Controlling and Testing a Space Instrument by an AI planner

MD. R-Moreno[1], M. Prieto[1], D. Meziat[1], J. Medina[2], C. Martin[2]

[1] *Departamento de Automática. Universidad de Alcalá. Ctra. Madrid-Barcelona, km 31,600 28871 Alcalá de Henares (Madrid). Spain.*
[2] *Departamento de Física. Universidad de Alcalá. Ctra. Madrid-Barcelona, km 33,600 28871 Alcalá de Henares (Madrid). Spain.*
e-mail: mdolores@aut.uah.es

Abstract. The PESCA instrument has been designed and built with the purpose of studying the Solar Energetic Particles and the Anomalous Cosmic Rays. It will be part of the Russian PHOTON satellite payload that is scheduled to launch in December of 2001. The instrument comprises two different blocks: the PESCA Instrument Amplification and Shaping Electronics (PIASE), for the amplification and analog to digital conversion, and the PESCA Instrument Control and Acquisition System (PICAS), for the control of the whole instrument and manage the communication with the satellite. An Electrical Ground Support Equipment (EGSE) software has been implemented using AI planning techniques to control and test the PESCA instrument and the communication process with the satellite. The tool allows complete and autonomous control, verification, validation and calibration of the PESCA instrument.

1 Introduction

The AI community has been working over the last decades in different and complex domains like robotics, satellites or logistics. Of special attention in the last few years has been the development of an autonomous architecture that can carry out on the ground a large number of functions such as planning activities, tracking the spacecraft's internal hardware, and ensuring correct functioning and repair when possible, without (or little) human intervention. In these new models of operations, the scientists and engineers communicate high-level goals to the spacecraft, these goals are translated into planning and/or scheduling sequences; then a continuous check of the spacecraft status is verified in order to detect any damage and finally execution is performed. It must also have the capability to understand the error occurred during the process of accomplishing the goals. Several approaches have faced this challenging domain as the New Millennium Remote Agent architecture [4] that reside in the flight processor of the Deep Space 1 (DS-1) capable of carrying out a complete mission with minimal commanding from Earth or the ASPEN system [6] that enables the Earth

Orbiting 1 (EO-1) mission, an Earth imaging satellite, to be controlled by a small operation team.

In this paper we present the integration of AI Planning Techniques into the EGSE software designed to allow autonomous/manual control of the instrument PESCA that will be on-board the Russian PHOTON satellite for the acquisition of Energetic and Anomalous particles. It will also allow testing, calibrating and validating the instrument (unit tests, integration tests and system tests).

The paper is structured as follow: section two presents an overview of the PESCA instrument. Then, we describe in detail the EGSE architecture. Finally conclusions and future work are outlined.

2 The PESCA instrument

The control system that has been mounted in the laboratory is depicted in figure 1. The main elements are: the PESCA instrument, an On Board Data Handling (OBDH) Emulator and, finally, a PC computer with the Electrical Ground Support Equipment (EGSE) software.

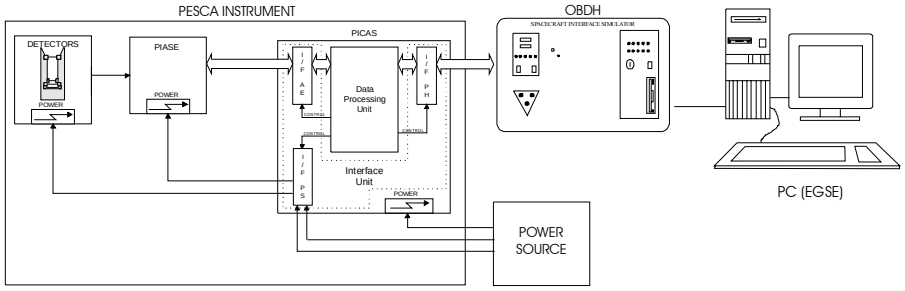

Fig. 1. Control System diagram.

The PESCA instrument [2, 5] is composed of a telescope, made up of four silicon ion implanted detectors, which register the particle crossing through the detectors, the PIASE for signal amplification and analog to digital conversion, and the PICAS, for the control of the whole instrument. PICAS is composed of a Data Processing Unit based on the MAS281 microprocessor and a set of interfaces with the rest of the instrument and the satellite. PICAS admits telecommands sent from ground for data acquisition and control parameter setting. The communication between ground and PICAS is performed through the On Board Data Handling (OBDH). The OBDH emulator acts as the satellite on board computer and it fulfills the OBDH European Space Agency standard. The communication between OBDH emulator and EGSE is performed through two serial RS232 ports, one for telecommands and one for telemetry.

In the other hand, the EGSE software allows the complete control over the instrument, being the end point of the system. It basically allows the telecommand sending and data reception, making possible the verification, validation and calibration of the PESCA instrument.

3 The EGSE Architecture

The EGSE is an object-oriented system that provides a reusable set of software components that make it feasible to extend the control to more instruments. The main features are:

- Telecommand sending. EGSE allows telecommand sending to both, OBDH and the PESCA instrument. By means of these commands, we can act over the PESCA instrument, changing the data acquisition parameters and other PESCA operation states such as detectors power on/off, coincidence/anticoincidence mode setting, threshold setting, etc. Through telecommand sending PESCA operation is tested.
- Telemetry reception. Telemetry consists of two kind of data: status data (housekeeping data) and scientific data. Through the housekeeping data, user may know the PESCA instrument status at any time (detector status, for instance) and therefore may act in consequence, for example in case of failure detection. In the other hand, scientific data contain the interest data, purpose of the instrument, that is, the energy lost in the detectors.
- Scientific Data processing. The user can make some processing over the received scientific data. This processing includes, among others, real time data visualization, data representation in different modes and data export to different formats.
- Planning/Scheduling. EGSE has an autonomous operation mode that avoids human supervision. In this mode, EGSE continuously checks housekeeping data and thanks to the integration of the planner PRODIGY [7] takes the specific actions to react against possible on flight failures.

3.1 Telecommand module

The user can send telecommands to the OBDH emulator or to the instrument itself. The communication with the OBDH emulator is performed through two RS232 serial ports completely configurable from the EGSE main window. Telecommands can be sent though a command line provided by the graphical interface or through a list control that displays the telecommands listed in a user defined file. This ASCII file, easily editable from a common text editor, contains the telecommand identifiers and the telecommand data bytes (header, body and terminator) to be sent through the serial port. The default file is the one for the PESCA instrument but the telecommand list can be modified through the interface. This gives versatility to EGSE because it is not restricted to a fixed set of telecommands of a given telecommand format, and therefore, it is not restricted to a specific instrument.

For every telecommand sent, the OBDH response is shown in order to know if the telecommand has been successfully delivered. EGSE allows to program the telecommand sending in time (Time Tag Telecommands). From a menu option, the user can program a list of telecommands to be sent in a specific time and date. This feature allows to program a telecommand sequence for delivery.

All telecommands issued and OBDH responses are registered in a log file for later inspection. Log files are stored in ASCII format, legible from any text editor.

3.2 Telemetry module

The user can select through the interface the number of data packets to collect. The telemetry data are received according to the loaded telemetry profile and are usually divided into housekeeping data and scientific data. Both data types are stored in different files in raw binary format. Telemetry profiles are user defined, allowing the EGSE configuration for other instruments. During the data acquisition, an integrity check is performed and bad data are discarded. EGSE also permits continuous acquisition, storing the corresponding data files.

3.3 Data processing module

After the data acquisition, both housekeeping data and scientific data may be processed through the data processing option. Housekeeping data contains the status data provided by the instrument according to the telemetry profile specified. These status data are useful to know if an error has occurred during the flight and are specific of the instrument.

Scientific data may be displayed in a histogram form, transforming EGSE in a multichannel analyser, or representing the data in two axes.

As stated in previous section, all the data received are stored in binary format. In order to allow the data export to other data processing programs, EGSE allows the data conversion into ASCII format.

3.4 Planning module

The EGSE program also has the capability of working in an autonomous mode thanks to the integration of AI planning techniques. We have used the deliberative independent planner PRODIGY [7] for this task. The problem solver is a non-linear planner that uses a backward chaining means-ends analysis search procedure with full subgoal interleaving. The planning process starts from the goals and adds operators to the plan until all the goals are satisfied. Three types of knowledge are the inputs of the planner. First, the domain theory that contains all the actions/telecommands represented by the operators. The Prodigy domain theory is based on the STRIPS representation originally proposed by Fikes and Nilsson [3].

One has first to define variables that are handled within the operator. For that, one needs to declare the type of each variable. The type hierarchy in PRODIGY forms a

tree structure. Each type in the hierarchy belongs to a super-type called root and each type may have several subtypes. All the components that are part of the control system are represented as a sub-type of the super-type. For example, that is, PICAS, OBDH, LOBT (Local On Board Time), etc are sub-types of the root and this is represented as:

```
(ptype-of DETECTOR :top-type)
(ptype-of LOBT :top-type)
(ptype-of PIASE :top-type)
```

Some types have an infinite number of instances (such as threshold setting), they are called infinite types. This type of variables are generated when PRODIGY needs to instantiate the corresponding operator. Figure 2 shows the syntax of a PRODIGY operator to turn on a detector.

With respect to the pre-conditions, this operator has two: the status of the detector, *off*, and the status of the PIASE, *on*. As effects, the detector will be *on*.

```
(Operator On_Detector
 (preconds
  ((<dect>  DETECTOR)
   (<Eanal> PIASE))
   (and(statis <dect> OFF)
       (on <Eanal>))
  (effects
   ()
   ((del(statis <dect> OFF))
    (add(statis <dect> ON)))))
```

Fig. 2 PRODIGY Operator to switch on a detector.

In some cases one telecommand correspond to one PRODIGY operator but in others, several telecommands are translated into one PRODIGY operator. The user can select from the interface what telecommand(s) correspond to each PRODIGY operator.

The second input to the planner is the problem, described in terms of an initial state and goals. Figure 3 shows a small fraction of the initial conditions and goals. The initial conditions are automatically generated from the telemetry reception process, that is, all knowledge that EGSE has about PESCA. The goal(s) are chosen from a list of goals available through the interface. In the example, one wants that PESCA acquire scientific data from any of the possible modes that the detectors can be, without any interruption. In PESCA there are defined four modes, in each mode a detector is not operative.

```
(and
   (on OBDH)
   (no-synchronize lobt)
   (off CDPUNOM)
   (off PIASE)
   (statis d1 OFF)
   (statis d2 OFF)
   (statis d3 OFF)
   (statis d4 OFF)))
   ...
(goal (scientific_acquisition data))
```

Fig. 3. Some initial conditions and goals for the problem of acquiring scientific data.

When there is more than one decision to be made at decision points, the third input to the planner, the control knowledge (declaratively expressed as control rules) could guide the problem solver to the correct branch of the search tree avoiding backtracking. There are three types of rules: selection, preference or rejection. One can use them to choose an operator, a binding, a goal, or deciding whether to apply an operator or continue sub-goaling. Figure 4 shows an example of a selection operator rule. It says that if the PIASE is *on* and the detector 2 is *off* or *broken* then chose the *ChangeMode2* operator. This control rule prunes the search tree and chooses the correct mode to detector 2.

```
(Control-Rule select-op-ChangeMode2
(IF   (and(current-goal (statis <piase> ON))
      (or(true-in-state (statis d2 OFF))
         (true-in-state (statis d2 BROKEN)))
      (TYPE-Of-Object d2 DETECTOR)))
(THEN select operator ChangeMode2))
```

Fig. 4. Control rule to select the detector Mode.

As a result, PRODIGY generates a plan with the sequence of operators/telecommands that achieve a state (from the initial state) that satisfies the goal(s). The obtained plan does not consider any optimisation with respect to resource use or availability, given that for PESCA it is enough to find a plan. However, the planner could obtain an optimal plan according to some criteria using the QPRODIGY version described in [1].

Every time a telecommand is sent following the plan generated by PRODIGY, an exhaustive check of the operator effects is performed in order to detect differences between the operator effects and the status data received after the telecommands sent. If any difference is found, a re-planning process starts. If a solution cannot be obtained, the program will abort and a log file will be generated with all the decision made by the planner. Figure 5 shows the architecture that integrates EGSE and the PRODIGY planner.

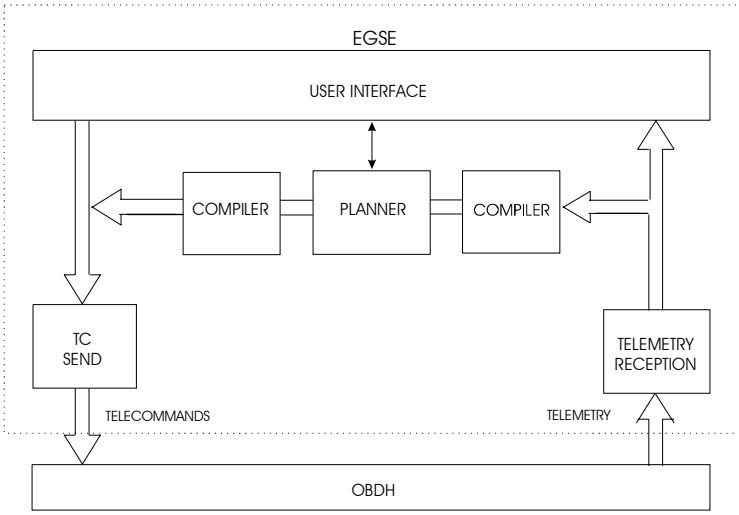

Fig. 5. EGSE-planner architecture.

For example, one of the possible actions in the plan if one wants to receive scientific data will be to turn on the detectors. If in this process, the program realise that the detector is still off, the planner will generate a new plan. In this new plan, one of the actions will be to send the same telecommand three more times (sometimes the problem is not due to a detector damage but a bad transmission/reception). If this fails, a new plan is generated to find a different mode of acquisition but if no valid solution can be obtained, the program will abort and a log file will be generated with dates and decisions made by the planner. It is also possible, when EGSE is running in *auto* mode, that the user interacts or inhibits the planner decision.

This module has also let us to automatically perform the unit test, integration test and system test that PESCA needs before the PHOTON integration. All the tests need to be performed several times, varying the types of particles, the detector mode, the target and the redundant components. Thanks to the planning process we have successfully run the test with the minimum human cost.

4 Conclusions

A program for the control and test of the PESCA instrument has been presented. The software has been successfully proved in the laboratory, and it has become a key part in PESCA testing. These tests are essential for the integration with PHOTON satellite. EGSE also has a great versatility, and it may be extended to other instruments modifying several parameters.

5 Acknowledgements

This work has been supported by Spanish CICYT (grant ESP99-1066-C02), by the European Community-Access to Research Infrastructure action of the Improving Human Potential Programme, No HPRI-CT1999-00019 and by the NATO grant PST.CLG.977003. We also want to thanks Daniel Borrajo for the PRODIGY support.

References

1. Borrajo D., Vegas S. and Veloso M. Quality-based Learning for Planning. Working notes of the IJCAI'01 Workshop on Planning with Resources. IJCAI Press. Seattle, WA (USA). August 2001.
2. del Peral, L., Bronchalo, E., Medina, J., Rodríguez Frías, M.D., Sánchez, S., and Meziat, D. An Electronic Device that Suits Space Research Requirements for Ion Detection. IEEE Transaction on Nuclear Science, 44, 1442-1447, 1997.
3. Fikes R. E, Nilsson N. J. STRIPS: a new approach to the application of theorem proving to problem solving . In Artificial Intelligence, 2:189-208, 1971.
4. Muscettola N., Smith B., Fry Charles, Chien S., Rajan K., Rabideau G. And Yan D. On-Board Planning for New Millenium Deep Space One Autonomy. In Proceedings of the IEEE Aerospace Conference, Snowmass, CO, 1997.
5. Prieto, M., Martín, C., Quesada, M., Meziat, D., Medina, J., Sánchez, S., and Rodríguez-Frías, M.D. Control and acquisition system of a space instrument for cosmic ray measurement. Nuclear Instruments and Methods in Physics Research, A443, 264-276, 1999.
6. Sherwood R., Govindjee A., Yan D., Rabideau G., Chien S. and Fukunaga A. Using ASPEN to Automate EO-1 Activity Planning. Proceedings of the 1998 IEEE Aerospace Conference, Aspen, CO, March 1998.
7. Veloso M., Carbonell J., Perez A., Borrajo D., Fink E., and Blythe J. Integrating planning and learning: The PRODIGY architecture. Journal of Experimental and Theoretical AI, 7: 81-120, 1995.