Innovative AI Technologies for Future ESA Missions

María Dolores R-Moreno
Departamento de Automática, Universidad de Alcalá
Ctra. Madrid-Barcelona, km 33,6, 28805 Alcalá de Henares (Madrid), Spain.
e-mail: mdolores@aut.uah.es

James Kurien
NASA Ames Research Center
MS 269-3, Moffett Field, CA 94035
e-mail: James.A.Kurien@nasa.gov

Amedeo Cesta
ISTC-CNR, Italian National Research Council
Via S. Martino della Battaglia 44, Rome, Italy
e-mail: amedeo.cesta@istc.cnr.it

Abstract

In this paper we will focus on identifying the missions main requirements, objectives and future trends in terms of on board and on ground functionalities and tasks. We will make the distinction between 'Robotic Exploration' category and the 'Planetary Orbiter' category.

The analysis of where autonomy can be applied which may prove critical. It deserves a special attention in order to enhance mission safety, system overall reliability and scientific and operational returns.

We compare recent directions in AI research using planning and scheduling, intelligent execution and model-based diagnosis as examples. We attempt to synthesize trends in both spacecraft operations and the use of AI technology, and suggest how they might impact future research and its adoption by missions.

1 Introduction

Space science missions have continually become more ambitious, from early planetary flybys, to orbiters, to recent long duration surface rovers, to future missions to fly in alien atmospheres or descend through alien ice caps. Future goals we can mention include further exploration of the Moon, Mars and elsewhere in the solar system, understanding the Earth-Sun connection, and revealing the structure and evolution of the universe. These future science missions will require of increasingly capable systems to achieve the goals they are designed for. In some cases these capabilities can be met by adapting or further developing existing technology, but in other cases new innovative technology should be injected in order to exploit science opportunities and reduce overall operation costs.

The ability to modify and validate a predefined sequence of operations for a science mission quickly, perhaps even on a daily basis for some missions, is generally regarded by scientists as a highly desirable requirement. This feature is known as adaptability. It measures how well the system responds to changes in requirements or initial assumptions in terms of ease of response and efficient optimization of all the resources when new actions are considered. Maximizing the number of science experiments is a priority goal for the scientists involved in the missions.

Another important feature is the ability to manage uncertainty during the mission. Future space missions should have the capability to detect and recognize science opportunities and hardware malfunctions onboard. These missions should also have the capability to respond autonomously to these events by deciding a safe mission state, or planning and carrying out observations to capture short-lived science events and rare phenomena based on previous criteria set by humans. Although the maturation of technology could allow to manage different levels of uncertainty, it is not clear if the final users would like that an intelligent system takes on scientific decisions, as we will discuss in next section.

Although a high degree of automation of science-related mission decisions may not be desirable, commanding spacecraft at a higher level of abstraction where much of the sequence development is done by the flight software, could enormously reduce ground operations costs. Safe decisions could be taken without waiting for the on-ground engineer team, what it is very important in case of long delays as it occurs with Mars exploration.

The paper is structured as follows. Next section introduces where to apply autonomy in the mission systems functions and which level is really needed. An introduction to AI techniques is next explained. Then, future trends in AI technology are envisioned. Finally, conclusions are outlined.

2 What Level of Autonomy is Needed?

A system is Autonomous if it can achieve its goals without or little human intervention. There exists a few number of automated ground systems, although work continues toward the goal of reducing operations costs to even lower levels, but on-board automated systems are still very limited. The need for Autonomy in ESA has been anticipated by the ECCS Space Segment Operability standard ECSS-E-70-11 [34] which defines three levels of autonomy: the execution of pre-planned missions operations on-board, the execution of adaptive mission operations on-board and the execution of goal-based mission operations on-board

The scope of autonomy comprises the major space system functions. Depending on the type of the mission, Planetary Orbiter (such as Earth Observation or Space Observatories) or Planetary Exploration, some functionalities may not be available. Extreme environments, such as the surface of planets, pose special challenges for autonomous robots. The robot must navigate in such environments avoid colliding with obstacles, such as rocks, and avoiding falling into a pit that would cause it to tip over. The basic categories that we have identified where autonomy can be applied to space system functions are: On board diagnosis and reconfiguration, Navigation (depending on the mission), Planning/Scheduling and Intelligent Execution, Command Sequence Generation, and Data handling, validation and processing.

Based on the output of the onboard science algorithms, the autonomous mission will replan its activities in order to capture high value science events. This new observation plan will then be executed by an executor system, capable of dynamically adjusting the plan in order to achieve the goals despite run-time anomalies and uncertainties. Increasing the level of autonomy on the Planetary Exploration will allow not to interrupt the scientific operations, so improving the overall scientific return of investment. But although this is desirable from the technical point of view, scientist may no agree with that. From the experience gained with the interaction with the scientists team at the operation center in Tucson (Arizona) for the Phoenix Mars Lander mission, we have realized that scientist don't want that a planner system controls the schedule of their experiments. Two of the instruments dedicated to the chemical and geological experiments, TEGA (Thermal and Evolved Gas Analyzer) and MECA (Microscopy, Electrochemistry, and Conductivity Analyzer) where most of the time the center of the debate. Putting dust on MECA before or after for example taking images, would not be optimal from the technological point of view but it does make sense from the scientist point of view. But in other cases, using the planner to optimize the pictures taken by the SSI (Surface Stereo Imager) instrument to a target (i.e. if there is dust in one of the pictures, then do more pictures so the planner will reschedule the pictures that have to been taken at the beginning) can be very beneficial.

So sometimes although the degree of maturity for some techniques can be considered high, it cannot substitute the human expertise. Though, case-based reasoning techniques could be used to learn from the human decisions and integrated them with planning systems.

3 A Selection of AI Technologies

Artificial intelligence (AI) is one of the computer science areas that has generated more expectation in the last years from the point of possible impact on space applications. Broadly speaking, AI is a scientific discipline which tries to operationalise human intellectual and cognitive capabilities in order to make them available through information processing systems. In order to do so, two aspects are faced: knowledge representation and reasoning algorithms. The whole area has pursued for a long time the attempt at generating a universal approach to cover both aspects and encountered several difficulties due to the extreme challenge of the main task (reproducing human abilities). Over the last ten-fifteen years a more practical approach is being followed: the subdivision of the field in separated, more focused research sub-areas. In this way several sub-communities developed each sharing a common language. In this way more in-depth results have been obtained with respect to narrower tasks. Somehow the idea is the one of reproducing single cognitive abilities instead of the whole cognition that is underlying the "intelligence". Hence we have sub-areas dedicated to machine learning, planning and scheduling, constraint reasoning, multi-agent systems, etc. The focus on narrow problems have been beneficial for the robustness of achieved results. Each area have obtained huge progresses over the years and produced specific results that are increasingly used in applications. An associated difficulty stays in the fact that for external observers it is more difficult to understand the progresses of the whole field due to the deep technicality behind each result.

In this paper we restrict our attention to the AI work that is more related to the issues of autonomy in space hence we will report on areas like, Planning & Scheduling, Sequence Execution, and Fault Protection that refer to activities that are routinely carried out within space missions. Planning and scheduling refer to the ability to synthesize courses of action (plans) to obtain specific objective over time while taking into account the available resources to perform the single tasks. Usually planning refers to the reasoning for synthesizing

the single steps of the plan while and scheduling refers to the reasoning on the temporal and resource aspect connected to a given plan. Indeed the pure planning and pure scheduling problem are not so frequent in space applications. For this reason increasingly over the last years they have been addressed in as integrated way. A second aspect relevant for space missions is the need to dispatch courses of actions for execution. Here the problem consists in studying how a baseline plan can be executed in all its parts and how all the sources of uncertainty due to the real world can be take into account maintaining the general objectives of the initial plan. There is a need of real time monitoring of execution and of applying strategies for re-planning, plan repair, plan adaptation in case some exogenous events modify the reality. Fault protection, often referred to as Fault Detection, Identification and Recovery (FDIR), is an engineering process that incorporates robustness to faults into spacecraft hardware, software, systems engineering and operations. The on-board system for responding to faults is one output of this endeavor, and typically involves some hardware-level protection and on-board software. Model-based diagnosis systems are based on physical parametric constraints where a generic software engine or set of principles is developed in the hope of addressing a large class of diagnosis problems.

The following three subsection reports more detail on the three identified subareas.

3.1 AI Planning and Scheduling Systems

As said before planning and scheduling in AI have had a separate evolution but in recent years are seen as a unique research area. *AI Planning* studies the causal reasoning needed to change a state of the world to obtain a desired goal state. A planner is an algorithm that given a causal theory that describes what can be done in a domain (also called the domain theory) and a description of world initial conditions and goals synthesizes courses of actions (plans) that achieve the goals satisfying the initial conditions and the constraints given by the domain causality. *AI Scheduling* addresses the problem of organizing tasks in time satisfying a set of resource availability constraints. A scheduling problem consists of a set of activities, a set of resources with bounded capacity, a set of temporal constraints that temporally qualify the activities and an optimization criterion that describe a comprehensive property that the solution should optimize. A scheduler is an algorithm that assigns start times and required resources to the activities, obeying both the temporal restrictions on the activities and the capacity limitations of the resources. It is worth noting that several optimization function are possible: even if the most used is the shortest completion time for the whole schedule (makespan) in space application also load balancing and the schedule robustness are very important metrics.

Modeling Assumptions. Many planning systems have been studied and proposed: basically the only things they have in common is the general task of coming up with a partially ordered sequence of decisions, legal with respect to a set of rules (called a *domain theory*), that will achieve one or more goals starting from an initial situation. However, the analogies do not go further: there have been a lot of differences in the formalism chosen to represent the world in which the planner performs its task, in the general conception about what a plan is and about how the planning process is performed.

In general, it is possible to distinguish two distinct approaches to the planning problem:

- In the first one, called here *action-oriented*, the world is seen as an entity that can be in different *states*, and in which a state can be changed by performing *actions* (*a-la* STRIPS[12]). The domain theory specifies which rules must be followed when actions are performed, i.e., where they can be applied and how the world state is modified as a consequence of the actions. The planning problem is to find a partially ordered sequence of actions that, applied to an initial world state, leads to a final state (the goal) eventually satisfying several conditions about which sequence of states the world must go through (often called *extended goals*). This is the most classical and commonly accepted idea of what planning is.

- In the second approach, named here *timeline-based*, the world is modeled in terms of a set of functions of time that describe its evolution over a temporal interval. These functions change by posting planning *decisions*. The domain theory specifies what can be done to change these evolutions, and the task of the planner is to find a sequence of decisions that bring the entities into a final situation that verifies several conditions. By analogy, we can call these conditions goals. This approach, which is more recent than the more "classical" action-based approach, is evolving rather rapidly due to the fact that it is well suited for modeling and solving non-trivial real world problems. A timeline is a logical structure used to represent and reason about the development of an attribute over a period of time.

It is worth saying that STRIPS has evolved into different languages used by different planners, all integrated in the proposed standard PDDL, actually in its 3.0 version [16]. Notwithstanding this integration effort instrumental for promoting convergence of research results, this language do not have a widespread use in applications, in particular has had a limited impact in space missions although there are some recent notable initiatives [38]. Timeline-based representations have been used in several space mission starting from a proposal for the Hubble Space Telescope [28], then the experiment on autonomy at NASA [27], several other NASA real missions applications such as MER, Phoenix Mars Lander or in MSL through EUROPA [14]. Also ESA has invested on timeline representation they are used in MEXAR2 [5], RAXEM [6] and more recent work described in [7]. It

is also worth mentioning that there is no unified or standard language for AI scheduling systems. Additionally, while it is not particularly natural to model pure scheduling problems in PDDL, the interconnection between scheduling and timeline-based reasoning is well known, in fact several state of the art AI schedulers are naturally built on a timeline representation of resources – e.g., see [8, 18].

Reasoning Techniques for Planning. While research in planning has been historically "system oriented" from the late 60's to the beginning on the 90's, since then there have been significant algorithmic advancements with a level of generality to allow re-use within the area. A number of distinct research results are worth being mentioned. Among the classical approaches to search control it is worth mentioning:

- Partial Order Causal Link planners [37] that search through the plan space. In this space, nodes represent partially specified plans, and edges denote plan-refinements operations such as the addition of an action to a plan. The planning algorithm implements a least commitment approach, that is, only the essential ordering decisions are recorded, there is no need to prematurely commit to a complete, totally sequence of actions. These planners must perform constraints satisfaction to ensure the consistency of the order constraints. In order to avoid interferences between actions, dependencies are recorded in a data structure called causal links. Causal links must be checked periodically during the planning process to make sure that no other action can threaten them. In case a plan contains a threat, some additional ordering constraints can be added to avoid it. It is worth saying that Partial Order Plans Refinement search is today the classica technique used by timelin-based planners like RAX-PS [21], IxTeT [17], EUROPA [14] and more recently OMPS [15].

Somehow the use of partial order plan refinement is considered as the weak point of the timeline approach in term of planning performance. This is because a number of techniques have been proposed for action-based planning that have significantly leveraged the performance on standard competition problems. Let us remember among others:

- GRAPHPLAN-based planners [2] search through a plan graph. They alternate between graph expansion and solution extraction. The graph expansion extends the plan graphs forward until it has achieved a necessary condition for plan existence. The solution extraction phase performs a backward-chaining search on the graph, looking for a plan that solves the problem. If no solution can be found, the cycle repeats expanding the planning graph. The basic idea is to perform a kind of reachability analysis to rule out many of the combinations and set of actions that are not compatible.

- Heuristic Search Planners (HSP) transform planning problems into problems of heuristic search by automatically extracting heuristics functions h from STRIPS encoding instead of introducing them manually [4]. The bottleneck in HSP is the computation in every new state of the heuristic from scratch. It uses a declarative language for stating problems and a general mechanism for extracting heuristics from these representations. The same code is then able to process problems from different domains.

- A SAT-based planner takes a planning problem as an input, guesses a plan length and generates a set of propositional clauses (CNF) that are checked for satisfiability. After the translation is performed, fast simplification algorithms as unit propagation and pure literal elimination are used to shrink the CNF formula. For satisfying assignment, a SAT solver can use Systematic [25], Stochastic [33] or Incremental [26] methods.

It is worth saying that we are referring to the performance of generic domain independent problem solvers. It is worth saying that most of the planners that have been pragmatic effectiveness in application use some form of domain-dependent knowledge to obtain "heuristic adequacy". A family of planner that allow a structured introduction of operators that force/suggest search control knowledge are those using the *decomposition operators* allowing the creation of plans relying on different levels of abstraction:

- HTN [11] planners try to use the knowledge available of the domain to solve the planning problem. This knowledge can be obtained using additional task and goal structures, search control techniques, interaction with humans or different type of constraints. A method maps a task into a partially ordered network of tasks, together with a set of constraints. The basic algorithm is to expand tasks and resolve conflicts iteratively, until a conflict-free plan can be found that consists only of primitive tasks. The task network may contain conflicts caused by the interaction among tasks: after each reduction, a set of critics is checked so as to recognize and solve interactions between this and any other reductions. Thus critics provide a general mechanism for detecting interactions early, so as to reduce the amount of backtracking.

A further set of techniques concerns the synthesis of plans that explicitly consider the uncertainty of the description given to planners:

- Probabilistic planners use probabilities to represent and reason about uncertainty in the planning domain. That is, when an action is selected the probabilities of the possible outcomes are evaluated. Additional actions may be added to construct plans that are likely to succeed even individual actions are uncertain. [3].
- Contingency/conformant planners must be able to produce plans even if the initial conditions and the outcomes of some of the actions are not known but with the ability to observe some aspects of the current world state. A contingent plan is a plan that contains actions that may or may not actually be executed, depending on the circumstances that hold at the time. Since the search space of these planners can be intractable for medium/big problems, they are some approaches that plan by considering only the most problematic action outcomes [13].
- A Markov Decision Process (MDP) expands the notion of uncertainty. It is defined by an initial probability distribution over all possible states of the system being modeled, a distribution that represents the likelihood of the system transitioning from the current state to each other state given an action, and the reward for taking each action in each state. Solving an MDP requires determining the action in each state that maximizes the expected cumulative reward of the system over time. Techniques based on policy iteration or value iteration [32] have been traditionally used. This can be intractable for most applications, so most of the work in this area has focused on using an approximation or abstraction of the state space. Similar work has been done with Partially Observable Markov Decision Processes (POMDP) in which the assumption of complete observability is relaxed. Although work in this area is promising, it has only been used to solve small POMDP problems.

Reasoning Techniques for Scheduling. Scheduling research is an indepented area which have developed an amount of techniques for problems coming from Production Systems, Manufacturing, etc. A complete summary of such techniques is difficult to produce for the sake of space. We just introduce some pointers that influence the current debate in AI scheduling technology. First of all we should consider the influence on the current research of competitive approaches from:

- Operational Research (OR). The most representative techniques are Linear Programming and Sensitivity Analysis. This two techniques allow to find the optimal solution of a function that satisfies a set of linear constraints, in the first case or the consequences in the solution if some changes are made in either the data or in some of the solution values, in the second case.
- Constraint Satisfaction (CSP). A Constraint Satisfaction Problem (CSP) prescribes some requirements for a finite number of variables in the form of constraints. The set of possible values, that is, the domain for each variable is finite. Each constraint specifies a relationship between the values of the variables that most hold. Solving a (CSP) means finding an assignment to each variable such that all of the constraints are satisfied.

It is worth mentioning how most scheduling solvers rely on the effectiveness of the heuristic guidance given to general search procedures. Most of the algorithms used for solving a CSP fall into the two categories:

- Refinement or Constructive Search Methods: a CSP that progresses *incrementally* assigning values to variables, checking the constraints and backtracking when violations appear. The evaluation of global criteria can only be approximated given that there is only a partial schedule.
- Iterative Repair or Local Search Methods: they start with a complete assignment of values to variables and then reassign new values to variables to resolve violated constraints. The evaluation of global criteria is evaluated with low cost. However, one of the main disadvantages is that they can suffer from local minima and they are inherently incomplete.

One of the most relevant open issues in scheduling research regards schedule support at execution time. The dynamism and unpredictability which inherently permeate real-world application domains, make the ability to cope with unexpected events during the schedule execution phase an absolutely primary concern. For instance, the ability to respond automatically (and efficiently!) to unexpected events that occur during normal job-shop floor operations is considered highly desirable. Also this aspect is being studied in several scientific communities, such as OR and AI. Current approaches have tackled the scheduling uncertainty problem according to one of the two following directions (see for example [31, 9] for detailed discussions):

- *proactive approaches* in which effort has been put into the development of methodologies producing solutions which are characterized by a certain degree of robustness, therefore retaining the ability to absorb the effects of exogenous events.
- *reactive approaches* that take into account the fact that the buffer that protects the solution against possible disruptions is inherently limited, hence the need to devise mechanisms to reactively counteract circumstances that fall beyond the current problem boundaries.

3.2 Intelligent Execution Systems

Traditional autonomous control architectures are often based on 3 tiers (sometimes generically referred to as 3T architectures). The lowest tier constitutes the functional layer, that is, the interface between software and the hardware functionality of the spacecraft or other system. The top tier is the AI P&S system that takes several mission goals, finds activities to meet them, and schedules them for execution over an extended period of time, having in mind the resources available. The middle tier is an executive that can run procedures that achieve mission goals as guided by the plan.

Then, an executive is a software component that realizes preplanned actions. Executives are particularly useful in the presence of uncertainty. An executive can be seen as an onboard system that takes the actions of a plan and their expected outcomes (assuming a certain level of certainty) as input and manages their execution in an uncertain and possibly dynamic environment. The technology varies from systems that execute simple linear sequences of commands [35], to complex systems [1] that can plan and schedule in reaction to unexpected changes that endanger completion of the plan.

These systems require a language for the plans, actions and commands they will manage. They may also represent interdependence between actions in terms of precedence or other constraints, in addition to the expected effects of executed commands in order to monitor progress. We can differentiate two types of languages:

- Procedural execution languages are used to develop real world software where loops, branching and parallel execution are needed. Conceptually they specify how the execution should proceed, and thus are often at too low level to be generated by a planner. In this category we can mention VML [19].
- Declarative languages use a more expressive representation of actions that focus on what the actions are rather than how they are to be executed. The intent is for the executive's semantics of execution to interpreting the actions and yield a robust behavior not achieved with traditional execution scripts. This requires significantly more modeling by the user and many constructs such as looping and branching cannot be expressed in a natural way. The Remote Agent Exec [30] uses this type of representation.

Although the literature identifies three categories of executive systems [36], just two have been really used in real missions:

- Execution-only systems: The system accepts actions or commands for execution, but does not explicitly exchange information with an automated planner. The simplest approach is direct commanding where commands are sent via radio from mission operators on the ground and executed immediately. For deep space missions, this appropach is not appropiate due to light-time delays. Instead, it can be used time tagged commands to send a sequence of commands each with a time tag all at once. At the appropriate time, each command is executed. Greater flexibility is afforded by event-based sequences, where timing of commands may be relative to external events, calculated durations or completion of other commands.
- Execution systems coupled with an external planner: the systems have explicit interfaces to planners through an execution language or a standard plan format. They provides a representation that captures control constructs and can also be projected by a search-based planner. They intend to close the gap between the declarative action model used by a planner and the procedural languages used to develop real-world software [24].

The third approach is still in an inmature phase, that is, the Execution systems with internal planners. The systems integrate planning and execution more tightly by using a planner internally within the execution system to select control actions. The disadvantage of using planning from first principle at every execution cycle is that patterns of constraints (temporal and parametric) are always assembled from scratch, causing higher latency than possible when using pre-compiled execution scripts. Consequently, execution may halt if the planner can't deliver a response in time. A representative of this approach is IDEA [1].

3.3 Fault Protection Systems

The primary purpose of fault protection is to ensure that anomalies or operational problems encountered during operation of the spacecraft do not result in permanent reduction in the spacecraft's capabilities or loss of the mission itself. It is an engineering process [29] that incorporates robustness to faults into spacecraft hardware, software, systems engineering and operations. The on-board system for active fault detection, isolation and recovery (FDIR) of possible faults is one output of this process. We note though that the on-board system is just one aspect of the overall fault protection engineering process, and that the on-board system for protecting the spacecraft is typically a mix of hardware and software. Traditionally, the fault protection engineering process is driven by fault modes, effects and criticality analysis (FMECA). This process typically determines the possible faults of a system or subsystem, some notion of their likelihoods, and an analysis of the impact of each. If the likelihood of a fault is deemed sufficiently high and the impact sufficiently negative, the analysis would also include how the fault would be detected and what the appropriate response would be. For most

spacecraft, at least a portion of the FDIR system is in hardware. We will focus on on-board FDIR software, keeping in mind it is only a portion of the overall fault protection system for a spacecraft.

The appropriate response to a fault may depend upon the phase of the mission. We use the term *critical phase* of a mission to mean periods of a mission where the spacecraft must take specific actions (a *critical sequence*) or loss or serious degradation of the mission will result. We say the system must *fail operational* if any failure that occurs must be taken into account by the fault protection approach, for example by switching between redundant subsystems, to allow execution to continue. Accordingly, during development of the spacecraft an enormous amount of attention is given to the precise critical sequence the spacecraft will execute, which failures are likely, how they will be detected, and how they will be immediately mitigated. Since response must be timely, it must be available on-board. This may involve something as simple as a table mapping observed sensor values to commands that should be issued in response, for example to switch to a redundant backup.

In a *non-critical phase*, it is still possible to damage or lose the mission due to a fault, but there isn't the added constraint of having to execute a critical sequence. Thus typical fault protection systems tend to focus on mitigating or postponing the impact of the fault by changing the spacecraft's state or behavior. Often if a serious problem is detected, the spacecraft is placed into a *safe mode* where the only actions taken are those that maintain the spacecraft in a quiescent state.

The approach of safing the entire spacecraft or select subsystems when faults are suspected has the advantage that one need not know exactly which fault is causing the operational problem. So it aims to carefully engineer the robust but minimal fail operational capabilities needed for critical periods, and otherwise engineer hardware, software and operations so the spacecraft can be put in a safe state in response to plausible anomalies. This characterizes a wide range of fault protection systems that have been flown.

Model-based diagnosis are diagnostic systems arising from Computer Science or Artificial Intelligence approaches where a generic software engine or set of principles is developed in the hope of addressing a large class of diagnosis problems [10]. Later, models that adapt the generic engine to a specific diagnosis problem are created. The diagnostic engine is given the model and fed observations from sensors on the mechanical system being diagnosed, and is intended to automatically infer which components of the system are failed, and perhaps in what manner. The particular representation and algorithms used are beyond the scope of this paper, but an excellent survey of the seminal work in this field can be found in [20].

The Livingstone system [22] builds on this work, and is meant to monitor execution of commands, diagnose failures and provide recovery actions for complex systems such as spacecraft.

During operation, Livingstone is fed the stream of commands that are being given to the spacecraft and the readings from sensors that relay the internal state of the spacecraft (e.g. switch status bits or temperature sensors). Livingstone uses the model of the spacecraft's components and the command stream to predict the values of the sensors that should result from the commands assuming no components are failed. If there is a discrepancy between the predicted and observed sensors, then a failure is assumed. Livingstone uses the model of the components to simulate different combinations of failures. It uses a diagnosis algorithm to quickly focus on the combination of failures that would predict the sensor values that are being observed. This combination of component failures is the diagnosis. In case of failure, Livingstone is able to use the same predictive model to suggest commands that achieve a desired property (e.g. engine is receiving fuel) and thus mitigate the failure.

4 Possible Trends in Selected Technologies

Once investigated the present status of AI techniques for space domains, we can discover possible trends which might be useful in improving future missions. We will focus just on the 3 main areas mentioned in the previous section. But it is worth mentioning that machine learning, i.e., case-based reasoning techniques could be integrated when planning the experiments to learn from scientist decisions. Genetic Algorithms can be used to optimize routes in the Navigation functionality. And data mining techniques can be used to pick up the relevant information from the data received.

Planning & Scheduling. One recurring theme is the need for planning software and users to generate plans in a mixed-initiative (cooperative) fashion. Traditionally, the planning problem is posed as accepting a set of goals and automatically generating a feasible plan. In many cases, this breaks down for several reasons. Firsr, the initial goals often oversubscribe the spacecraft's resources. Users need to explore and negotiate which goals to remove rather than having the planner decide. Second, in some sense every planner model of a realistic domain is incomplete. In practice a valid plan may be deemed infeasible by the planner due to circumstances the author of the model did not foresee. If a set of goals are not achievable due to oversubscription or violation of some aspect of the domain constraints, it's important for the user to examine infeasible plans, view plan flaws, and determine which activities to remove or what other repair strategy to consider. Third, during operations, domain modeling bugs are discovered and mission managers decide that in certain cases a constraint in the domain model should be ignored. Finally, although the plan could be feasible and optimal, experiments may be carried out in a different other for scientist propose.

Thus the planner must be able to work with infeasible plans and users must be able to ignore changes or advice the planner is providing, investigate plan flaws, and quickly add new operating rules or relax others for special circumstances.

We also believe scheduling is the bigger issue than planning in the traditional sense. To achieve a goal, there are not typically lengthy sequences of actions with many alternatives. Instead, there are many independent goals that are easy to achieve individually, but interact with each other in complex ways through the resources limitations and operating constraints.

Finally, the ability of handling uncertainty is important. It would be useful to have plans that have some handling of uncertainty but that are easily generated, understood, and verified, such as a small number of contingency branches. Also when a plan is executed and the goal cannot be achieved no matter the state the mission is in, achieving other secondary goals can be very helpful. The only problem is how to decide (in based on what) the secondary goals to achieve.

Plan Execution. Concepts from execution research have been transitioning to practice, but fielded systems tend to favor simplicity, focusing on monitoring of execution rather than dynamic rescheduling or other responses to execution problems. Executives are usually script languages with simple iterative sentences as loops or branching. Estimating bounds on resource usage and duration, often through simulation, is important for spacecraft, and eased by simpler semantics. In practice, uncertainty about real-time execution is often handled with conservative resource and duration margins. If a plan executes more quickly than expected, the spacecraft may wait for further instructions. On the MER rovers "bonus activities" are appended to the end of a plan but the expected duration is not increased. If the plan executes quickly, bonus activities are executed until the expected duration expires. Also it would be interested that executor can also handle some degree of uncertainty. Missions are also carrying increasingly complex instruments, often with their own processors, scripting languages and conditional execution. This trend only accelerates the need for simple synchronization constructs to coordinate the operation of multiple independent operations. But, they do not have in mind mission goals or automatic synchronization. There is a tendency to develop more expressive languages but at the cost of gaining complexity for the final user. We believe future mission will work in that direction.

Fault Protection. Model-based diagnosis was demonstrated on ds-1 and eo-1, along with a planning system, as well as on testbeds and simulators for the X-34 and X-37 vehicles and many other systems. Yet to our knowledge, no spacecraft has flown this type of general purpose diagnosis and recovery engine as part of a baseline fault protection system. We believe there are relatively simple explanations. One hope for systems like Livingstone was that they could increase science return during routine operations by automatically returning spacecraft to operations. The summary of MER anomalies suggests the rovers have lost less than 3% of operating time to anomalies, and for many of those (*e.g.*, stuck in the sand) it's not clear a system like Livingstone would help. Thus it seems fair to question the cost and risk of adding "fail operational" capabilities and continuing to execute after an anomaly. A second hope was that during critical periods such as orbital insertion where one has to continue operating in the face of anomalies, the ability of a model-based diagnosis system to generate novel diagnostic combinations could save a mission. For periods where a misstep could result in mission loss, one has to weigh the potential benefit of generating novel but untested responses to less likely failures against a set of carefully engineered and validated responses to a smaller set of more likely failure scenarios. A broader discussion of this issue is found in [23].

5 A Preliminary Discussion on Innovation Infusion at ESA

To complement the previous analysis we discuss here some points which we consider as relevant for a more effective infusion within ESA programs of the more mature of the AI innovative technologies. This analysis focuses on the subareas identified in this paper and include observations that span over the last decade[1]. Such observations are either subjective and preliminary but are inserted here to complement the information survey and give a complete picture of the effort which is underway to identify improvements of innovation infusion within the Agency.

Technological innovation spreads within the Agency in extremely slow manner. On one hand this is due to the natural and understandable conservativeness of the environment given the criticality of the space missions. On the other hand this is due to the limited role that research groups have within the agency. The technologies we have targeted in this paper can be considered as "enabler" for injecting flexibility in future missions. Given the current ESA organization, such technological innovation is mostly delegated to industrial partners that directly manage all the technological aspects of Agency space programs under the restriction given by specific contract. In general, apart the role of little specific offices no official role is recognized to technological research. All the innovation effort, the "real research" of the agency, is dedicated to science part of each mission, while enabling technologies that can orthogonally innovate the way all the missions are run do not have an explicit

[1]One of the co-authors have been observing ESA efforts in the area since May 1999 when the iSAIRAS Symposium was hosted at ESA-ESTEC.

role and receive a spotted attention within single program with no or little exchange of information not only with the external world but also internally.

We can use planning and scheduling as an example. We have had spotted examples in the 80s, during the "expert systems era", then almost no activity during the 90s, then an increasing interest from 1999 after observing the NASA DS-1 autonomy experiment. Unfortunately, such interest has not been supported by any organic investment with respect to the European research community in P&S that in the meantime (from 90s to 00s) has achieved levels of absolute excellence and international competitiveness. Nevertheless, few interesting success stories have happened, e.g., the MEXAR2 [5] and RAXEM [6] within the Mars Express mission planning are two of them. Such cases were motivated by "local chemistry" between a single research group and a group of mission specialists that met at the right time and with the right problems at hand. They have represented brilliant examples of good practice that were not generated by any systematic Agency investment in innovative ideas from research labs. Some other more recent efforts, like the 2006 ITT called "Advanced Planning and Scheduling Initiative (APSI)" and other efforts which are active at ESA-ESTEC, e.g., the collaboration for robotic planning described in [38], are example of developing research ideas strongly motivated by space problems. Result of these efforts has been an increasing awareness within the agency that the P&S technology is mature enough to be used in real mission. Nevertheless the related Agency investment in attention, focus, and money remains limited in temporal extent and extremely localized with respect to involvement of larger groups of people.

Even if it is difficult to offer clear recipes for improving the current status of affair, we end this section pointing our few aspects that need some attention to start improvements of future scenarios. A major problem is due to the agency exclusive contractualization. Even in case of limited investments the contract with a research group are considered as industrial one. Hence the outcome of the contract become "exclusive property" of the agency. A relaxation of this aspect, for example adopting a 75%-25% co-sharing of the investments in the project would allow the research groups to maintain the property of their know how. It is worth saying that usually such know-how generated from years of experience and seldomly generated on-purpose for a spot grant from the Agency. Another problem is represented by the "single winner" modality for obtaining any ESA contract. While we acknowledge this as a very natural regulation for industrial contracts, when applied to research it prevents the creation of a social network of researcher laboratories around the mission problems of the agency. In this respect a multiple stage procedure with increasingly restrictive selections at each stage, like is done for example in various of the DARPA programs, would allow to obtain more innovative ideas in the early stages and the subsequent decision to select one for the current program would allow the groups which are not selected to gain experience that can return useful to the agency in future programs. A third problem is connected to the difficulty for the external research community to identify the clear long term strategies of the Agency with respect to research directions. In order for a research line to be productive an investment is needed on a medium term (5-10 years). While the Agency program for innovation in science initiatives is quite clear, the policy for innovation in technological innovation is not. In order to have a fruitful interaction with research labs would be beneficial to create roadmaps that clarify the open points that need innovation, to make available test cases of open problems, even make available software simulators for certain scenario (e.g., for the ESA ExoMar rover) and then identify clearly which are the unsolved problems and their connections with the current missions.

6 Conclusions

This paper has identified some aspect where autonomy can be applied and discussed the level of autonomy that is useful for missions to achieve. It has presented an overview of some of AI techniques that can contribute to inject Autonomy in space missions. In particular we have considered AI Planning and Scheduling, Intelligent Execution and Model-Based Diagnosis. Finally, the authors envisioned what they thought possible trends in each area would be and what future ESA missions will require.

Acknowledgments. We would like to acknowledge the support of European Space Agency under project AO-1-5415/07/F/VS (AMOCT), Junta de Castilla-La Mancha under project PAI07-0054-4397 and NASA.

References

[1] Aschwanden, P., Baskaran, V., Bernardini, S., Fry, C., R-Moreno, M., Muscettola, N., Plaunt, C., Rijsman, D., and Tompkins., P. Model-Unified Planning and Execution for Distributed Autonomous System Control. In *AAAI 2006 Fall Symposia. Washington DC (USA), October* (2006).

[2] Blum, A., and Furst, M. Fast Planning Through Planning Graph Analysis. *Artificial Intelligence 90* (1997), 281–300.

[3] Blythe, J. Decision Theoretic Planning. *AI Magazine 20*, 2 (1999), 37–54.

[4] Bonet, B., and Geffner, H. Planning as Heuristic Search: New results. In *Procs. of the ECP-99. Springer* (1999).

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

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

[7] Cesta, A., Cortellessa, G., Fratini, S., and Oddi, A. MrSPOCK: Generating a planning system through a timeline representation framework. In *ASTRA-08. Proceedings of the 10^{th} Workshop on Advanced Space Technologies for Robotics and Automation* (2008), ESA-ESTEC.

[8] Cesta, A., Oddi, A., and Smith, S. A Constraint-Based Method for Project Scheduling with Time Windows. *Journal of Heuristics 8*, 1 (2002), 109–136.

[9] Cesta, A., Policella, N., and Rasconi, R. Coping with Change in Scheduling: Toward Proactive and Reactive Integration. *Intelligenza Artificiale* (2006).

[10] de Kleer, J., and Williams, B. C. Diagnosis with behavioral modes. In *Proceedings of IJCAI-89* (1989), pp. 1324–1330. Reprinted in [20].

[11] Erol, K. *HTN Planning: Formalisation, Analysis and Implementation.* PhD thesis, Computer Science Dept., University of Maryland, USA, 1995.

[12] Fikes, R., and Nilsson, N. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence 2*, 3/4 (1971).

[13] Foss, J., Onder, N., and Smith, D. Preventing Unrecoverable Failures through Precautionary Planning. In *Procs of the ICAPS'07 Workshop in Moving Planning and Scheduling Systems into the Real World* (2007).

[14] Frank, J., and Jonsson, A. Constraint Based Attribute and Interval Planning. *Journal of Constraints. Special Issue on Planning. 8*, 4 (2003), 339–364.

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

[16] Gerevini, A., and Long, D. Plan Constraints and Preferences in PDDL3. The Language of the Fifth International Planning Competition. Tech. Rep. Technical Report, Department of Electronics for Automation, University of Brescia, Italy, 2005.

[17] Ghallab, M., and Laruelle, H. Representation and Control in IxTeT, a Temporal Planner. In *Procs. of the Second International Conference on AI Planning Systems (AIPS-94)* (1994).

[18] Godard, D., Laborie, P., and Nuitjen, W. Randomized Large Neighborhood Search for Cumulative Scheduling. In *ICAPS-05. Proceedings of the 15^{th} International Conference on Automated Planning & Scheduling* (2005), pp. 81–89.

[19] Grasso, C. The fully programmable spacecraft: Procedural sequencing for jpl deep space missions using vml (virtual machine language). In *IEEE Aerospace Applications Conference* (2002).

[20] Hamscher, W., Console, L., and de Kleer, J. *Readings in Model-Based Diagnosis.* Morgan Kaufmann, San Mateo, CA, 1992.

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

[22] Kurien, J., and Nayak, P. P. Back to the future with consistency based trajectory tracking. In *Proceedings of AAAI-00* (2000).

[23] Kurien, J. A., and R-Moreno, M. D. Costs and benefits of model-based diagnosis. In *IEEE Aerospace Conference* (2008), IEEE.

[24] Levinson, R. Unified Planning and Execution for Autonomous Software Repair. In *Procs. of the ICAPS05 Workshop on Plan Execution: A Reality Check. Monterey, CA, USA* (2005).

[25] Liberatore, P. On the Complexity of Choosing the Branching Literal in DPLL. *Artificial Intelligence 116*, 1–2 (2000), 315–326.

[26] McAllester, D. Truth Maintenance. *Procs. of the 8th Nat. Conf. AI* (1990), 1109–1116.

[27] Muscettola, N., Nayak, P. P., Pell, B., and Williams, B. C. Remote agent: To boldly go where no ai system has gone before. *Artificial Intelligence 103*, 1-2 (1998), 5–47.

[28] Muscettola, N., Smith, S., Cesta, A., and D'Aloisi, D. Coordinating space telescope operations in an integrated planning and scheduling architecture. *IEEE Control Systems 12*, 1 (1992), 28–37.

[29] Neilson, T. C. MER on-board surface fault protection. In *Proceedings of IEEE SMC 2005* (October 2005), IEEE.

[30] Pell, B., Gamble, E. B., Gat, E., Keesing, R., Kurien, J., Millar, B., Nayak, P. P., Plaunt, C., and Williams, B. C. A hybrid procedural/deductive executive for autonomous spacecraft. In *Second International Conference on Autonomous Agents* (1998).

[31] Policella, N., and Cesta, A. Uncertainty: an open issue in project scheduling. Tech. rep., ISTC-CNR, Planning and Scheduling Team, November 2006.

[32] Puterman, M. *Markov Decision Process: Discrete Stochastic Dynamic Programming.* John Wiley and Sons., 1994.

[33] Selman, B., Levesque, H., and Mitchell, D. A New Method for Solving Hard Satisfiability Problems. *Procs. of the 10th Nat. Conf. AI* (1996), 440–446.

[34] standard, E. S. S. O. Space Engineering Space Segment Operability. Tech. Rep. ECSS-E-70-411A, 2005.

[35] Verma, V., Estlin, T., Jonsson, A., Pasareanu, C., Simmons, R., and Tso, K. Plan Execution Interchange Language(PLEXIL) for Executable Plans and Command Sequences. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS), Munich, Germany, 5-8 September* (2005).

[36] Verma, V., Jonsson, A., Simmons, R., Estlin, T., and Levinson, R. Survey of command execution systems for nasa robots and spacecraft. In *Workshop at The International Conference on Automated Planning and Scheduling (ICAPS05)* (Montery, CA, USA, 2005).

[37] Weld, D. An introduction to least commitment planning. *AI Magazine* (1994).

[38] Woods, M., Baldwin, L., , Wilson, G., Hall, S., A., P., Long, D., , Fox, M., Aylett, R., and Vituli, R. MMOPS: Assessing the Impact of On-Board Autonomy for Planetary Exploration Missions. In *SpaceOps-06. Proceedings of the 9th International Conference on Space Operations* (2006).