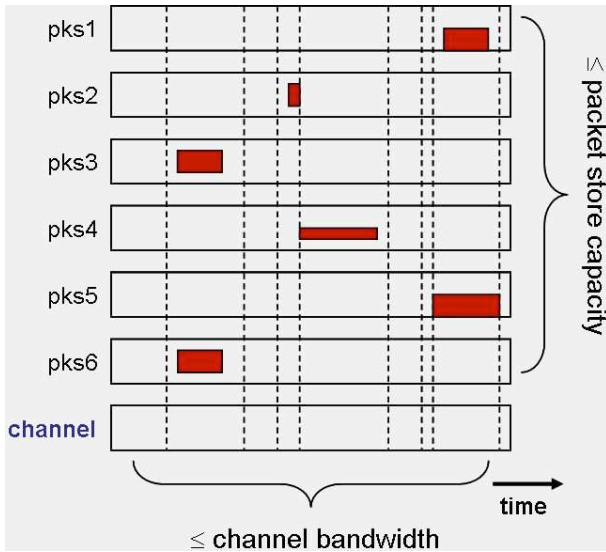


Figure 3: A computational model for MEX-MDP

intensive approaches to Planning & Scheduling like in ASPEN (Chien *et al.* 2000), RAX-PS (Jonsson *et al.* 2000), COMIREM (Smith, Hildum, & Crimm 2005), O-OSCAR (Cesta *et al.* 2001) even though each of these systems gives emphasis to different aspects.

To sum up in this work we have pursued the idea that a user is part of the real world and MEXAR2 endows her with an additional “lens” to analyze the world and act on it. Given this bi-modular framework, the user can concentrate on strategic high level decisions and what-if analysis, delegating to the system repetitive and difficult computations.

In the remainder of the paper, we describe in more details the two modules that compose the MEXAR2 architecture, focusing on both the solving algorithms and the interactive features.

## Modeling and solving

We mentioned before that our approach is based on the synthesis of a model for the aspects of the problem that are relevant to the MEX-MDP (see Fig. 3). In particular we consider the temporal evolution of a specific set of system components, namely the packet stores,  $\{pk_i\}$ , and the transmission channel. In this way we restrict the problem to consider the resource profiles for each packet store and the channel availability for downlink. The core of the problem is to decide the amount of data that are manipulated over time by these key components. A consistent solution synthesizes temporal functions that map data amounts for each of these components.<sup>1</sup>

<sup>1</sup>It is worth noting how the modeling approach is close to the one used in many works in AI P&S literature that approach planning by focusing on *timeline synthesis* (Muscettola *et al.* 1992; Jonsson *et al.* 2000; Chien *et al.* 2000; ?).

---

### Algorithm 1: Generate down-link commands

---

**Input:** problem P, policies parameter robust and priority  
**Output:** Down-link commands Cmd

```

// Data Dump Level
// Produce a data allocation over the
// communication channel
while S incomplete do
  S ← HousekeepingAllocation(P, priority)
  S ← MaxFlowAllocation(P, S, robust, priority)
  if No more HK allocation then
    break
// Packet Level
// Generate data commands
Cmd ← GenerateDownLink(S, priority)
return Cmd

```

---

How to synthesize consistent temporal functions in order to satisfy the main problem constraints given by the packet stores capacity and the channel bandwidth is the problem to be solved. We further refine the model subdividing the temporal horizon in contiguous intervals (called *windows*) such that instantaneous memory operations may happen only at the edges of these windows. Decision variables are then associated to each interval on the packet stores and represent the volume of data dumped within each time slot.

Our first approach to the resolution of the MEX-MDP problem was a specialized Constraint Programming method based on heuristic search and constraint propagation rules – for further details see (Oddi *et al.* 2003; 2005). In a second stage, we introduce a Max-Flow reduction of the problem (Oddi & Policella 2004) and this second algorithm has been actually operationalized within MEXAR2. It is worth remarking the role of the Max-Flow-based procedure. This permits to find, if it exists, a consistent solution given any initial situation (or dump allocation). The initial situation can be empty (no dumps decided yet) or not (some dumps already allocated). Thanks to this fundamental property it is possible to use the algorithm to complete the timelines when a partial solution exists. This property is very important to manage some particular packet stores (like the housekeeping ones) and to give the user the opportunity to fix some dump operations (for instance, a user can decide the dump operation for a given packet store in order to have the data available in a specific time instant).

**Algorithm.** A high level description of the solving process is shown in Algorithm 1. Following the abstraction into levels, it is subdivided in two steps. Goal of the first step (Data Dump level) is to produce a data allocation over the communication channel: for each packet store and for each time window, the amount of data to download is decided. The second step (Packet level), generates the down-link commands starting from the allocation produced in the first step.

As mentioned, the key aspect is the generation of the data dumps. This step is represented in Algorithm 1 by the *while* loop – which is entered with an empty initial solution  $S$ . Each iteration of the loop involves the two procedures:

- **HousekeepingAllocation** that aims to find a consistent solution of the sub-set of the housekeeping packet stores. Indeed these packet stores contain important data relevant for the probe’s safety and, as said before, neither residual data nor preemption is allowed.
- **MaxFlowAllocation** that, given a partial solution produced by the previous step, generates the data dumps from the remaining (scientific) packet stores. This procedure is based on a reduction of the problem  $P$  (included the solution of the previous step,  $S$ ) to a Max-Flow problem: the former problem has a solution when the maximum flow of the latter equates the total amount of data to dump. For further details please see (Oddi & Policella 2006).

The **HousekeepingAllocation** is essentially a backtracking search that, at each step, generates a possible dump plan for the housekeeping packets store. The **MaxFlowAllocation** involves the possibility of preemption, for this reason it is not applied to the housekeeping packet stores.

These two steps will be iterated until a complete solution is found or no further alternative exists (we thus check if there is any allocation available). At the end of the while loop we will have a solution  $S$  that is used to generate the final list of dump commands.

Additionally for all the procedures it is possible to set different parameters. The allocation in **HousekeepingAllocation** can be done considering the priority values of the housekeeping packet stores (parameter *priority*). It is worth noting that because we are facing a real problem we need to always give an answer. In case overwriting cannot be avoided, the solver relaxes the problem and creates a solution, that includes overwriting, according to the relaxation.

The procedure **MaxFlowAllocation** accepts a parameter *robust*. In this case we have an iterative max-flow based search that tries to flat possible peaks in the usage of the storage devices (Oddi & Policella 2006). Broadly speaking robustness copes with the uncertainty on the exact amount of produced data. Sources of uncertainty stem mainly from the unpredictability of the scientific observations’ outcome. For instance, the data volume produced by the High Resolution Stereo Camera (HRSC) – one of the most memory consuming payload – depends on the target and on the context. Thus the impossibility to have an exact estimation of the memory usage of each payload request leads to producing *brittle* solutions. Our aim has been to control the level of memory use in order to avoid possible loss of data due to overwriting. One possibility for overwriting can occur when a greater than expected volume of data has to be stored and there is not enough space in the packet store. For this reason

we define as robust a solution in which a specified amount of space of each packet store is preserved in order to safeguard against overwriting. In other words, solutions robustness is related to the concept of distance to the overwriting state.

Finally **GenerateDownLink** can consider the packet store priority values. In fact the input of this procedure consists of a set of data dumps for each time window. These are allocated in the interval but are not ordered, therefore they are ordered according to the priority values.

## Interacting with the user

The MEXAR2 Interaction Module has been designed with the aim of (a) reproduce the usual problem solving cycle of real users, collecting in a unique tool all features and functionalities to guarantee their traditional problem solving style; (b) allow to exploit the domain expert abilities and foster her involvement and contribution to the problem solving.

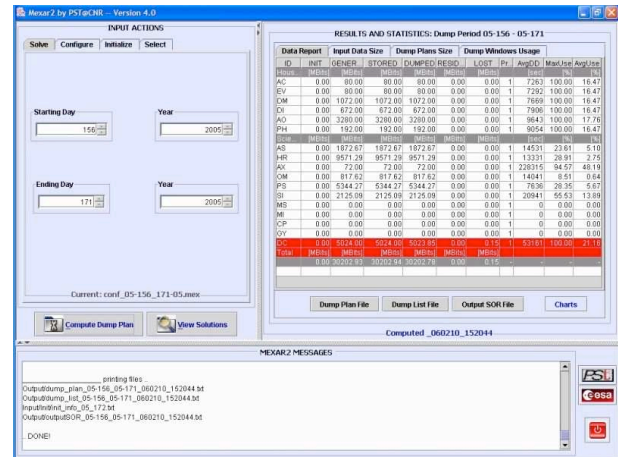


Figure 4: MEXAR2 interaction module

The main interaction layout (see Fig. 4), reflects immediately the dialogue with the user: (a) the channel from mission planners to MEXAR2 is implemented through the pane called “Input Actions”; (b) the communication from MEXAR2 to the users is implemented mainly through the pane “Results and Statistics” that provides different output and secondarily through the pane “MEXAR2 Messages” that provides messages during each computation.

**Input actions.** The basic action consists in the specification of the dump interval, the starting and ending dumping days – e.g, Starting Day 156, 2005; Ending Day 171, 2005 in Fig. 4.

Then, through three sub-panels, users communicate to MEXAR2 additional information. In particular (a) the “*configure*” action allows to specify the domain model, e.g., the specification of packet stores, their size, their respective priorities, their need to be robustified, etc.; (b) the “*initialize*” action communicates to MEXAR2 the residual memory stores from the previous temporal interval that are to be

downloaded together with data related to new PORs; (c) “selecting” through this pane users can tune the parameters of the algorithms.

**Automated solution.** Once the needed context and the required parameters have been specified, users may ask the solver to compute a dump plan by clicking the “Compute Dump Plan” button. Feedback about the solver decisions is provided through the MEXAR2 Messages pane. As mentioned MEXAR2 provides different solving algorithms. A basic solving procedure looks for solutions without considering the packet stores priorities. A second one, takes into account the priorities, while a third algorithm aims at obtaining robust solutions. The pursued idea of robustness is the one of avoiding that a single memory bank is extremely full of data so that any variation at run time is difficult to be absorbed with consequent high risk of overwriting. An additional aspect that the algorithms have addressed concerns the “fragmentation” of the whole dump plan. In this respect the user can specify different thresholds, called input and output thresholds, that avoid production of commands too short or that dump too little data.

**Inspection of results.** Once the solver has found a solution, the Interaction Module shows results exactly as in Fig. 4. A set of statistics are shown in the “Results and Statistics” pane that provides aggregate information on the current solution. In particular a panel named “Data Report” is shown, with entries subdivided into packet stores. This report gives the user an immediate view of the input/output data balance. Apart the Data Report table, the basic pane allows to access the three main files that are produced by the solver through three buttons (“Dump Plan”, “Dump List”, “Output SOR”). Recently ESA provided us a Visualization tool, developed in-house to validate results of MEXAR2, see Fig. 6, that has been inserted as an additional plug-in.

**Computing different solutions.** As said before we have worked to endow the system with functionalities that support the user in exploring different solutions. The mission planner can indeed generate different *use profiles* of the tools acting on the “configure” and “select” features and compare various results. To facilitate this activity we have also introduced an additional structure called *Solution Data Base* that, for any time interval, shows a table with all the solutions computed for the period. For each solution in the table, users can see the settings used to obtain it and a summarization of different metrics for evaluation. In Fig. 5 there is a sketch of the solution data-base pane. In particular, Fig. 5(a) shows a set of stored solutions, for each of which three kinds of information are shown: 1) a solution id; 2) the heuristic strategy used and its parameters; 3) the values of the adopted quality metrics. In addition, each line of the table works as a reference to the stored solution. By click on it it possible to visualize its Data Report table (see Fig. 5(b)).

It is worth commenting on the different role played by the Interaction Module in the two steps of our work. The MEXAR prototype from the first study (Cortellessa *et al.*

SEL	ID	CONFNAME	ORE	PRIT	RBS	MCD	LOST	SIZE	WTT_LHR	WTT_SC	USE	
	051205_154804	Conf3_max	4	p	-	-	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190306	conf3_max	8	p	r2	d30	0.00	9.00 (-9.00)	161786.67	338935.00	100.00 (-5.39)	
	051205_190318	conf3_max	4	p	-	-	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190338	conf3_max	4	p	r11	-	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190482	Conf3_max	4	p	r11	-	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190455	Conf3_max	4	p	r11	-	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190516	Conf3_max	6	p	r11	d30	0.00	9.00 (-9.00)	161766.67	338935.00	100.00 (-5.39)	
	051205_190531	Conf3_max	6	p	r11	d30	0.00	9.00 (-9.00)	161786.67	338935.00	100.00 (-5.39)	
	051205_190552	Conf3_max	9	p	r11	d30	0.00	9.00 (-9.00)	161786.67	338935.00	100.00 (-5.39)	
	051205_190603	Conf3_max	9	p	r11	-	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190659	Conf3_max	9	p	r7	d3	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190807	Conf3_max	9	p	r7	d3	0.00	9.00 (-9.00)	162056.67	341185.00	100.00 (-5.39)	
	051205_190841	conf_21-05_45-05_h	9	p	r7	d3	0.00	9.00 (-9.00)	82601.66	340115.00	100.00 (-5.27)	
	051205_190858	conf_21-05_45-05_h	9	p	-	-	0.00	9.00 (-9.00)	82601.66	340115.00	100.00 (-5.27)	
	051205_190935	conf_tx_priorities_max	9	p	-	-	0.00	9.00 (-9.00)	82601.66	340115.00	100.00 (-5.27)	
	051205_190949	conf_tx_priorities_max	9	p	r10	d36	0.00	9.00 (-9.00)	82471.16	340130.00	100.00 (-5.28)	
	051205_191021	conf_tx_priorities_max	9	p	-	-	d36	0.00	9.00 (-9.00)	89042.18	338935.00	100.00 (-5.39)

(a) Solution data-base

ID	INIT	GENER	STORED	DUMPED	RESIDU	LOST	Priority	AvgOD	MaxUp	AvgUse
AC	0.00	80.00	80.00	80.00	0.00	0.00	1	7293	100.00	16.47
EV	0.00	80.00	80.00	80.00	0.00	0.00	1	7292	100.00	16.47
OM	0.00	1072.00	1072.00	1072.00	0.00	0.00	1	7698	100.00	16.47
DI	0.00	672.00	672.00	672.00	0.00	0.00	1	7906	100.00	16.47
AO	0.00	3280.00	3280.00	3280.00	0.00	0.00	1	9843	100.00	17.76
PH	0.00	192.00	192.00	192.00	0.00	0.00	1	9554	100.00	16.47

(b) Inspecting a solution

Figure 5: Supporting solution exploration

2004) made a larger use of graphical interaction to show the internal structure of the symbolic model the solver was working on. The current MEXAR2 version is rather oriented toward serving the different features of the problem solver. During the first study the role of the interaction has been devoted to convince users to trust not only the solver but the AI approach as a whole. In the current operational tool, the interaction pursues a more “conventional” role of access facilitator.

## Supporting continuous operations

The current procedure for synthesizing dump plans with MEXAR2 in the loop is shown in Fig. 6. We first observe that MEXAR2 accepts as input directly the POR requests from the science plan, and the specification of the downlink windows from MPS (Mission Planning System). It produces as output the dump plan in the three formats expected by ESA people. This has been obtained by encapsulating the solver between two software modules: a first one responsible for processing the input files and selecting the relevant information for the symbolic model used by the solver, and a second one for generating the output according to external formats.

A second observation concerns the change of perspective offered to the mission planners. Users do not directly iterate in the attempt of “almost manually” producing the dump plan but rather establish a dialogue with MEXAR2, having

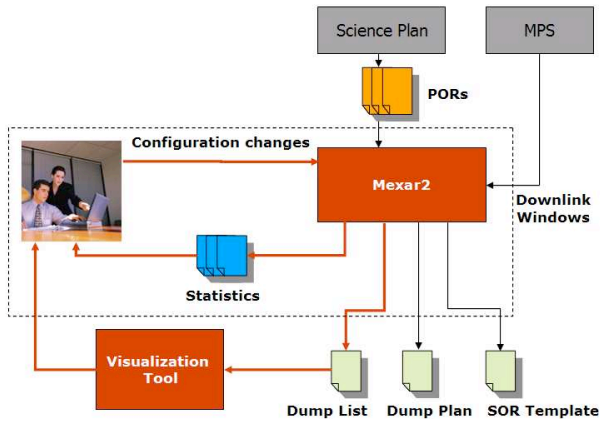


Figure 6: Mixed-initiative dump plan synthesis

access to the model of the domain and dials to tune the algorithms. From MEXAR2 they receive also additional information, for short referred to as Statistics in Fig. 6. This information enrich their ability to analyze the current solution. In general we have pursued the goal of allowing users to take more strategic decisions, and maintain control over the synthesis of *dump plans*, delegating to MEXAR2 not only the repetitive part of the work but also the proactive role of creating the dump plan. This has been achieved by exploiting MEXAR2 ability to work quickly and to consider clues given by solving strategies. For more details on the different modalities of work before and after the introduction of MEXAR2 see (Cesta *et al.* 2006).

### MEXAR2 evaluation

The MEXAR2 evaluation is presented considering three main aspects: the performance of the automated Problem Solver, the effectiveness of the interaction Module and the main operative results of the overall tool.

#### Evaluating the problem solver

As mentioned before, the main concern of the Mission Planning System team is to avoid as much as possible potential overwriting (which corresponds to data loss). Therefore the quantity of data *lost* can be considered as the main *quality measure* for evaluating solutions produced by the automated algorithms. A second important aspect that characterizes the quality of a dump plan turned out to be its *size*. Indeed, the time needed to upload the dump commands to the spacecraft (during the subsequent phase called uplink), is proportional to the size of the dump. For this reason a small dump plan size is to be preferred to a longer one. As additional quality measures we also mentioned plan *robustness* and a plan's adherence to the *priority* scheme. However a complete report on the evaluation of all these aspects is out of the scope of this paper. This section focuses on the main important aspects which correspond to the first two parameters: the data *lost* and the plan *size*. The remainder of this paragraph describes the benchmarks we used for the experiments and the

performance of the proposed algorithm with respect to the chosen quality measures.

**Test data and settings.** We report the system performance on four sets of data, respectively labeled B1, B2, B3 and B4, which refer to different interval of real mission days within the first six months of 2005. Each period is represented by the format  $[start\text{-}day\ year, end\text{-}day\ year]$ . We have: B1: [014 05, 045 05], B2: [101 05, 120 05], B3: [142 05, 156 05], B4: [156 05, 171 05]. The four proposed benchmarks involve quite different values for the volumes of generated data in comparison with the overall communication channel dump capacity. Table 1 shows the duration of each input benchmark as number of mission days ( $N.Days$ ) and compares the total volume of data generated ( $VG$ ), in the interval of mission days with the maximal dump capacity of the communication channel ( $DC$ ) in the same period (ratio  $VG/DC$ ). Notice that the benchmark *B1* is the most critical and is the most probable candidate to lose data during the execution of the related dump plans.

**Quantitative performance.** We report here a set of quantitative evaluations for MEXAR2 by using two different input parameters settings on the test data of Table 1. The two settings are labelled *S1* and *S2*. All the tests were executed with a Java2 Runtime Environment, Standard Edition (build 1.4.2.07-b05), on a SUN Ultra 10 with an Ultra Sparc III 440 Mhz processor and 256 MByte RAM, under Solaris 7. The following data are reported in tabular format (see Table 2) for the evaluation: (a) Volumes of data stored (*Stored*), dumped (*Dumped*), residual in the packet store at the end of the dump period (*Residual*) and lost (*Lost*) (in [Mbits]); (b) Number of stores (*NStores*). It represents the total number of memory stores considered by the system as input data and it represents the size of the input data which can be directly related to the complexity of the solving algorithm, that is, its computational time and memory requirement. (c) The average plan size (*AvgPlanSize*) and the maximal size of a daily dump plan (*MaxSize*). We define the size of a daily dump plan, as the number of dump commands contained in a plan (included the commands for the housekeeping packet stores). (d) The total computation time (*Time* in seconds) to find a solution given an input benchmark set.

Table 2 compares the performance of MEXAR2 with respect to *S1* and *S2*. The main difference between setting *S1* and *S2* are the so-called *threshold* values imposed on the reading of the input data records from the scientific instruments. In few words, each scientific observation, the POR, produces a sequence of data records (called ZD records) which represent the input stores for the solving algorithms described in the previous sections. A threshold value can be set for each packet store and represents a given percentage of its overall capacity. By setting a threshold to a value  $T$ , a sequence of many small data records targeted on the same packet store are grouped into a single cumulative record of size  $T$  and stored at the end of the grouped sequence.

Table 1: Volume of data generated [Mbits] vs. overall channel dump capacity [Mbits]

Benchmark	N.Days	Volume Generated (VG)	Dump Capacity (DC)	VG/DC
B1	31	78323.4	98268.1	0.80
B2	19	71938.6	114566.1	0.63
B3	14	22763.0	125516.2	0.18
B4	15	30747.2	155412.7	0.20

Table 2: Performances on S1 and S2 data. S2 differs for additional threshold settings

	Benchmark	Stored	Dumped	Residual	Lost	Nstores	AvgPlanSize	MaxSize	Time (secs)
<b>S1</b>	B1	78323	78323	0	0	1534	17.8	57	209
	B2	71938	71938	0	0	1314	22.9	32	323
	B3	22763	22763	0	0	605	16.7	33	7
	B4	30747	30747	0	0	850	23.7	32	22
<b>S2</b>	B1	78323	78323	0	0	838	17.3	34	22
	B2	71938	71938	0	0	688	22.5	33	12
	B3	22763	22763	0	0	334	16.2	29	4
	B4	30747	30747	0	0	399	20.1	27	8

(a) many small data records are grouped into a single one and dumped by a single command. In fact, if we observe S2, within the most critical input benchmark, B1, the longest plan is reduced from 57 to 34; (b) the number of input stores is also rather reduced (from 1534 to 838), as a consequence, the computation time is also reduced. As an example, for B1 the computational time is reduced from 209 seconds to 22 seconds. The use of thresholds in the packet stores is an example of the practical mechanisms that MEXAR2 has provided to the mission planners for tuning the input data and changing the behavior of the solving algorithm. This is a key aspect for allowing the users with the ability of exploring the solution space inserting hints from their experience in choosing the set of solving parameters.

### Evaluating the interaction

In order to facilitate the use of the system and obtain users' trust, we paid particular attention to evaluate user-system interaction since the early stages of the tool development. Indeed, the success of MEXAR2 is also due to an iterative and incremental work on intermediate versions. Receiving continuous feedback from the users has been fundamental to fully understand their requirements and needs and develop incremental versions increasingly accurate and reliable. In line with this iterative design and implementation approach, we decided to test the usability of the interaction module since the early steps.

**Evaluation method.** A *heuristic evaluation* method has been adopted to evaluate the intermediate prototype. This method (Nielsen & Molich 1990) allows to find usability problems in a user interface design so that they can be attended to as part of an iterative design process. Heuristic evaluation entails having a team of evaluators examine the

interface and judge its compliance with recognized usability principles, also called *heuristics* (see (Nielsen & Molich 1990) for a description of the heuristics considered).

From a practical point of view the heuristic evaluation consists of different usage sessions of the interface by usability experts or/and system developers during which both the static aspects of the interface (layout, labels, buttons etc.) and the dynamics (interaction patterns, diagnosis, etc.) are evaluated with respect to the heuristics. At the end of each session experts reason about the results so as to synthesize the main usability problems of the tool and eventually agree on possible solutions to the problems. The aim of this evaluation has been to verify the adherence of the interface design to the main principles and guidelines of usability.

**Results.** The preliminary usability test based on the heuristic evaluation highlighted several usability problems and also some good features of the interface. The most relevant problems were related to the "visibility of system's status" (e. g., lack of feedback to users), or to a not completely satisfactory "match between the system and the operational environment" (e. g., terminology system-oriented, relevant input dialog missing etc. ), and to the lack of "help and documentation" (e. g., lack of help dialogs to guide the user while providing the input). All problems that received a high severity rating have been resolved and introduced in the subsequent releases of the tool. A subsequent Thinking Aloud test have been also carried out with the real users of the system on the final version of MEXAR2. The final test confirmed the general effectiveness of the interaction. Additionally a final interview to real users has highlighted their satisfaction and also a number of important benefits derived from the use of the overall tool. In the remind of this section the main operative results of MEXAR2 are described. For a

complete description of all the advantages derived from the use of the MEXAR2 please see (Rabenau *et al.* 2006).

## Main operative results of MEXAR2

In evaluating the main results obtained with MEXAR2 we consider two perspectives, namely the advantages for mission planners and the advantages for science management.

With respect to MPS a considerable advantage stems in the mixed-initiative style of the tool. The role of software which supported the decision making in previous practice was really secondary and relegated to constraint checking tasks. On the contrary MEXAR2 is proactive with respect to plan synthesis while fosters, at the same time, human intervention and control during problem solving. In general data dump generation can now be performed quicker and with less effort and the quality of plans exceed that provided by the tools previously used. The problem of uncertainty in data production is addressed very satisfactorily from the user standpoint and the time saved for producing plans has been estimated to 50%.

The ability of MEXAR2 to generate plans over multiple days very quickly, allows mission planners to consider alternative solutions in order to avoid data loss. It is worth noting that before the introduction of MEXAR2 in the Mission Planning System, mission planners were satisfied with a single solution, while the tool guarantees the possibility of performing optimizations. In fact, this allows to identify problematic intervals in the near future and negotiate with scientists alternative allocation of involved Payload Operation Requests thus minimizing overwrites.

The science community benefits from MEXAR2 as well: data from observations are available earlier and loss of data minimized. As already said potential overwrites can be quickly detected and fed back to the science community for PORs update. Thanks to MEXAR2 the use of the downlink channel is optimized and more data can be downlinked, thus increasing science return from the mission.

## Conclusions

This paper has described the main outcome of the MEXAR project that started from a research study for a specific open problem, in a space mission, usually solved with a lot of daily human effort, ended up producing an complete tool, MEXAR2, that endows the user with the ability to solve the problem easily and concentrate on high level decision making tasks.

The results of this work represents one of the few examples of problem solving tools based on AI planning and scheduling technology that is successfully being used in the operational phase of a space program. Most of previous examples are from the US space missions led by NASA and JPL (Jonsson *et al.* 2000; Smith, Engelhardt, & Mutz 2002; Ai-Chang *et al.* 2004; S. Chien *et al.* 2005). MEXAR2 is one of the few cases of operational use of such technology within an ESA program.

The authors experience started from the demonstration prototype that captured the problem features but was not tested on real data due to their unavailability. This preliminary experience has enabled an impressive speed up in the development of the actual operational prototype that is currently in use. It is worth saying that, even though a significant amount of work has been needed to interface the active part of the tool with the real data flow in the ESA mission control center, the path from theory to practice has been pursued completely. In addition, we have been able to transform the initial skepticism of ESA personnel with respect to new technology in huge satisfaction for the real enhancement of the quality of work that MEXAR2 guarantees.

**Acknowledgements.** This work has been sponsored by the the European Space Agency (ESA-ESOC) under contract No.18893/05/D/HK(SC). The authors would like to thank the MARS EXPRESS mission control team at ESA-ESOC for the valuable feedback on their work and in particular Erhard Rabenau and Alessandro Donati for their continuous assistance.

## References

- Ai-Chang, M.; Bresina, J.; Charest, L.; Chase, A.; Hsu, J.; Jonsson, A.; Kanefsky, B.; Morris, P.; Rajan, K.; Yglesias, J.; Chafin, B.; Dias, W.; and Maldague, P. 2004. MAPGEN: Mixed-Initiative Planning and Scheduling for the Mars Exploration Rover Mission. *IEEE Intelligent Systems* 19:8–12.
- Cesta, A.; Cortellessa, G.; Oddi, A.; Policella, N.; and Susi, A. 2001. A constraint-based architecture for flexible support to activity scheduling. In *Lecture Notes in Artificial Intelligence*, N.2175. Springer.
- Cesta, A.; Cortellessa, G.; Oddi, A.; and Policella, N. 2003. A CSP-Based Interactive Decision Aid for Space Mission Planning. *Lecture Notes in Artificial Intelligence* LNAI 2829.
- Cesta, A.; Cortellessa, G.; Fratini, S.; Oddi, A.; and Policella, N. 2006. Software Companion. The MEXAR2 Support to Space Mission Planners. In *Proceedings of the 17th European Conference on Artificial Intelligence, ECAI-06*.
- Chien, S.; Rabideau, G.; Knight, R.; Sherwood, R.; Engelhardt, B.; Mutz, D.; Estlin, T.; Smith, B.; Fisher, F.; Barrett, T.; Stebbins, G.; and Tran, D. 2000. ASPEN - Automating Space Mission Operations using Automated Planning and Scheduling. In *Proceedings of SpaceOps 2000*.
- Cortellessa, G.; Cesta, A.; Oddi, A.; and Policella, N. 2004. User Interaction with an Automated Solver. The Case of a Mission Planner. *PsychNology Journal* 2(1):140–162.
- Jonsson, A.; Morris, P.; Muscettola, N.; Rajan, K.; and Smith, B. 2000. Planning in Interplanetary Space: Theory and Practice. In *Proceedings of the Fifth Int. Conf. on Artificial Intelligence Planning and Scheduling, AIPS-00*.

- 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).
- Nielsen, J., and Molich, R. 1990. Heuristic evaluation of user interfaces. In *Proc. ACM CHI'90 (Seattle, WA, 1-5 April)*, 249-256.
- Oddi, A., and Policella, N. 2004. A Max-Flow Approach for Improving Robustness in a Spacecraft Downlink Schedule. In *Proceedings of the 4<sup>th</sup> International Workshop on Planning and Scheduling for Space, IWSPSS'04*.
- Oddi, A., and Policella, N. 2006. Improving Robustness of Spacecraft Downlink Schedules. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*. To appear.
- Oddi, A.; Policella, N.; Cesta, A.; and Cortellessa, G. 2003. Generating High Quality Schedules for a Spacecraft Memory Downlink Problem. *Lecture Notes in Computer Science* LNCS 2833.
- Oddi, A.; Policella, N.; Cesta, A.; and Cortellessa, G. 2005. Constraint-Based Random Search for Solving Spacecraft Downlink Scheduling Problems. In G Kendall and A. Burke and S. Petrovic and M. Gendreau., ed., *Multidisciplinary Scheduling: Theory and Application*. Springer. 133–162.
- Rabenau, E.; Cesta, A.; Oddi, A.; Donati, A.; and Léauté, T. 2006. Using an Artificial Intelligence Tool to Perform Science Data Downlink Planning as Part of the Mission Planning Activities of Mars Express. Technical report, European Space Operations Centre.
- S. Chien, S.; Cichy, B.; Davies, A.; Tran, D.; Rabideau, G.; Castano, R.; Sherwood, R.; Mandl, D.; Frye, S.; Shulman, S.; Jones, J.; and Grosvenor, S. 2005. An Autonomous Earth-Observing Sensorweb. *IEEE Intelligent Systems* 20:16–24.
- Smith, B. D.; Engelhardt, B. E.; and Mutz, D. H. 2002. The RADARSAT-MAMM Automated Mission Planner. *AI Magazine* 23(2):25–36.
- Smith, S. F.; Hildum, D. W.; and Crimm, D. R. 2005. Comirem: An Intelligent Form for Resource Management. *IEEE Intelligent Systems* 20:16–24.
- Tsang, E. 1993. *Foundation of Constraint Satisfaction*. London and San Diego, CA: Academic Press.