# ERGO: A FRAMEWORK FOR THE DEVELOPMENT OF AUTONOMOUS ROBOTS

Jorge Ocón[(1)], Juan Manuel Delfa[(1)] , Alberto Medina[(1)], Daisy Lachat [(2)], Robert Marc [(2)], Mark Woods [(3)], Iain Wallace [(3)], Andrew Coles [(4)], Amanda Coles [(4)], Derek Long [(4)], Thomas Keller [(5)], Malte Helmert [(5)], Saddek Bensalem [(6)]

[(1)] *GMV Aerospace and Defense, Isaac Newton, 11, PTM Tres Cantos, 20760, Spain, Email: jocon@gmv.com*
[(2)] *Airbus Defence and Space Ltd., Gunnels Wood Road, Stevenage, SG1 2AS, UK, Email: gnc.uk@astrium.eads.net*
[(3)] *Scisys UK Ltd, Methuen Park, Chippenham, SN14 0GB, UK, Email: mark.woods@scisys.co.uk*
[(4)] *King's College London, Strand, London WC2R 2LS, UK, Email: andrew.coles@kcl.ac.uk*
[(5)] *University of Basel, Spiegelgasse 1, Basel 4051, Switzerland, Email: {tho.keller,malte.helmert}@unibas.ch*
[(6)] *Universite Grenoble Alpes, 700 Avenue Centrale, 38400 St Martin D'Heres, France, Email: saddek.bensalem@univ-grenoble-alpes.fr*

ABSTRACT

The European Robotic Goal-Oriented Autonomous Controller ERGO (http://www.h2020-ergo.eu/) is one of the six space robotic projects in the frame of the PERASPERA SRC (http://www.h2020-peraspera.eu/). Its goal is to provide an Autonomy Framework capable of operating at different levels of autonomy, from tele-operations to full on-board autonomy. Even though it has been originally conceived for space robotics, its domain independent design facilitates its application to any terrestrial robotic system. This paper presents the approach followed, current status and future steps.

## 1. INTRODUCTION

For many years, space agencies have pursued the development of autonomous systems. While this technology can be applied to virtually any mission, deep space and more specifically planetary exploration have benefited the most. This is for many different reasons, mainly the combination of a harsh environment, and limited communications due to the long distances and communication windows, that characterize deep space missions and planetary exploration.

NASA's approach to autonomy differs between satellites and rovers. The former have shown high levels of autonomy at mission level, with notorious examples such as EO-1 [1] and DS-1 [2]. On the other hand, autonomy in rover missions has been applied to resolve specific problems, from the autonomous navigation system AUTONAV [3] to the opportunistic science agent AEGIS [4]. European approach has proceeded in other directions. While some missions such as Rosetta [5] have displayed specific solutions, ESA and the European Commission are pursuing the development of a generic autonomous controller to be applied in any type of mission requiring autonomy.

The European Research Agency is leading a Strategic Research Cluster (SRC) in Space Robotics Technologies that is the most recent, most ambitious European programme ever, aimed at developing key robotic technologies organized in six areas. This paper presents ERGO, the SRC project oriented to develop a highly autonomous, mission-independent controller based on the lessons learnt from previous efforts, especially from GOAC [6], ExoMars Rover GNC [7] and MASTER.

## 2. OBJECTIVES

The ERGO System aims to achieve the following objectives:

- Mission-independent: ERGO shall be suitable for different kind of individual/collaborative robotic systems, from space (e.g. satellites and rovers) to terrestrial (e.g. mobile platforms).
- Goal-based: ERGO shall be able to be commanded via high-level goals, that is, it is capable of reaching the E4 level of autonomy defined in the ECSS Standards [8]
- Multiple levels of autonomy: ERGO shall be able to handle the four autonomy levels defined in the ECSS standards, that is, shall allow teleoperation in real-time (E1), commanding via time-tags (E2), to be event-driven (E3) as well as supporting goal commanding (E4). Moreover, it shall be able to dynamically reduce its level of autonomy in case the conditions for a high level of autonomy are not met.
- (Re)Planning capabilities: ERGO shall be able to generate plans on-board based on a list of high-level goals, and to autonomously fix the plan on-board, adapting the plan to the exogenous events that occur during its execution.
- Suitable for flight: ERGO shall be designed bearing in mind future requirements to deploy it on-board spacecraft.

Focusing on the domain, space applications pose a number of constraints with a big impact on the design of ERGO. The environment might present high levels of uncertainty (partial observability, non-determinism and dynamism); spacecraft on-board resources such as CPU power and memory are scarce; communication links with ground can be interrupted during long periods, or have low bandwidth and high latency. In addition, operations can be highly complex due to the inherent sophistication

of the mission/payload and the constraints associated to their operations; spacecraft represent critical systems for which high safety standards must be enforced.

## 3. THE ERGO AGENT

ERGO is an architecture (Figure 1) that inherits its basic principles from the architecture developed in the GOAC project. GOAC, in turn, was heavily based on T-REX [9], an architecture developed and tested in autonomous underwater robots, at the Monterey Bay Aquarium Research Institute. The ERGO architecture, inherited from GOAC and T-REX, conceives the system as a set of different control loops managed by a single agent (the controller). Each of these control loops is encapsulated into a so-called *reactor*. Reactors can be deliberative or reactive, and they share a common interface with the controller. Working at the highest level of autonomy (E4), a deliberative reactor, the mission planner, performs the high-level goal decomposition; meanwhile, purely reactive reactors are in charge of handling lower levels of autonomy.

The controller maintains a level of autonomy as a parameter of the system that determines the level of commanding that can be issued. It can be set from ground, or modified internally whenever the conditions for a high level of autonomy are not met.

The level of autonomy is managed by a single reactor: the ground controller interface reactor (GCI), that processes telecommands received from ground. Commanding at the lower levels of autonomy (E1, E2 & E3) is performed by purely reactive reactors using a combination of the traditional PUS Services. Commanding at the highest level of autonomy (E4) requires the mission planner to perform high-level goal decomposition into lower-level goals. An additional reactor, the so-called command dispatcher is in charge of interfacing with the functional layer that provides abstraction w.r.t. the underlying hardware.

While these three mentioned reactors (Mission planner, Ground Control Interface and Command Dispatcher) are common to any space mission, other reactors can be added to the agent for specific purposes.

In particular ERGO is aimed to tackle two different scenarios: an orbital scenario (specifically for a robotic arm), and a planetary exploration rover scenario. The number of reactors for each use case will be different and tailored to suit their needs. For this purpose, a set of additional, deliberative reactors complement the ERGO architecture:

- A Rover Guidance reactor is in charge of performing the navigation of a planetary exploration rover.
- An opportunistic goal detector is in charge of detecting serendipitous events, autonomously posting new goals to the planner whenever an interesting event is raised. This reactor will take part of the configuration for the planetary exploration rover as well.
- An arm motion planner is in charge of planning the movements of the robotic arm.

As in GOAC and T-REX, the agent controller is responsible of the correct flow of information between the different reactors. All reactors share a common interface that is used to exchange facts (observations) and commands (goals) to/from other reactors. This communication is based on state variables (i.e. timelines). Time is discretized, and the agent controller follows an algorithm to periodically synchronize the status of each reactor; to forward the goals to those reactors that are in charge of performing them; and to inform the reactors interested in given timelines of those observations that affect them. All of this process is performed using the reactors' common interfaces.

By doing so, the architecture has a series of advantages against other traditional architectures, namely:

- Scalability: the architecture can be extended very easily by adding new reactors, without the need to redefine the interfaces between them.
- Consistency: the algorithm being used (detailed in [9]) guarantees that all reactors share the same status of the system.
- Portability: the system does not depend on a particular scheduling policy. All reactors are controller by the agent.
- Easy integration of new deliberative reactors.

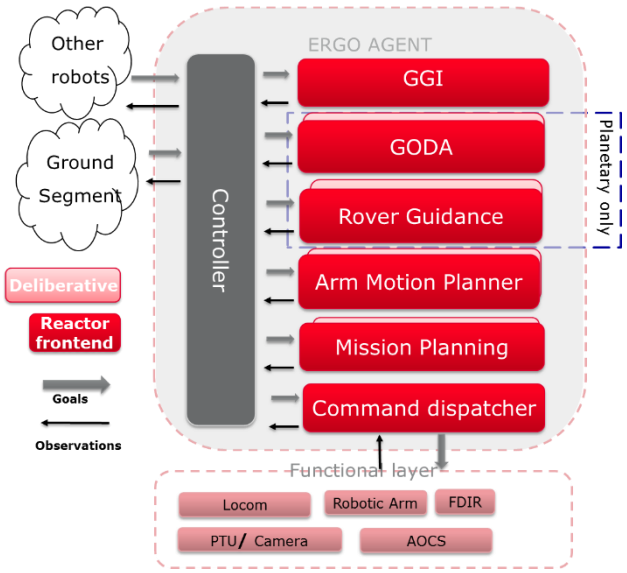


**Figure 1 - ERGO Architecture**

In addition, the agent can be extended to communicate with other agents, sending and receiving goals and facts to/from other agents running in other robots, using the same interfaces for both remote and local reactors. Finally, the ERGO agent is conceived as a TASTE [10] component. TASTE [11] is a software framework that allows the development of critical embedded, real-time systems, but it can be used also for terrestrial domains. It relies on key technologies such as standardized modelling languages (e.g., ASN.1 [12] and AADL [13]),

code generators and real-time systems. This allows the generation of suitable skeletons, and glue code that can be combined with code developed by the user into an application's executable. TASTE currently supports the generation of code for Linux and SPARC/RTEMS platforms. For a smooth integration of the agent into TASTE, we count on the know-how and experience of Ellidiss, one of the main partners involved in the development of TASTE.

## 4. DECISION LAYER – AN INTEGRATED APPROACH TOWARDS DELIBERATION

A big part of ERGO's effort has been focused on the decision layer, providing a number of novelties.

From an architectural point of view, two principles are considered fundamental: 1) All deliberative subsystems exchange the same type of information, that is, they send goals and they receive observations related to these goals. This aspect facilitates the adaptation of ERGO to the specific requirements of future missions; 2) In order to create more accurate plans, the mission planner can query dedicated systems, specific for each mission, for detailed information about certain aspects of the plan. This approach, derived from QuijoteExpress [14], proved to be extremely helpful in order to generate more accurate plans in the frame of the FASTER project [15].

With respect to specific deliberative subsystems, there are three technical aspects to remark: 1) ERGO represents the first effort to develop a PDDL planner for space applications, which is an incredible challenge taking into consideration the complexity of temporal planning and the scarce computational resources available in a spacecraft; 2) A global approach to rover guidance for long traverses beyond the field of view, capability that is crucial in future missions such as MSR. It exploits satellite imagery and on-board LIDAR scan to enable fast and robust traverse;3) Finally, ERGO will provide automatic detection, not only of scientific targets as demonstrated with other systems such as Aegis, but also of other space assets, such as the canister containing samples in the MSR mission. These three capabilities, crucial for future surface exploration, will be further detailed in the following subsections.

## 5. STELLAR - A NOVEL PDDL PLANNER FOR SPACE APPLICATIONS

One of the most important novelties proposed by ERGO is the development of the first-ever PDDL-based planner for space applications, named STELLAR, inspired by three established planners: Optic [16], Fast Downward [17] and QuijoteExpress.

There have been two traditionally antagonist approaches to (temporal) planning: state-based and action-based. There are several similarities between them: both receive as inputs a domain (formally defining the system which activities need to be planned) and problem (containing the current status of the system and goals); both divide the search for a plan in planning and scheduling phases; both use variants of the same algorithms: heuristic search (e.g. A*) for planning and All-pairs shortest paths (e.g. Floyd Warshall) for scheduling.

But they present as well important differences: timeline planners (a variant of state-based), are typically used in space while PDDL (a variant of action-based) are widely used in academia; the former have proprietary languages (NDDL, DDL, …) while the latter are based on the international de-facto standard language PDDL [18]; the outputs are also different, in the former being represented as (flexible) timelines (one for each subsystem of the functional layer) while the latter contains a set of possibly concurrent actions with rigid times.

At first glance the move to PDDL for space applications does not seem obvious, apart from the benefit of using a standard. However, PDDL-based planners often outperform their timeline-based counterparts, which is a major advantage especially due to the limited on-board resources. The following paragraphs present the on-going work to extend PDDL to cover those areas where it falls short and the design of a new planner based on this extension.

The on-board resource constraints set a tight envelope on the way in which on-board deliberation can be performed. Traditional search-based planning approaches typically either combine cheap heuristics (that offer only little guidance) with a search procedure that requires gigabytes of local memory to store large parts of the state space of the planning task; or, they compute informative, state-of-the-art heuristics that offer excellent search guidance (but are expensive to compute), which allows to consider only small parts of the search space. Unfortunately, both approaches are infeasible on-board, as memory, time, and energy constraints tightly constrain how the planner may operate. For these reasons, it is essential to minimize the search effort on-board, e.g., by finding ways that allow us to precompute resource-intensive subtasks of the planning process on-ground, which can then be used to lower resource consumption of the planning process that takes place on-board.

STELLAR aims to plan from scratch only when necessary. If a plan exists that, for some reason, is no longer valid, it uses re-planning techniques that build on the existing plan. For instance, re-planning is triggered as a consequence of the mismatch between observed and expected facts during plan execution or because new goals have been received. To re-plan, we "start" the search from all states that are reachable under the current plan, which reuses some of the computational effort gone into finding the existing plan; and we aim to reach some point along what was intended to be the remainder of the plan, which "patches" the existing solution by finding actions that reach a state that allows some of the rest of the existing plan to succeed.

Another technique that allows on-board planning under scarce resources is plan-refinement: the planning task is modelled as a set of abstract high-level tasks, each achieved by a number of low-level tasks. This approach has demonstrated potential benefits in terms of performance. Moreover, it does not require any additional development at software level, as these operations can be performed by the same mission planner just by providing it with the appropriate inputs.

The inputs of the planner (domain and problem) are modelled with PDDL 3.0, the standard language of the planning and scheduling community. The planning domain is static over the course of a mission. Only the planning problem is dynamically created by combining information from all sub-systems. The result of the ERGO mission planner is a plan based on flexible timelines, which guarantees its integration into the planning reactor of the ERGO agent, an output that is compatible with timeline-based planners, such as APSI [19].

Planning in ERGO is based on heuristic search in a search space of states that are connected via action application. Each state consists of a propositional and a numeric part, which we either store explicitly for fast access in a 'bit packed' representation (if memory allows) or construct implicitly on demand by application of the sequence of actions that leads to the state. Internally, actions may correspond to starting a new action, or ending one that is currently executed. To start a new action, it must be applicable, i.e., its preconditions must have been met, and as long as an action is executing, its invariants must be respected. Invariants of running actions are encoded in linear temporal logic, and all invariants are combined to a simple temporal network (STN) that is associated with each state.

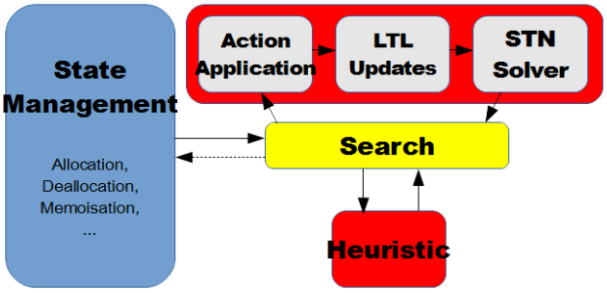


**Figure 2 - STELLAR planner components**

The requirement to find a plan that respects temporal constraints means we cannot assume that any sequential plan containing successive logically applicable actions is reasonable. In temporally lifted progression planning, applying an action orders it after a subset of the previous actions in the plan. Additionally, ending an action adds a duration constraint placing the end of the action after its start. The advantage of such an approach is that it can support problems with *required concurrency*: the timestamps of actions are determined by applying a suitable shortest-path algorithm to the STN, which will assign feasible timestamps to actions (or show that no satisfying set of timestamps exists). The disadvantage is it carries a time overhead (STN solving) and in principle a space overhead (storing additional temporal information in each state).

To minimize time overheads, we only need to solve the STN when necessary – i.e. when it might be inconsistent. Inconsistency in the STN arises due to negative-length cycles, and there are two useful cases where consistency is guaranteed. First, if we start an action, and it must come only after previous actions by an unbounded amount – with no other relevant temporal constraints – this cannot make the STN invalid, as no 'maximum duration' edges (with finite positive weights) have been added between the existing actions, and the new one. Second, if we end an action and the only constraint is it must follow its start, again this cannot make the STN invalid, as the only cycle introduced is between the start and end of the action; and we can assume actions' minimum durations do not exceed their maximum.

In other cases, it is necessary to run a consistency check on the STN. To militate against costs here, incremental algorithms can be used. When expanding a state S during search, we know its STN is consistent – or we would not consider it for expansion. Thus, consistency checking in a state S' reached by extending the STN from S (to contain an additional node, with the relevant temporal constraints) is an incremental update.

With regards to memory overheads, when using an STN approach, the question is what temporal information actually needs to be stored in the state? In principle, all information can be derived from a (partial) plan on demand. However, to minimize computation time, we adopt the concept of OPTIC and keep an explicit record of: 1) the current STN; 2) each variable that was changed to enable the application of an action; 3) the minimum timestamp that could be given to each action.

Storing the first of these has the advantage that the STN does not have to be reconstructed by stepping through all the actions that lead to a state. The second reduces the time taken to identify which previous actions each new action needs to be ordered after. The third would be the most expensive to derive, needing to both reconstruct and solve the STN; and with reference to incremental STN solving, the minimum timestamps of existing actions are required as input, so removing this information from the state would prevent us from reducing time overheads using incremental techniques.

To find a plan in the search space, we apply heuristic search. Given the limited resources, it is unlikely that plans that are guaranteed to be optimal can be computed. We therefore do not use A* search in combination with an admissible heuristic, but use Weighted A* instead. This has the advantage that we can still provide a bound

on the quality of the plan, but we are able to react on long planning times by allowing plans of lower quality. Moreover, it also allows us to enhance search with a simple anytime component, where we continue search with decreasing weights after an initial solution has been found, which leads to a sequence of plans of increasing quality as long as time and resource bounds allow the planner to continue search, while an initial plan is found as quickly as possible.

The decision which heuristic to use is a challenging one. Apart from the limited resources, this is also because most (if not all) well-known heuristics have been developed in the context of classical planning, i.e., for environments without temporal aspects, resources, external functions, intermediate goals, complex metrics or uncertain effects. However, due to dependencies between the heuristic and search, it is not the case that a heuristic function that uses less processing power and memory is always preferable: a heuristic that uses more resources is often better informed, which leads to significantly less search effort and hence to overall fewer consumed system resources. We therefore aim to use a heuristic with a good ratio between accuracy and resource consumption.

*Delete relaxation heuristics* [20] estimate the cost of reaching a goal state by considering a *relaxed task* derived from the actual planning task where all delete effects of operators are ignored. The most promising candidate is the FF heuristic [21], which approximates the optimal (NP-hard) delete relaxation heuristic $h^+$ in polynomial time and space in an inadmissible but often fairly accurate way.

Most *abstraction heuristics* have in common that they are cheap to evaluate in a state, but require a rather expensive preprocessing step. We plan to perform the preprocessing step on-ground (on high-performance computers), while only the (compact) result of the preprocessing is made available on-board to guide search. If this is successful, informative abstraction heuristics like the merge-and-shrink heuristic [2221] or a cost-partitioned [23], [24] set of abstraction heuristics are promising candidates.

*Potential heuristics* [25], which encode a linear function over a weighted set of features, can also optimized on-ground in an elaborate preprocessing step and are very cheap to evaluate. As we aim to *learn* the potentials, they have the advantage that they can incorporate arbitrary complex environment features as long as they are reflected in the training set. Furthermore, they are suitable for a "human-in-the-loop" approach where experts on ground provide training examples which are used to improve the resulting potential heuristic

## 6. ROVER GUIDANCE – AN EFFICIENT NOVEL APPROACH FOR LONG-RANGE NAVIGATION

The ERGO Rover Guidance is the responsibility of Airbus Defence and Space Ltd. It builds upon expertise and know-how from the ExoMars Rover Guidance, Navigation and Control (GNC) in order to enable an extremely long distance travelled of the order of 1km per day, while keeping the rover safe at all times. The Rover Guidance (RG) includes five building blocks (Figure 3).

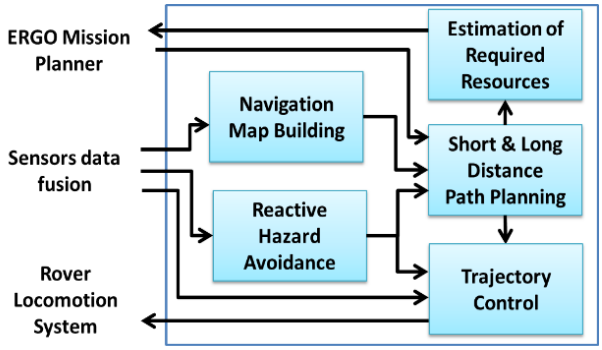


**Figure 3 - Rover Guidance building blocks**

The Rover Guidance receives inputs from:

- The ERGO mission planner which provides the long distance goals to travel towards.
- A sensor data fusion system, providing the estimated rover position & attitude and a digital elevation map of the area to be explored. This would be performed by fusing data from LIDAR, stereo-camera, IMU and HiRISE orbital maps.

For the ERGO demonstration, the sensor data fusion inputs will be provided by the test platform which is implemented by FACILITATORS, another PERASPERA Operational Grant (OG6) running in parallel. In the future, the Rover Guidance system will use for these inputs the data provided by InFuse, the data fusion system being developed in another PERASPERA Operational Grant (OG3). The RG outputs a long-term path with required resources estimation to be used by the mission planner, and also outputs rover commands in order to follow the short-term path.

The *Navigation Map* summarizes the results of terrain modelling and terrain analysis. Essentially it describes the rover's understanding of its neighbourhood with respect to hazards, objects, terrain and sites of interest. In order to create this map, the Rover Guidance uses inputs from the fused sensory data: including terrain representation (e.g. digital elevation map – DEM) and orbital information. This information is analysed to select the safe areas to travel through and also to estimate the terrain difficulty for these areas.

The *Short-Term Path* is planned on the navigation map and is dynamically compatible with rover driving capabilities. The *Long-Term Path* planner computes the

rover path in the long distance taking into account identified obstacles and likelihood of the challenging terrain areas from HiRISE orbital data.

*Resources* are estimated solely for the mission planner, using the long-term path and orbital navigation map.

The *Reactive Hazard Avoidance* detects obstacles which have not been identified at the time the path was planned as they will become apparent in the near vicinity of the rover. In addition, closed-loop trajectory control will enable the rover to maintain its path along the planned trajectory while encountering environmental disturbances.

The requirements of the Rover Guidance system can be summarized as follows:

- Rover Safety: ensure the rover to be commanded to drive solely in safe areas of the terrain. A safe area is defined as geographical zone where the rover hardware cannot be damaged by the static environmental conditions. Moreover the rover will not get physically trapped in these safe areas.
- Navigation map & cost: create a navigation map which defines the areas of the terrain that are traversable by the rover. Including a cost will help the path planner algorithm to decide on the most desirable area to be traversed.
- Long-distance path planning: plan a long-distance path from the current rover location to the long distance target location.
- Short-distance path planning: plan a short distance path with associated desired manoeuvres (trajectories) that is both safe and drivable by the locomotion system.
- Rover commanding: control the rover by issuing manoeuvre commands towards the locomotion system, adapt these as needed to stick to the desired path, and check if the rover stays within a safe corridor limit.
- Hazard avoidance: Identify and react to new hazards appearing across the planned short-distance path. This approach will enable the usage of a non-comprehensive navigation map and will provide flexibility in the design.
- Resources Estimation: estimate the required resources to execute the path that the mission planner requires for decision making.
- Rover type and mechanical configuration: to be designed as agnostic as possible with respect to the rover type and mechanical configuration. For highly dependent functions, the rover type and its mechanical configuration assumed for the RG design is the one considered most likely to be used for a future Mars Sample Return mission.

## 7. OPPORTUNISTIC SCIENCE AND PLANETARY ASSET DETECTOR FOR FLEXIBLE OPERATIONS

Bandwidth or communication limitations may make real-time control of instruments for scientific discovery difficult or impossible. For planetary rovers there is a trade-off between detailed observation to ensure important targets are not missed, which requires slow traverses to downlink all the data, and maintaining sufficient progress to visit many science targets. For the orbital case, similar technology can aid in the detection of targets of interest and misplaced or unexpected objects that may lead to dangerous situations.

The GODA component of ERGO builds on work in several previous ESA studies. These include planetary aerobots [26] which ranked images for downlink based on geological science content and the CREST [27] opportunistic science system that analysed images for targets of scientific interest. The PRoViScout [28] project developed a science assessment and response agent (SARA) to identify science targets and allow different reactions dependent on power and time constraints. Wider afield NASA JPL are actively investigating this topic and have deployed a basic form of autonomous science detection known as AEGIS on the MER Opportunity Rover [4]

More recently, SCISYS's work in the MASTER [29] study represents the cutting edge of autonomous space system development, and relates to several topical developments in the wider robotics and vision literature. Whilst overall the work was firmly aimed at closing the action-perception loop, MASTER only considers the initial attention-acquisition event.

The MASTER project showed a promising future for computer vision and machine learning approaches in the space domain. The system prototyped could be developed into a broad range of potential applications ranging from flight systems on planetary rovers to labelling and annotation tools to support terrestrial scientists. The work on ERGO will represent one such further development of the system and the chance to advance the state of the art in autonomous scientific agents.

The GODA component design, shown in Figure 4, pairs a MASTER-like detector with a goal generation component. The goal generation component maps from detections of phenomena of interest onto concrete goals for the planner to achieve. For example, detection of novelty could trigger a goal to acquire high resolution imagery or detecting a broken manipulator might trigger a goal to put the spacecraft in a safe state. As well as the work of MASTER, the detector component will also benefit from ongoing research into science autonomy that is being pursued in parallel, the ESA NOAH [30] project looks to extend the capability and raise the TRL of work started in MASTER. Recent advances in deep

convolutional neural networks for image processing hold great promise for space applications. Whilst training them requires great compute, inference at detection time is a fixed known cost bounded by model complexity. The outputs from NOAH will allow us to take advantage of such improvements.

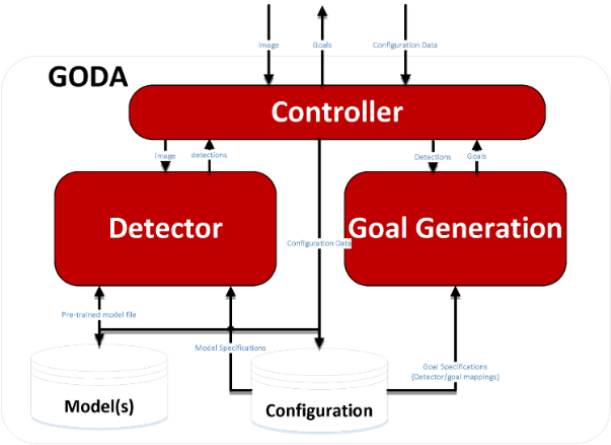


**Figure 4 - GODA Component Design Overview**

## 8. OFFLINE AND ONLINE VERIFICATION AND VALIDATION

The techniques investigated in the project will be incorporated into a modelling and verification framework, called BIP Framework [31] which provides a facility for model extension, that is, integration of nominal and error models, and simulation of the behaviour of the system in presence of faults once one or more fault injections are defined.

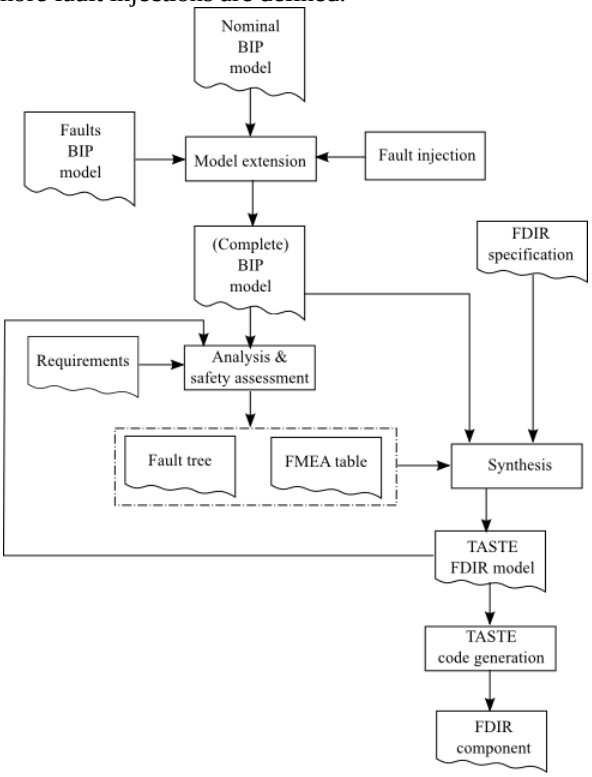


**Figure 5 - Planned workflow for the FDIR component generation.**

The BIP framework provides several capabilities by using a single model description, including requirements validation, functional verification [32] and performance analysis [33].

The results of this project will provide a good basis for dependability and safety assessment, fault tolerance evaluation, and FDIR development. We will obtain the necessary elements for the FDIR design and modelling together with the facilities for diagnosability and FDIR analysis. The workflow of this process is illustrated in Figure 5. The FDIR development approach will be supported by rigorous formal methods, providing the possibility of application in the early development stages allowing for effective use of the available software and system designs and corresponding RAMS analysis data. Furthermore, FDIR design will be implemented in accordance with the FDIR requirements, software and system architectural design, and system-level dependability requirements.

## 9. CONCLUSIONS

The ERGO architecture is a novel architecture that benefits from the experience and know-how of many companies across Europe. The requirements for ERGO have been defined and approved, and the project is currently in the design phase.

The ERGO agent controller represents a new architecture that provides the four levels of autonomy, and guarantees modularity, cohesion and scalability.

STELLAR is the first PDDL planner specifically conceived for space applications. Its design inherits from the combined lessons learnt from classical (temporal) PDDL planners such as Optic and Fast Downward, and timeline-based planners such as QuijoteExpress. The objective is to achieve the high performance characteristic of the former while preserving high level of expressiveness (especially temporal) typical from the second. In ERGO these restrictions do not apply, since the mission planner uses flexible time boundaries for the execution, and it has an interface based in timelines with the executive.

The system is conceived to be easily instantiated and tailored to suit the needs of any particular robotic platform by 1) adapting the planning domain, 2) adapting the interface with the functional layer, and 3) defining the commands and the telemetry to be sent and received from/to ground.

One of the strengths of the project is that it is built based on the experiences on previous projects for space robotics, for instance GMV's experience on autonomous controllers from GOAC, the rover guidance design builds on the expertise of Airbus Defence and Space Ltd. from the ExoMars Rover GNC; meanwhile GODA inherits a long experience from Scisys in previous ESA projects.

ERGO conceives the use of rigorous formal methods for the FDIR design and modelling, and model-driven techniques based on the TASTE framework and BIP.

Moreover, ERGO is in line with the other PERASPERA building blocks, as it is designed to be easily integrated with the robotic operating system being developed in ESROCOS (OG1) or data fusion components developed in InFuse (OG3).

In the near future, the team will start to develop the different ERGO components. A number of field tests have been defined, based on the Mars Sample Return mission. The feedback obtained will be key to demonstrate the concept and to improve the robustness of the system; we expect that it will pave the way for a future use of the proposed architecture in real space missions.

## ACKNOWLEDGEMENTS

We would like to thank the European Commission and the members of the PERASPERA Programme Support Activity (ESA as coordinator, ASI, CDTI, CNES, DLR and UKSA) for their support and guidance in the ERGO activity. Finally, we would like to thank Alberto Medina for his work and support in the initial phases of this project and in the elaboration of this paper. The project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 730086.

## REFERENCES

1. **G. Rabideau, D.Tran, et al** "*Mission Operations of Earth Observing One with on-board autonomy"* IEEE International Conference on Space Mission Challenges for Information Technology. Pasadena, CA. July 2006
2. **E. Bernard, Douglas and Gamble, Jr., Edward B.** . "*Remote Agent Experiment DS1 Technology Validation Report".* Jet propulsion Laboratory. CIT, Pasadena, California
3. **Maimone, Jeffrey J. Biesiadecki and Mark W.** *The mars exploration rover surface mobility flight software driving ambition.* s.l. : IEEE Aerospace Conference (IAC), 2006.
4. **Estlin T. A. et. al,** "*AEGIS Automated Science Targeting for the MER Opportunity Rover":* ACM Transactions on Intelligent Systems and Technology (TIST), 2012, Vol. 3, p. 50.
5. **Klaus Schilling, Jürgen Walter, Samuel Kounev.** *Spacecraft Autonomous Reaction Capabilities, Control Approaches, and Self-aware Computing.* s.l. : Springer International Publishing, 2017.
6. **Medina, A., et al.** in "Aerospace Robotics II". Ottawa, Canada: "*Online of an Autonomy framework for space robotics*". Springer International Publishing pp 187-198, Switzerland 2015
7. **M. Winter et al** *ExoMars Rover Vehicle: Detailed Description of the GNC System***.** s.l. : Proceedings of Space Technologies in Robotics and Automation (ASTRA), 2015.
8. ESA/ESTEC. ECSS Secretariat. (ESA/ESTEC), "*ECSS-E-70-11 Space Segment Operability*" .August, 2005. Norwjik, the Nederlands.
9. **McGann, Connor, Rajan, Kanna and Py, Frederic** *T-REX., a model-based architecture for AUV Control.***.** International Conference on Automated Planning and Scheduling : s.n., 2007.
10. **Perrotin, M, Terraillon, J.-L. and Honvault, C.** "*Taste: towards a space system development framework"*, Oct. 2015.
11. TASTE WebSite http://taste.tools
12. ASN.1 Website: http://www.itu.int/en/ITU-T/asn1/Pages/asn1_project.aspx
13. AADL WebSite: http://aadl.info
14. **J. M. Delfa Victoria, N. Policella, Y. Gao, and O. V. Stryk.** "*Quijoteexpress - A novel APSI planning system for future space robotic missions*". In ASTRA, 2013. Noordwijk, the Netherlands
15. **E. Allouis, R. Marc, J. Gancet, Y.Nevatia, F.Cantori, R.U Sonsalla, M. Fritsche, J. Machowinski, T. Vogele, F.Comin, W. Lewinger, B. Yeomans, C. Saaj, Y.Gao, J.Delfa, P.Weclewski, K. Skocki, B. Imhof, S. Ransom, and L. Richter.** *Fp7 faster project - demonstration of multi-platform operation for safer planetary traverses.* s.l. : 13th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), 2015.7
16. **J. Benton, Amanda Coles and Andrew Coles.** *Temporal Planning with Preferences and Time-Dependent Continuous Costs.* s.l. : International Conference on Automated Planning and Scheduling (ICAPS), 2012.
17. **Helmert, M.** *The fast downward planning system.* s.l. : Journal of Artificial Intelligence Research (JAIR), 2006, Vol. 26.
18. **Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld and David Wilkins.** *PDDL - The Planning Domain Definition Language - Version 1.2.* 1998.
19. **Fratini, A. Cesta and S.** *The timeline representation framework as a planning and scheduling software development environment.* s.l. : 27th Workshop of the UK Planning and Scheduling Special Interest Group (PlanSIG), 2008.
20. **Bonet, Blai and Geffner, Hector.** *Planning as Heuristic Search.* s.l. : Artificial Intelligence, 2001, Vol. 129.
21. **Hoffmann, Jörg and Nebel, Bernhard.** T*he FF Planning System: (Fast) Plan Generation Through Heuristic Searc.* s.l. : Journal of Artificial Intelligence Research, 2001, Vol. 14.
22. **Helmert, Malte and Haslum, Patrik and Hoffmann, Jörg and Nissim, Raz.** *Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces.* s.l. : Journal of the ACM, 2014, Vol. 61.
23. **Katz, Michael and Domshlak, Carmel.** *Optimal Additive Composition of Abstraction-based Admissible Heuristics.* s.l. : ICAPS, 2008.
24. **Yang, Fan and Culberson, Joseph and Holte, Robert and Zahavi, Uzi and Felner, Ariel.** *A General Theory of Additive State Space Abstractions.* s.l. : Journal of Artificial Intelligence Research, 2008, Vol. 32.
25. **Pommerening, Florian and Helmert, Malte and Röger, Gabriele and Seipp, Jendrik.** *From non-*

*Negative to General Operator Cost Partitioning.* s.l. : AAAI, 2015.

26. **Woods, M. et al.** *Image based localisation and autonomous image asessment for a Martian aerobot.* Hollywood, CA : 9th International Symposium on Artificial Intelligence, Robotics and Automation in Space, 2008.
27. **Woods M., Shaw A., Barnes D., Price D., Long D., and Pullan D.** *Autonomous science for an ExoMars Rover-like mission.* s.l. : Journal of Field Robotics, 2009, Vol. 26, pp. 358-390.
28. **Paar G. et al.** *PRoViScout: a planetary scouting rover demonstrator.* s.l. : SPIE 8301, Intelligent Robots and Computer Vision XXIX: Algorithms and Techniques, p. 83010A-83010A-14, 2012.
29. **M., Wallace I. and Woods.** *MASTER: A Mobile Autonomous Scientist For Terrestrial and Extra-Terrestrial Research.* s.l. : 13th Symposium on Advanced Space Technologies in Robotics and Automation, ASTRA, 2015. 3.
30. **Wallace I., Woods M. and Read N** *labelmars.net: driving next-generation science autonomy with large high quality dataset collection*.**.** s.l. : 14th Symposium on Advanced Space Technologies in Robotics and Automation, 2017.
31. **Basu, A., Bozga, M., & Sifakis, J.** *Modeling heterogeneous real-time components in BIP., SEFM, pp. 3-12.* 2016. Fourth IEEE International Conference in Software Engineering and Formal Methods.
32. **Ben Rayana, S., et al.** *RTD-Finder: A Tool for Compositional Verification of Real-Time Component-Based Systems. Tools and Algorithms for the Construction and Analysis of Systems.* 2016. Tools and Algorithms for the Construction and Analysis of Systems: 22nd International Conference, TACAS. pp. 394-406.
33. **Nouri, A., Bensalem, S., Bozga, M., Delahaye, B., Jegourel, C., & Legay, A.** *Statistical model checking QoS properties of systems with SBIP. STTT,.* 2015. pp. 171-185.