JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 5, July 2008

Multi-Agent Planning and Execution Approach for Controlling Multiple Satellites

Adriana Carniello*, Andreia Carniello†, Luciana S. Cardoso‡, Mauricio G. V. Ferreira§, and José Demisio S. da Silva¶
National Institute for Space Research, São José dos Campos, SP, 12227-010, Brazil

DOI: 10.2514/1.29252

Reducing the costs of space operations is an increasing demand which may be achieved by automating space operation planning and execution. Towards this aim, this work proposes a Multi-Agent Space Operation Automation architecture. Multi-Agent Space Operation Automation architecture manages ground resource allocation for multi-satellite tracking, plans the satellite control operations, and automates plan execution. For managing the sharing of ground resources, it identifies satellites with time-conflicting visibility periods and reduces the least priority satellite tracking. For the planning process, Multi-Agent Space Operation Automation architecture automatically generates planning problem descriptions and analyzes whether a satellite tracking period is sufficient to achieve all the tracking goals. If there is insufficient time, it allows the elimination of lower priority goals. The satellite control planning problem was specified in PDDL 2.2 and the satellite control plan generated according to the temporal planning paradigm.

I. Introduction

IN the field of satellite control, there has been a general interest in automating the space operation planning and execution. The automation of space operations represents a way of reducing the in-orbit satellite maintenance costs as well as increasing the satellite control system reliability.

The aim of the present paper is to support the personnel responsible for satellite control with the objective of designing new solutions to fit Space Program budgets concerning the launching of new satellites.

The automation described in this paper is strongly concerned with the tasks of controlling multiple satellites, generating satellite control plans, and automating plan execution. To realize these tasks, there is a set of restrictions to be managed, such as the sharing of ground stations for multi-satellite tracking and time-critical operations to be executed during the limited period of time in which a satellite becomes visible to a ground station.

Ground stations are the sites at which the antennae that capture satellite signals are located. The time window in which a satellite becomes visible to a ground station is named satellite visibility period or satellite pass [1].

Concerning the task of controlling multiple satellites that share the use of ground stations, the visibility period of one satellite may conflict with the visibility period of another. When this situation occurs, the conflicting portion

Received 11 December 2006; revision received 5 March 2008; accepted for publication 20 March 2008.

* Doctorate student, Applied Computing Postgraduate Program (CAP), Av. dos Astronautas 1758 Jd. Granja, adcarnie@lac.inpe.br.
† Doctorate student, Applied Computing Postgraduate Program (CAP), Av. dos Astronautas 1758 Jd. Granja, ancarnie@lac.inpe.br.
‡ Master researcher, Ground System Department (DSS), Av. dos Astronautas 1758 Jd. Granja, luciana@dss.inpe.br.
§ Doctor researcher, Satellite Control and Tracking Center (CRC), Av. dos Astronautas 1758 Jd. Granja, mauricio@ccs.inpe.br.
¶ Doctor researcher, Applied Mathematics and Computing Associated Laboratory (LAC), Av. dos Astronautas 1758 Jd. Granja, demisio@lac.inpe.br.

of the lower priority satellite's visibility period must be cancelled. Canceling a satellite conflicting visibility period means reducing the time window in which the satellite will be tracked by the ground station. This reduction in the time window generates a time restriction that must be considered in the satellite control plan generation.

Satellite control plans, named Flight Operation Plans (FOP), contain all the operations related to the control of in-orbit satellites. FOP generation requires that these operations be inserted at specific instants of time to achieve the satellite tracking goals. These goals comprise, for instance, calibration measure execution, telemetry reception, telecommand sending, and distance and speed measure execution.

Multi-Agent Space Operation Automation (MASOA) architecture presents a solution by planning the ground resource utilization to meet multiple satellite trackings by generating a FOP for each satellite and by automating FOP execution. For planning the ground resource utilization, it considers the satellite priority rankings to reduce their tracking periods when conflicts occur. For plan generation, it reasons whether or not there is sufficient time to achieve all the tracking goals and allows disconsidering goals if there is insufficient time. For plan execution, MASOA architecture restricts the human satellite operators to the execution monitoring and allows their intervention in case of anomalies.

Section II relates some fundamental concepts of the artificial intelligence (AI) planning area with the satellite control domain. Section III introduces the agent-oriented software engineering methodology applied to deduce MASOA architecture agents. Section IV presents MASOA architecture agents, defined by a stepwise process of different views, and describes their services and the architecture general behavior. Section V documents results obtained with the implementation of MASOA prototype tool. Following a discussion of related research in Sec. VI, the paper concludes in Sect. VII with a summary of this paper's contributions.

II. Planner Agent

Planning problems involve a set of initial states, a set of goals, and the corresponding actions that contribute to achieve these goals. A planner agent is an agent responsible for solving planning problems. This type of agent represents a planning problem by propositional/first-order representations that allow effective heuristic derivations and the development of planning algorithms (planners) used to plan a sequence of actions, the execution of which will lead to the desired goals.

The representation of planning problems has been a concern since 1971 when Fikes and Nilsson developed the STRIPS language [2]. From this time on, other researchers have proposed planning problem representation languages based on STRIPS, aiming at developing a more expressive language for real planning problems.

In 1998, the AI planning groups made an attempt to standardize a language for real planning problem description by proposing Planning Domain Description Language [3,4]. PDDL has been used as the standard language in international planning competitions, allowing planning problems to be represented in a comparable notation and planner performance to be evaluated. In its version 2.2, PDDL currently allows planning problem modelers to specify actions with duration and deterministic unconditional exogenous events, which are facts that will become true or false at time points that are known to the planner in advance, independently of the actions that the planner chooses to execute. These two features are very important for modeling satellite control planning problems owing to the fact that space operations have duration and must be adjusted to the well-defined time windows of the satellite tracking periods.

MASOA architecture adopts PDDL for it is the standard language in the planning research community and for we aim at using a research planner as a black box component in our larger intelligent control system. Adopting a planner developed in the research community has the advantage of assuring the use of a high-performance model. The AI planning competitions led to the development of many high-performance planners that are readily available. Another advantage is that when integrating a research planner into a larger application, one generally has the advantage of a crisply defined and comprehensible algorithm, often published and peer reviewed [5].

PDDL separates the description of parameterized actions (planning domain behavior) from the description of the initial conditions and the goals to be achieved (problem instance). Therefore, PDDL distinguishes the planning domain description from the problem description, which together represent a planning problem. By separating these definitions, PDDL allows the same planning domain description to be used by several different problem descriptions to produce different planning problems for the same planning domain.

```
1   (define (domain DOMAIN_NAME)

2     (:requirements [:equality][:typing][:fluents][:durative-actions][:timed-initial-literals])

3     (:types TYPE_1_NAME  …  TYPE_N_NAME)

4     (:predicates (PREDICATE_1_NAME  ?PAR1 - TYPE_OF_PAR1 …  ?PARN - TYPE_OF_PARN)
5                          . . .
6                  (PREDICATE_N_NAME  ?PAR1 - TYPE_OF_PAR1  …  ?PARN - TYPE_OF_PARN)
7     )

8     (:functions (FUNCTION_1_NAME  ?PAR1 - TYPE_OF_PAR1  …  ?PARN - TYPE_OF_PARN)
9                          …
10                (FUNCTION_N_NAME  ?PAR1 - TYPE_OF_PAR1  …  ?PARAM_N - TYPE_OF_PARN)
11   )

12   (:durative-action ACTION_1_NAME
13       [:parameters (?PAR1 - TYPE_OF_PAR1 …  ?PARN - TYPE_OF_PARN)]
14       :duration(= ?duration (FUNCTION_NAME ?PAR1 - TYPE_OF_PAR1 … ?PARN - TYPE_OF_PARN))
15       [:condition (CONDITION_FORMULA)]
16       [:effect (EFFECT_FORMULA)]
17   )

18   (:durative-action ACTION_N_NAME
19      …
20   )

21 )  ;;Comentary - end of domain definition
```

Fig. 1 Format of a PDDL Domain Description file.

A Planning Domain Description file contains the set of elements necessary to model the planning domain behavior: domain types, functions, predicates, and actions. The format of a domain description file is presented in Fig. 1 in a semiformal notation. Elements in brackets are optional and ellipses are used to omit intermediate predicate or parameter declarations. Lines are numbered simply to facilitate the explanation of the PDDL format.

In line 15, a CONDITION_FORMULA may be: 1) an atomic formula: (PREDICATE_NAME ARG1…ARGN), where the predicate arguments must be parameters of the action; or 2) a conjunction of atomic formulae: (and ATOM1…ATOMN). In line 16, an EFFECT_FORMULA may consist of: 1) an added atom: (PREDICATE_NAME ARG1…ARGN), where the predicate arguments must be parameters of the action; 2) a deleted atom: (not (PREDICATE_NAME ARG1…ARGN)); or 3) a conjunction of atomic effects: (and ATOM1 … ATOMN).

Figure 2 presents a simplified code extracted from the MASOA architecture domain definition that exemplifies the application of the domain description file format to the satellite control domain.

In the example code given in Fig. 2, the first line declares DOMAIN_NAME = satellite and the second line declares a set of requirement flags for the domain. The use of these flags gives the opportunity for the planner to reject attempts to plan with domains that make use of more advanced features of the language than the planner can handle [3]. The requirement :equality means the domain can use the predicate =, interpreted as equality; the requirement :typing means the domain uses types; the requirement :fluents allows the use of numeric fluents; and the requirements :durative-actions and :timed-initial-literals means the domain accepts actions with duration and deterministic unconditional exogenous events, respectively.

Type names are declared in line 3 before they are used in :predicates and :functions declarations (in lines 4 and 8). The types defined for the satellite domain are PassNumber, SatCode, and GSCode that represents the pass number of a specific satellite, the satellite code, and the ground station code in charge of the satellite tracking, respectively.

In line 4, predicates are declared. For instance, in the predicate (ReducedTrack ?pass - PassNumber ?sat - SatCode ?gs - GSCode) the parameters pass, sat, and gs are of types PassNumber, SatCode, and GSCode, respectively. When evaluated true, this predicate means that a specific satellite pass over a certain ground station had its tracking period reduced owing to a time conflict with a higher priority satellite pass.

```
1    (define (domain satellite)

2      (:requirements :equality :typing :fluents :durative-actions :timed-initial-literals)

3      (:types PassNumber SatCode GSCode)

4      (:predicates (GSAvailable ?gs - GSCode)
5                   (ReducedTrack ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
6                   (PassTrack ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
7                   (LOSAOSTimeOk ?pass - PassNumber ?sat - SatCode ?gs - GSCode)
8      )

9      (:functions (LOSAOSTime ?pass - PassNumber ?sat - SatCode ?gs - GSCode) )

10    (:durative-action LOSAOSTimeWindow
11       :parameters (?pass - PassNumber ?sat - SatCode ?gs - GSCode)
12       :duration (= ?duration (LOSAOSTime ?pass ?sat ?gs))
13       :condition ( and (at start (ReducedTrack ?pass ?sat ?gs))
14                                (at start (GSAvailable ?gs))
15                             (over all (PassTrack ?pass ?sat ?gs))
16                  )
17       :effect (and (at start (not (GSAvailable ?gs)))
18                          (at end (GSAvailable ?gs))
19                          (at end (LOSAOSTimeOk ?pass ?sat ?gs))
20             )
21    )

22 )
```

Fig. 2 Part of the PDDL Domain Description file generated for describing the MASOA architecture planning domain.

In line 9, the function (LOSAOSTime ?pass - PassNumber ?sat - SatCode ?gs - GSCode) is declared. This function associates objects pass (of type PassNumber), sat (of type SatCode), and gs (of type GSCode) with a numeric value corresponding to the time that must be reserved to adjust the ground station settings to begin the satellite tracking.

In line 10, a durative action named LOSAOSTimeWindow is declared. A durative action is assigned an execution duration and is associated to a condition and an effect, which are conjunctions of literals that declare the environment states before and after the action execution, respectively.

All conditions and effects of a durative action must be temporally annotated. The annotation of a condition makes explicit whether the associated proposition must hold at the start of the interval (the point at which the action is applied), the end of the interval (the point at which the final effects of the action are asserted) or over the interval from the start to the end (invariant over the duration of the action). The annotation of an effect clarifies whether the effect is immediate (it happens at the start of the interval) or delayed (it happens at the end of the interval) [3].

The durative action LOSAOSTimeWindow is executed when a satellite pass over a ground station has its tracking period reduced owing to a time conflict with a higher priority satellite pass, thus requiring reserving a time window in the plan to switch ground station configurations between the loss of signal (LOS) of the previous satellite (the higher priority one) and the acquisition of signal (AOS) of the current satellite (the lower priority one). The environment states that must hold in the action condition are (at start (ReducedTrack ?pass ?sat ?gs)), (at start (GSAvailable ?gs)), and (over all (PassTrack ?pass ?sat ?gs)), which mean the satellite tracking is a reduced-period one, the ground station is available for starting a new tracking, and the satellite tracking is active, that is can be started and has not been concluded, respectively. The effect propositions that take place when the condition is true are (at start (not (GSAvailable ?gs))), (at end (GSAvailable ?gs)), and (at end (LOSAOSTimeOk ?pass ?sat ?gs)), which means that at the start of the action interval the ground station is not available because configurations are being set and, at the end of the interval, the ground station becomes available and the LOS-AOS time window is finally reserved.

A Problem Description file contains the objects present in the problem instance, the initial state and the goal. The format of a Problem Description file is presented in Fig. 3 in a semiformal notation.

The initial state description in line 4 is simply a list of all the ground atoms that are true in the initial state. All other atoms are by definition false. In line 5, the goal description is a formula of the same form as an action condition.

```
(define (problem PROBLEM_NAME))
    (:domain DOMAIN_NAME)
    (:objects OBJ1 OBJ2 … OBJN)
    (:init ATOM1 ATOM2 … ATOMN)
    (:goal CONDITION_FORMULA)
)
```

Fig. 3 Format of a PDDL Problem Description file.

```
(define (problem CBASCD1Pass60482-5Problem)

    (:domain satellite)

    (:objects
        PASS60482-5 - PassNumber
        SCD1 - SatCode
        CBA - GSCode
    )

    (:init
         (= (LOSAOSTime PASS60482-5 SCD1 CBA) 120.0)
         (GSAvailable CBA)
         (PassTrack PASS60482-5 SCD1 CBA)
         (ReducedTrack PASS60482-5 SCD1 CBA)
    )

    (:goal (LOSAOSTimeOk PASS60482-5 SCD1 CBA))
)
```

Fig. 4 Part of the PDDL Problem Description file generated for describing a MASOA architecture planning problem instance.

Unlike action conditions, however, the initial state and goal descriptions should be ground, meaning that all predicate arguments should be object names rather than parameters. All predicates used in the initial state and goal descriptions should naturally be declared in the corresponding domain.

Figure 4 presents a Problem Description file generated for the satellite domain definition of Fig. 2. The Problem Description file declares the objects, the initial state and the goal to be reached in a planning problem instance for the satellite domain.

In the satellite domain a planning action consists of a space operation that is concerned either with the control of satellite on-board equipments status or with obtaining the satellite exploitation expected products. Planning goals are represented by tracking goals to which we propose the assignment of priorities according to their execution relevance in the domain.

III. Agent-Oriented Analysis

For deducing MASOA architecture agents an agent-oriented software engineering methodology named MESSAGE/UML [6] was applied. This methodology has the concept of agent as its center and covers multi-agent system (MAS) analysis and design. There are other agent oriented methodologies such as Gaia [7] and Mas-Common-Kads [8] that also define models for both analysis and design. Gaia analysis models are based on well-defined concepts, but these only represent a subset of the concepts required for agent-oriented analysis and the design models are not clearly explained. Mas-Common-Kads has comprehensible models, but lacks a unifying semantic framework and notation.

Methodology for Engineering Systems of Software Agents (MESSAGE) [6] extends the basic UML concepts of class and association with knowledge-level-agent centric concepts which are: agent, organization, role, resource, task, interaction, goal, and message. An Agent is an automatic autonomous entity that is capable of performing some useful function. The autonomy means that an agent's actions are not solely dictated by external events or interactions but also by its own motivation, which is captured in an attribute-named purpose. The purpose influences whether an agent agrees to a request to perform a service (agent function) and also the way it provides the function. Software Agent and Human Agent are specializations of Agent. An Organization is a group of Agents working together to a common purpose. It is a virtual entity in the sense that the system has no individual computation entity corresponding to an organization.

The distinction between Role and Agent is analogous to that between Interface and (object) Class: a Role describes the external characteristics of an Agent in a particular context. Roles can also be used as indirect references to Agents. A Resource is used to represent non-autonomous entities such as databases or external programs used by Agents.

A Task is a knowledge-level unit of activity with a single primer performer. An Interaction by definition has more than one participant and a purpose which the participants collectively must aim to achieve. The purpose typically is to reach a consistent view of some aspect of the problem domain, to agree terms of a service or to exchange to results of one or more services. A goal is intrinsic to an agent identity and therefore is derived from its purpose. A Message is an object communicated between Agents. The attributes of a Message specify the sender, the receiver, a speech act (categorizing the Message in terms of the intent of the sender) and the content (an object).

The methodology proposes views to help the modeler focus on the above set of knowledge-level concepts. For deducing and modeling MASOA architecture agents, three views were applied: Organization view, Goal view, and Agent view. The Organization view shows concrete entities (organizations, roles, and resources) in the system and its environment and coarse-grained relationships between them, such as acquaintance relationships. An acquaintance relationship indicates the existence of at least one Interaction involving the entities concerned. The Goal view shows goals and the dependencies among them, for example the decomposition of a high level goals into subgoals. The Agent view focuses on the individual agents. For each agent it uses a textual Agent Schema to describe the agent characteristics such as what goals it is responsible for, what events it needs to sense, what resources it controls, what tasks it knows how to perform, and its behavior rules.

MASOA architecture was modeled by an agent-based approach owing to this approach appropriateness for modeling specific situations that characterize the architecture requirements. According to O'Malley and DeLoach [9], an agent-based approach is appropriate to systems that must perform well in situations where it is not practical or possible to specify their behavior on a case-by-case basis and to systems that require cooperation. In the satellite control environment it is not practical to foresee all possible situations that may arise in the multi-satellite plan generation and therefore goal oriented behavior is recommended. Moreover, the space operation planning for exceptional situations requires negotiation, for example for defining the set of operations (planning actions) to be inserted into a FOP for a satellite tracking period that suffered a time reduction or for a satellite tracking period that is the first/last of a sequence of passes.

IV. MASOA Architecture

MESSAGE/UML proposes producing system analysis models by stepwise refinement. The modeling process starts building the Organization and the Goal views, which both act as input for creating the Agent view. The following subsections present MASOA architecture analysis models.

A. Organization View

The Organization view considers the system to be developed as a black box and focuses on its relationships to the entities in the system environment. Figure 5 shows the main acquaintance relationships in MASOA architecture.

MASOA architecture is composed of a Mission Planning (MP) system organization that interacts with two Human Agents, databases, and external organizations to the architecture. The Operation Director (OPDIR) Human Agent is in charge of the satellite in-orbit operations. This agent defines the set of possible in-orbit routine operations, the operations for controlling orbit and attitude, the operation initial priorities, and so on. The Satellite Engineer (SATENG) Human Agent is a satellite specialist in charge of the control of a specific satellite. They are the professionals allowed to interfere in the execution of a satellite FOP when problems arise.

The MP system organization also interacts with databases. It retrieves Passage Visibility Prevision (PVP) files from the PVP Database. A Pass Visibility Prevision (PVP) file contains data about the future passes of one satellite over a specific ground station for a seven-day period and provides, for instance, predictions about the ground antenna

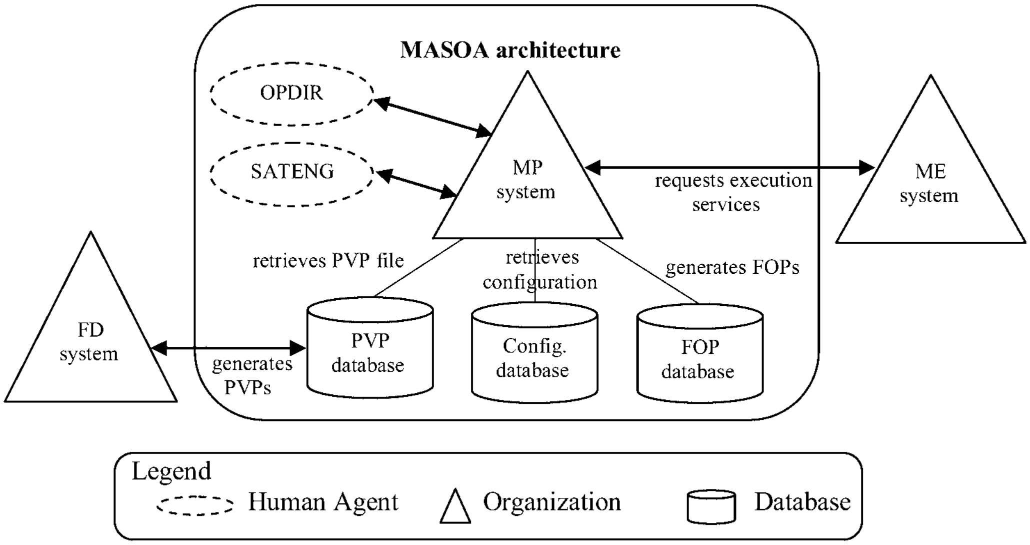


Fig. 5 MASOA architecture organization view (acquaintance relationships).

pointing data, the date and the AOS and end of signal (EOS) times for the satellite passes over the ground station, etc. This file is generated by the Flight Dynamics (FD) system. which is an external organization to the MP system.

The MP system organization retrieves data from the configuration database, which stores general data about the concerning set of satellites and ground stations. Such data are not dependent on a single satellite pass but are general to all the passes related to a specific satellite and ground station (e.g., a value that represents the satellite pass minimal duration, the calibration action duration that is generally inserted before the beginning of the satellite passes, the complete list of telecommands the satellite accepts, the silent zone initial and final angles, etc.).

The MP system organization generates FOPs and stores them into the FOP database. FOP contain satellite control operations (planning actions) that are executed when an internal agent to the MP system organization (an executor agent (EA)) requests execution services from the Mission Execution (ME) system. The ME system organization provides execution services related to the FOP operations such as telecommand sending and telemetry receiving.

B. Goal View

The main goal of the MP system organization is broken down according to the goal decomposition diagram in Fig. 6. The main goal "Mission controlled" is split into the subgoals "Flight operations planned" and "Flight operations executed" and is satisfied when both subgoals are satisfied. Thus, the "Mission controlled" goal is satisfied when flight operations are planned and executed. The "Flight operations planned" goal on its turn is satisfied when: 1) the satellite tracking period is defined after resolving the time conflicts that may arise with other satellite passes concerning the allocation of the same ground station and the conflicts that may arise when a single satellite pass or part of it is simultaneously visible to more than one ground station; 2) the planning problem is defined, comprising the definition of the PDDL planning domain description file and the PDDL Problem Description file; and 3) the FOP operations are generated by a planner mechanism that uses the PDDL files as input for the planning process.

The "Planning problem defined" goal is satisfied when the planning domain description file that contains the planning types, predicates, functions, and actions is generated. When it is finally generated, the Problem Description file contains the planning problem objects, initial states, exogenous events, and goals. The Problem Description file generation satisfies the goal "Planning problem description defined" which depends on the goal satisfaction "pass configuration provided" and "planning goals prioritized" to be able to generate the Problem Description file.

C. Agent View

This view focuses on the MP system organization individual agents. The delegation structure diagram in Fig. 7 shows how the subgoals obtained from the decomposition of the main goal of the MP system organization are assigned

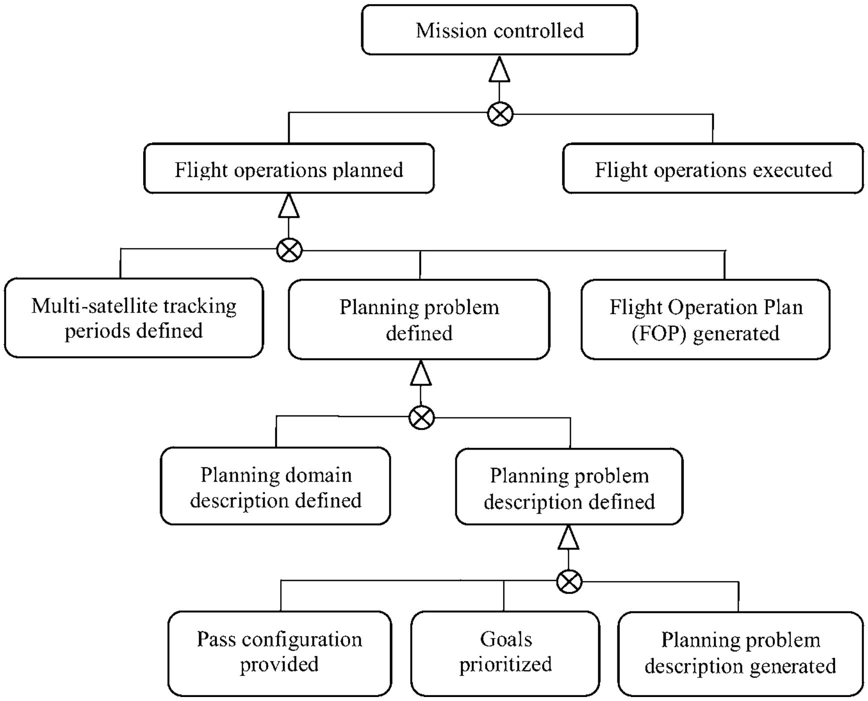

Mission controlled
Flight operations planned
Flight operations executed
Multi-satellite tracking periods defined
Planning problem defined
Flight Operation Plan (FOP) generated
Planning domain description defined
Planning problem description defined
Pass configuration provided
Goals prioritized
Planning problem description generated

Fig. 6 MP system organization goal view (goal decomposition diagram).

to the agents included in the organization. Clearly this diagram is strictly related to both the goal decomposition diagram and the organization diagram by showing the decomposition of the MP System Organization goal and the agents inside the organization.

Considering that this view focuses on the MP system organization individual agents, MESSAGE methodology proposes the specification of an Agent Schema for describing the characteristics of each software agent. The following subsections describe the MP system organization agent schemas.

1. Tracking Planner Agent (TPA)

This agent generates tracking plans (TP) that define which satellites will be tracked by a ground station, the order they can be tracked, and the tracking duration.

The time window a satellite can be really tracked is named the satellite tracking period which may comprise its whole visibility period or just a portion of it. A satellite tracking period comprises just a portion of its visibility period when the visibility period of a second satellite conflicts with that of the first one. In this case, the tracking period of the satellite with low priority is reduced to avoid time conflict. Thus, the Tracking Planner Agent (TPA) manages the problem of multi-satellite tracking with conflicting visibility periods (concerning the same ground station) by shortening the lower priority satellite tracking. Nevertheless, a satellite tracking period must satisfy a minimal pre-defined duration even after its reduction. Satellite tracking periods shorter than this pre-defined duration are cancelled.

When a satellite is visible to two or more ground stations simultaneously, TPA analyses the ground station priorities. In Brazil, for instance, Alcantara station supports the satellite acquisition phase whereas Cuiaba station supports the subsequent mission phases. Therefore, Cuiaba station has a higher priority to track a satellite in the routine phase. Nevertheless, Alcantara station plays the role of a backup station and can track a satellite when problems occur to Cuiaba station.

Table 1 presents the TPA schema which details the agent purpose and goal, the events it senses, the resources it uses, and its behavior defined by the agent task workflow.

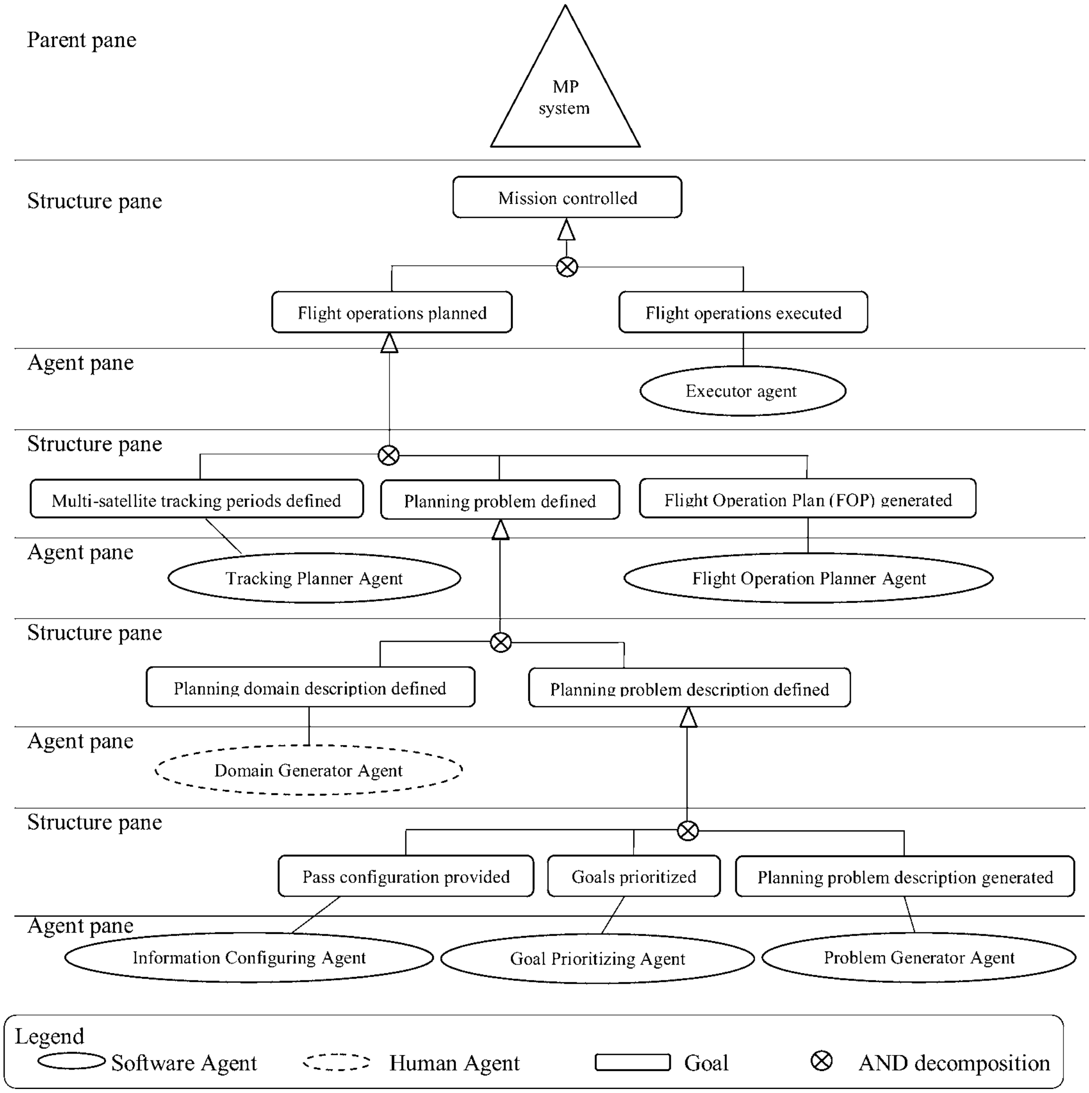


Fig. 7 MP system organization agent view (delegation structure diagram).

2. *Information Configuring Agent (ICA)*

This agent is responsible for managing the configuration database by allowing data insertion, edition, and deletion into the database and for generating satellite pass configurations. It generates configuration data for a satellite specific pass and these data allow the future generation of the pass objects, initial states, and goals (PDDL Problem Description file) by another agent (the Problem Generator Agent (PGA)). The Information Configuring Agent (ICA) extracts information from the configuration database and the PVP files to generate such data. Table 2 presents the ICA schema.

We present an example of the ICA behavior for generating information about a satellite specific pass. ICA determines the last telecommand sending time of a satellite specific pass according to the following behavior rules:

1) allow storing into the configuration database the ground antenna minimal elevation for telecommand sending from a specific ground station to the satellite;
2) extract from the corresponding PVP file the satellite pass last time related to this antenna minimal elevation;
3) attribute this time value to a pass configuration variable that represents the last telecommand sending time.

Table 1 Tracking Planner Agent schema

Agent schema	Tracking Planner Agent (TPA)
Purpose	Schedule the tracking of a set of satellites to a set of ground stations
Goal	Multi-satellite tracking periods defined
Events	End of the Pass Visibility Prevision (PVP) file generation for the concerning set of satellites and ground stations (by FD system)
Resources	PVP files database and configuration database. A PVP file is generated for a period of 7 consecutive days for a specific ground station and satellite. It provides the TPA, the date, and the AOS and EOS times for the satellite passes over the ground station during the one-week period. The configuration database provides the TPA, the satellite, and ground station priorities and the minimal duration that a satellite pass must have to be tracked
Task workflow	Retrieve from the PVP database the PVP files of a specific ground station for a pair of distinct satellites for the same one-week period For each PVP file, calculate the satellite pass durations. The satellite passes are represented by different orbit numbers Compare the satellite pass durations with a minimal time value retrieved from the configuration database. This minimal time value represents the minimal duration that a satellite pass must have to be tracked If the satellite pass duration is lower than this minimal time value, cancel satellite tracking owing to short duration. Otherwise, schedule the satellite pass to be tracked Retrieve the satellite priorities from the configuration database Compare a pair of PVP files (from different satellites for the same ground station) to verify if some time conflict arises between two passes If a time conflict occurs, cancel the pass conflicting part of the least priority satellite For a reduced tracking pass, compare the satellite tracking duration with the minimal time value retrieved from the configuration database If the satellite tracking duration is lower than this minimal time value, cancel satellite tracking owing to short duration. Otherwise, schedule the satellite reduced tracking pass to be tracked Store the satellite orbit numbers that identify the reduced tracking passes, the multi-satellite tracking order concerning a ground station, and the satellite tracking dates and initial and final times.

3. *Goal Prioritizing Agent (GPA)*

This agent acts when exceptional situations occur in the satellite control environment. It prioritizes satellite tracking goals to manage three exceptional situations: a satellite tracking period reduction performed by the TPA to avoid time conflict with another satellite and the first and the last passes of a satellite sequential pass group. A satellite sequential pass group is the satellite set of consecutive passes visible to a ground station. In the satellite control environment, there may be satellites that get visible to ground stations during consecutive passes (orbits) and get out of visibility for the rest of the day.

In the case of a satellite tracking period reduction, the satellite tracking period is generally not enough to execute all the tracking goals. The Goal Prioritizing Agent (GPA) by assigning priorities to goals, makes it possible to consider solely the most relevant goals for planning. The aim is to fit the planning actions to the short-time tracking period.

In the case of a first or last pass of a satellite pass sequence group, there are high priority extra-goals that must be considered for the Problem Description file generation and goals that must have their priorities decreased for not being considered in this file generation.

To implement this solution, each goal must be annotated with a priority which comprises a symbolic value that might change from one satellite tracking to another, to better specify the need for the goal execution in the next tracking period. Priority values range from 0 to 2: 0 islow priority goal, 1 is normal priority goal, and 2 is high priority goal. The PGA just considers the normal and high priority goals in the goal generation process by their order of relevance. Table 3 presents the GPA schema.

Table 2 Information Configuring Agent schema

Agent schema	Information Configuring Agent (ICA)
Purpose	Generate information about a satellite specific pass
Goal	Pass configuration provided
Events	Message from the TPA informing the ending of the multi-satellite tracking period definition
Resources	PVP files database, configuration database, and the satellite free-of-conflict tracking periods. The PVP file, generated for a period of 7 consecutive days for a specific ground station and satellite, provides the ICA predictions about the ground antenna pointing data for the satellite passes over the ground station. The configuration database provides the ICA general information (configuration) about the satellites and ground stations. The satellite free-of-conflict tracking periods generated by the TPA provides the ICA the satellite passes, represented by their orbit numbers, that will indeed be tracked by ground stations.
Task workflow	Allow the following operations into the configuration database: data insertion, data update and data deletion Obtain from the TPA the satellite passes (orbit numbers) that are Scheduled to be tracked by ground stations For each satellite tracking, generate the pass configuration data.

Table 3 Goal Prioritizing Agent schema

Agent schema	Goal Prioritizing Agent (GPA)
Purpose	Prioritize goals to generate a plan
Goal	Goals prioritized
Events	Message from the Tracking Planner Agent (TPA) informing the occurrence of an exceptional situation: a satellite tracking period reduction performed to avoid time conflict with another satellite or the first/last pass of a satellite sequential pass group.
Resources	The configuration database. The configuration database provides the GPA the goal priorities.
Task workflow	When a satellite tracking period is a reduced one or the first/last pass of a satellite sequential pass group: 1) add new high priority goals to the set of goal predicates retrieved from the Planning Domain Description file (previously generated by the Domain Generator Agent). The addition of these new goals makes them be further considered by the PGA for the Problem Description file generation (tracking goal generation); 2) decrease goal priorities to the low level so that these predicates will no longer be considered by the PGA for the Problem Description file generation (tracking goal generation).

We present examples of the GPA behavior for managing the three exceptional situations that occur in the satellite control environment: a satellite tracking period reduction, and the first and last passes of a satellite sequential pass group.

When it is a satellite reduced tracking period, GPA acts as the following behavior rules:

1) Decrease the “execute calibration operation” goal priority to the low level so that this goal will not be considered for the satellite tracking goal generation. The calibration operation consists of sending a signal from the satellite Control Center to test the ground station equipments before the satellite tracking beginning.
2) Add the new high priority goal “reserve LOS-AOS time window” into the set of goal predicates that will be further used to generate the satellite tracking goals (Problem Description file). This goal assures reserving a

minimal time interval between the end (LOS) of the previous satellite tracking and the beginning (AOS) of the current satellite reduced tracking period.

When it is the first pass of a satellite sequential pass group, GPA acts as the following behavior rule:

1) add the new high priority goal “decrease the onboard computer telemetry acquisition rate” into the set of goal predicates that will be further used to generate the satellite tracking goals (Problem Description file). The onboard computer stores telemetry values while the satellite is out of visibility but it has a limited storage capacity. Therefore, when generating a plan for the first satellite pass of a sequential pass group the telemetry acquisition rate must be decreased so that a greater number of telemetry values can be acquired and stored into the onboard computer for being downloaded in the next pass.

When it is the last pass of a satellite sequential pass group, GPA acts as the following behavior rule:

1) add the new high priority goal “increase the onboard computer telemetry acquisition rate” into the set of goal predicates. The telemetry acquisition rate must be increased so that a lower number of telemetry values are acquired and stored into the onboard computer to avoid exceeding the limit of its storage capacity because the telemetry values cannot be downloaded in the next passes (out-of-visibility passes).

Interrelation between goals is modeled in the PDDL Planning Domain Description file by a derived predicate. A derived predicate has its truth value derived by a set of other predicates, named basic predicates. Thus, when a goal achievement depends on the achievement of another goal, the former is modeled as a derived predicate and the latter as its basic predicate. Both predicates have independent priority values defined a priori and the existence of an inter-relation between them requires a further analysis to decide whether both, none, or just one of the interrelated goals will be considered for the Problem Description file generation (goal generation) according to their priority levels. Table 4 presents the GPA behavior rules for dealing with derived predicates.

When a derived predicate is of high priority and there are basic predicates of lower priorities (normal or low priority levels), the latter predicate priority values are promoted to high priority for assuring these basic predicates will be considered for the Problem Description file generation.

When a derived predicate is of low priority and there are basic predicates of higher priority (high and/or normal priority levels), the normal priority basic predicate is not promoted: it simply keeps its priority value and will only be considered for the Problem Description file generation whether it fits into the satellite tracking period. The derived predicate and the low priority basic predicate will not be considered for the Problem Description file generation whereas the high priority basic one will be.

When a derived predicate is of normal priority and there are low, normal and/or high priority basic predicates, the normal and high priority ones keep their priority values whereas the low priority one is promoted to normal priority. High priority basic predicates are considered for the Problem Description file generation and normal priority basic predicates will only be considered if they fit into the satellite tracking period. The derived predicate will be considered if the entire set of basic predicates was previously considered.

Table 4 Goal Prioritizing Agent behavior rules for dealing with derived predicates

Derived predicate	Basic predicate	Promotion action	Goal acceptance		
			Refused	Considered under analysis	Considered
○▩	△▨ □▤ ☆▩	△▩ □▩			○▩ △▩ □▩ ☆▩
○▨	△▨ □▤ ☆▩		○▨ △▨	□▤	☆▩
○▤	△▨ □▤ ☆▩	△▤		○▤ △▤ □▤	☆▩

Legend

○ Derived predicate	△ □ ☆ Basic predicates (three distinct basic predicates)

▨ Diagonal striped pattern: low priority level	▤ Horizontal striped pattern: normal priority level	▩ Chessboard pattern: high priority level

4. Problem Generator Agent (PGA)

This agent is responsible for generating the PDDL Problem Description file for each satellite pass. It senses the satellite control environment through the following perceptions: the Pass Configuration data and the PDDL Planning Domain Description file.

Once having the above information, the PGA automatically generates the PDDL Problem Description file for each satellite tracking period. The Problem Description file contains a satellite tracking period objects, initial states, deterministic unconditional exogenous events and the goals that must be achieved at the end of the tracking period. The PGA just considers the normal and high priority goals in the goal generation process by their order of relevance. Table 5 presents the PGA schema.

The Problem Description file generation consists of inserting into the file the goal predicates to be achieved during the planning phase. For instance, PGA acts as the following behavior rules for inserting the goal predicate "execute range rate measure" into the Problem Description file:

1) retrieves from the Planning Domain Description file the set of predicates that belongs to the goal category
2) for the goal predicate "execute range rate measure", verifies the Pass Configuration Data (generated by the ICA) related to this goal predicate. Therefore, the PGA verifies the value of the Boolean configuration variable

Table 5 Problem Generator Agent schema

Agent schema	Problem Generator Agent (PGA)
Purpose	Generate the PDDL Problem Description file
Goal	Planning problem description generated
Events	Message from the ICA informing the ending of the pass configuration generation and Message from the GPA informing the ending of the goal prioritizing process. The PGA only receives this message for the satellite tracking periods that were previously reduced by the TPA or for the ones that are the first/last pass of a satellite sequential pass group
Resources	Pass Configuration Data, the Planning Domain Description file and goal priorities. The Pass Configuration Data provides the PGA data that are specific to a satellite pass. The Planning Domain Description file is defined by the Domain Generator Agent (a human agent) and describes the satellite control planning domain. It contains the set of elements necessary to model the planning domain behavior: domain types, functions, predicates, and actions. Cardoso et al. [10] proposes that predicates are classified into four categories to allow the Problem Description file automatic generation. These four predicate categories are: environment initial states, intermediate states, exogenous events and goals. The Domain Generator Agent generates the Planning Domain Description file according to this proposal, i.e. it classifies the set of predicates into these four categories and inserts labels into the Planning Domain Description file to associate predicates to their respective category.
Task workflow	Retrieve from the Planning Domain Description file a set of predicates that belongs to a specific category Considering this set of predicates, compare each element (predicate) to the Pass Configuration Data generated by the ICA. A predicate is only inserted into the Problem Description file when it satisfies the restrictions imposed by the Pass Configuration Data Insert the predicates that satisfy the Pass Configuration Data restrictions into the Problem Description file For the goal category, compare each element (goal predicate) to the Pass Configuration Data generated by the ICA and the set of goal priority values Insert the goal predicates that satisfy the Pass Configuration Data restrictions and have normal or high priority values into the Problem Description file. Normal priority goals are inserted only when there is sufficient time for the goal achievement in the satellite tracking period. Insert a derived predicate only when the entire set of its basic predicates were already inserted.

Table 6 Flight Operation Planner Agent schema

Agent Schema	Flight Operation Planner Agent (FOPA)
Purpose	Generate the Flight Operation Plan (FOP)
Goal	FOP generated
Events	End of the PDDL Problem Description file generation (by Problem Generator Agent (PGA))
Resources	PDDL Domain Description file and the PDDL Problem Description file The Domain Description file is defined by the Domain Generator Agent (a human agent) and provides the FOPA the set of domain types, functions, predicates, and actions about the satellite control planning problem. The Problem Description file is defined by the PGA and provides the FOPA the objects, initial states, exogenous events, and goals of a specific satellite pass over a specific ground station.
Task Workflow	Obtain from the Domain Generator Agent the PDDL Domain Description file name Obtain from the PGA the PDDL Problem Description file name Invoke the LPG-TD planner algorithm to generate the plan providing it both PDDL files

named RangeRateAvailable that is true when the pass presents the ground antenna minimal elevation for sending/receiving the signal and false when the pass does not present the antenna minimal elevation required

3) if the RangeRateAvailable variable value is true, inserts the predicate "execute range rate measure" into the PDDL Problem Description file as a goal predicate.

5. *Flight Operation Planner Agent (FOPA)*

This agent is responsible for generating the FOP which contains satellite control operations (planning actions) to be executed by an Executor Agent (EA) during the satellite tracking period. Flight Operation Planner Agent (FOPA) generates a FOP for each satellite to be tracked by a specific ground station antenna.

FOPA has as input the Planning Domain and Problem Description files, written in PDDL 2.2. For plan generation, it uses the temporal planner LPG-TD as its reasoning mechanism. Table 6 presents the FOPA schema.

6. *Executor Agent (EA)*

This agent is responsible for the FOP automated execution at the specified instants of time.

Obeying the sequence of operations (planning actions) specified in the FOP, the Executor Agent (EA) calls the Mission Execution System functions related to each plan operation. In case of anomalies, the EA notifies the remote human satellite operator and allows his intervention. Table 7 presents the EA schema.

MASOA architecture performs a set of tasks to generate FOP. The first task is executed by the TPA, which obtains all the satellites' visibility periods initial and ending times for a specific ground station from the PVP Database. After receiving this information, this agent compares the satellites' initial and ending times, reducing the least priority satellite tracking periods when some intersection occurs and cancelling tracking periods shorter than a pre-defined minimal time duration. The satellite tracking periods that have their time window reduced are annotated with the flag *reduced* to sign it.

After performing its task, TPA provides the ICA the satellite passes (identified by their orbit numbers) that are scheduled to be tracked by ground stations. For each satellite tracking, ICA generates essential pass configuration data.

In parallel, TPA informs the GPA the occurrence of exceptional situations: a satellite tracking period reduction performed to avoid time conflict with another satellite (satellite tracking periods annotated with the flag *reduced*) or the first/last pass of a satellite sequential pass group. To manage these exceptional situations, GPA prioritizes goals by adding new high priority goals to the set of goal predicates that will make them be further considered for the Problem Description file generation (goal generation) and/or by decreasing goal priorities to the low level that will make them be eliminated from the Problem Description file generation process.

After ICA and GPA provide their services, the PGA considers the Pass Configuration Data provided by ICA, the goal priorities provided by GPA, and the Planning Domain Description file provided by the Domain Generator Agent (human agent) as inputs and generates a PDDL Problem Description file for each specific satellite tracking. The PGA

Table 7 Executor Agent schema

Agent schema	Executor Agent (EA)
Purpose	Execute the Flight Operation Plan (FOP) operations
Goal	Flight operations executed
Events	End of the FOP generation (by Flight Operation Planner Agent (FOPA)) System time that starts the agent behavior
Resources	The FOP files The FOP is defined by the FOPA and provides the EA the sequence of operations (planning actions) to be executed.
Task workflow	Obtain the FOP from the Flight Operation Planner Agent For each plan operation, request from the Mission Execution System the service related to the operation at the instant of time specified in the plan. The Mission Execution System provides services such as telecommand sending and telemetry receiving In case of anomalies, notify the Satellite Engineer (SATENG) Human Agent who can pause the automatic operation execution, interfere in the execution and play the plan automatic execution again from the point (operation) of his choice during the satellite pass Plan operations that could not be executed during a satellite pass have their priorities augmented to the high priority level for being executed in the next tracking.

inserts into the Problem Description file the goal predicates that satisfy the Pass Configuration Data restrictions and have normal or high priority values.

Once the PDDL Problem Description file fitting the most relevant goals into the satellite tracking period restriction of time is generated, FOPA finally generates a FOP for each satellite. The generated FOPs are ready for automatic execution by the EA.

V. Prototype Tool and Results

A prototype tool is being developed to realize MASOA architecture. Figure 8 illustrates the graphical user interface tool which provides three services: 1) update the configuration database parameters (Configuration button); 2) generate FOP (Generate FOP button); and 3)visualize the generated plans (Visualize FOP button).

When the users wish to generate a FOP, they must previously click on Select PVP button to choose a PVP file. A PVP file contains information about the tracking of a satellite seven-day group of passes. Then, the tool extracts the satellite set of passes from this file and the user can select the ones he wishes the tool to generate a FOP for. The selection of a group of passes for the FOP generation allows the same configuration database operation scenario to be applied to this entire group of passes.

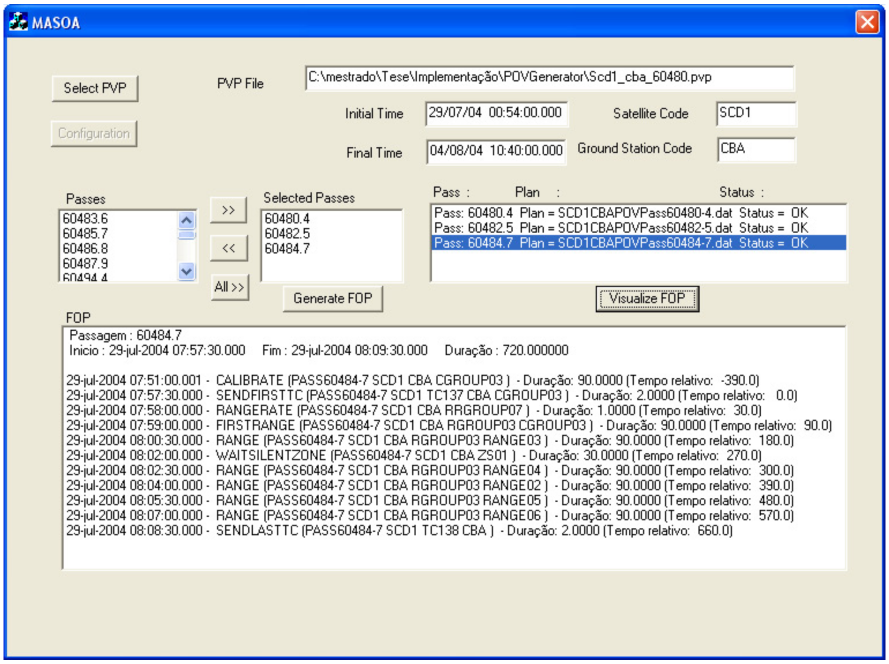
Once the PVP file and the satellite passes have been chosen, the Generate Plan interface button activates the PGA service which consists of automatically generating the PDDL Problem Description file. This agent communicates with FOPA, which then generates the plans according to the LPG-TD planner algorithm. The inputs for FOPA plan generation are the recently generated PDDL Problem Description files and the PDDL Domain Description file.

The user may also visualize a generated plan by choosing it and clicking on the Visualize Plans button. The plan content is presented on the screen and the user may choose the options to print it and manually modify some of its actions, if necessary. A plan line is composed of three items: 1) the pass initial time in which the action must be executed; 2) the action to be executed and its parameters; and 3) the action execution duration.

The tool main screen shows a generated FOP for pass 60484.7 of satellite SCD1 to be tracked by Cuiabá ground station.

The tool was tested for the in-orbit Brazilian satellites SCD1, SCD2, and CBERS2. PVP files used as input for the tool in the testing process were provided by the Satellite Control Center at INPE.

The testing consisted of automatically comparing the prototype tool generated plans with the real plans generated by the Satellite Control Center. This comparison was completely automated owing to the development of a second tool specially designed to realize this burdensome task. Table 8 presents the comparison results.



PVP File: C:\mestrado\Tese\Implementação\POVGenerator\Scd1_cba_60480.pvp

Initial Time: 29/07/04 00:54:00.000

Satellite Code: SCD1

Final Time: 04/08/04 10:40:00.000

Ground Station Code: CBA

Passes:

60483.6
60485.7
60486.8
60487.9
60494.4

Selected Passes:

60480.4
60482.5
60484.7

```
Pass :          Plan    :                                    Status :
Pass: 60480.4  Plan = SCD1CBAPOVPass60480-4.dat  Status =  OK
Pass: 60482.5  Plan = SCD1CBAPOVPass60482-5.dat  Status =  OK
Pass: 60484.7  Plan = SCD1CBAPOVPass60484-7.dat  Status =  OK
```

FOP

```
Passagem : 60484.7
Inicio : 29-jul-2004 07:57:30.000    Fim : 29-jul-2004 08:09:30.000    Duração : 720.000000

29-jul-2004 07:51:00.001 -  CALIBRATE (PASS60484-7 SCD1 CBA CGROUP03 )  - Duração: 90.0000 (Tempo relativo:  -390.0)
29-jul-2004 07:57:30.000 -  SENDFIRSTTC (PASS60484-7 SCD1 TC137 CBA CGROUP03 )  - Duração: 2.0000 (Tempo relativo:   0.0)
29-jul-2004 07:58:00.000 -  RANGERATE (PASS60484-7 SCD1 CBA RRGROUP07 )  - Duração: 1.0000 (Tempo relativo:  30.0)
29-jul-2004 07:59:00.000 -  FIRSTRANGE (PASS60484-7 SCD1 CBA RGROUP03 CGROUP03 )  - Duração: 90.0000 (Tempo relativo:  90.0)
29-jul-2004 08:00:30.000 -  RANGE (PASS60484-7 SCD1 CBA RGROUP03 RANGE03 )  - Duração: 90.0000 (Tempo relativo:  180.0)
29-jul-2004 08:02:00.000 -  WAITSILENTZONE (PASS60484-7 SCD1 CBA ZS01 )  - Duração: 30.0000 (Tempo relativo:  270.0)
29-jul-2004 08:02:30.000 -  RANGE (PASS60484-7 SCD1 CBA RGROUP03 RANGE04 )  - Duração: 90.0000 (Tempo relativo:  300.0)
29-jul-2004 08:04:00.000 -  RANGE (PASS60484-7 SCD1 CBA RGROUP03 RANGE02 )  - Duração: 90.0000 (Tempo relativo:  390.0)
29-jul-2004 08:05:30.000 -  RANGE (PASS60484-7 SCD1 CBA RGROUP03 RANGE05 )  - Duração: 90.0000 (Tempo relativo:  480.0)
29-jul-2004 08:07:00.000 -  RANGE (PASS60484-7 SCD1 CBA RGROUP03 RANGE06 )  - Duração: 90.0000 (Tempo relativo:  570.0)
29-jul-2004 08:08:30.000 -  SENDLASTTC (PASS60484-7 SCD1 TC138 CBA )  - Duração: 2.0000 (Tempo relativo:  660.0)
```

Fig. 8 Prototype tool screen with a generated FOP.

Table 8 Comparison results among the tool generated plans and the real plans generated for a seven-day multi-satellite tracking

	PVP file			Result		
Satellite code	Pass code	Period	Passes amount	Equal plans	Different plans	No solution
SCD1	60480	07/29–08/04/2004	57	36	14	07
				63%	25%	12%
SCD2	30444	07/29–08/04/2004	57	46	11	00
				81%	19%	0%
CBERS2	9668	08/24–08/31/2005	28	22	06	00
				79%	21%	0%

The difference among the plans occurred in the order and amount of distance measure actions. Considering that a distance measure action is not a crucial type of action, satellite engineers assure the plans to be still valid.

On the other hand, there were seven plans that could not be generated for satellite SCD1 owing to the fact that the set of goals inserted in the PDDL Problem Description files (generated by the prototype tool) could not fit into the satellite tracking periods. These goals could not fit into the satellite tracking periods because SCD1 satellite currently requires extra operations to switch the satellite on-board data collecting transponder on and off. The transponder must be switched off for energy saving. In fact, SCD1 battery is degrading thus requiring the transponder to be switched off while outside visibility.

VI. Related Works

Growing attention has been paid to multi-agent planning systems in AI. Xu et al.'s [11] proposal is close to ours owing to the adoption of distributed planning agents. However, the system they propose consists of a multi-agent planning system for deep space exploration.

The authors propose a formal planning model for the planning domain description. Although the model is a valid one, they justify the need for it by pointing out the STRIPS operator deficiencies, which is solely a subset of PDDL language. PDDL was not referenced.

We consider that using PDDL or extending it when it is not sufficient to model a planning domain is a good practice as the AI Planning groups are making a general effort to standardize PDDL and make it a practical planning language. Efforts may be found in the IPC (International Planning Competition) web site** (last accessed in February 2008).

Coddington [12] considers PDDL and proposes a planning framework that also adopts the temporal planning paradigm. This framework also reasons about goals with priorities in a distinct way from ours.

Coddington's framework edits the plan when there is insufficient time available to achieve all the goals. It removes from the plan a goal and all of its associated actions and constraints. By doing this, however, there is the associated cost of maintaining the dependencies between actions and goals during the planning process.

In MASOA architecture, when there is insufficient time to achieve the goals (reduced tracking period), the plan is not generated thus avoiding plan editing after its generation. Instead of generating the plan and editing it afterwards, the GPA edits the PDDL Problem Description file (input for the planner) until there is sufficient time to achieve a subset of the original goals.

By solely editing the input for the planner, we avoid having to interfere with the planner activity to obtain the dependencies between actions and goals during the planning process. Therefore, in MASOA architecture the planner is considered as a black-box subcomponent with its input being adjusted to portray the varying context of the satellite control domain. The advantage of treating the planner as a black-box subcomponent is the possibility of replacing it as the AI planning area evolves.

Cesta et al. [13,14] proposes the Mars-Express Scheduling Architecture (Mexar2), an AI-based tool in daily use on the Mars-Express mission since February 2005, which focuses deep-space missions that generally carry a larger set of different and complementary onboard payloads. Mexar2's main role is to synthesize data generated by the payloads for optimized downlinking thus reducing the ratio of mission costs to science return.

MASOA architecture focuses are on low-orbit missions and on defining real-time and time-tagged telecommands to be transferred to satellites. Despite that, MASOA and Mexar2 architectures both solve, at their core, temporal planning problems. But they perform this task in different ways. Mexar2 is based on a CSP (Constraint Satisfaction Problem) approach for the modeling phase and provides a hybrid solving procedure that combines backtracking with a max flow algorithm. On the other hand, MASOA adopts the LPG-TD planner algorithm as its reasoning mechanism. A key aspect is that both approaches build an AI-based representation of the domain that is input for the planner algorithms and contains the relevant objects to describe the planning problem features.

VII. Conclusions

In this paper a new multi-agent automation planning and execution architecture is proposed to the context of multi-satellite control domain. MASOA architecture automatically plans the space operations to be uplinked and downlinked regarding the restricted period of time that low Earth orbiting satellites are visible to ground stations.

MASOA architecture solves the problem of controlling satellites with conflicting time tracking periods. When a set of satellites share the same ground station, there may be intersection in the visibility period of one satellite with another one. When this situation takes place, the TPA reduces the tracking period of the lowest priority satellite. This time period reduction means the MASOA architecture planner agent (FOPA) will not have sufficient time to achieve the entire set of original goals (tracking goals).

At this point, the GPA must attribute new priorities to goals so that FOPA can generate a plan that achieves the set of the most important goals within the reduced satellite tracking period. Goal priorities are originally defined by the OPDIR Human Agent and may be reconfigured to determine the next satellite tracking requirements. Priority goals that were lowered during the generation of a satellite control plan may have their priorities elevated in the next plan generation. This enables configuration of the planning task to reflect the actual satellite tracking requirements.

To manage this situation, the GPA is currently being modeled. By the time the GPA coding is finished, the MASOA architecture prototype tool will be able to generate currently non-generated plans. The tool implements the PGA and FOPA functionalities so far. Following coding, the aim is to have the entire MASOA architecture tool functioning appropriately.

**http://teamcore.usc.edu/koenig/icaps/competitions.htm

Concerning the aspect of operation execution automation, MASOA architecture intends to facilitate the work of human satellite operators by performing the most repetitive tasks. This feature is still under development.

The results obtained are concerned with the aim of increasing the configurability of the space operation planning task. The definition of a configuration database for the satellite control domain and the automatic generation of the PDDL Problem Description file allows the change of INPE's satellite operation scenario with no need to modify the PDDL Domain Description file and the planner code.

For instance, in the case of controlling an additional new satellite or considering a new ground station during the planning process the only thing to do is to insert the new satellite or the new ground station data into the configuration database. The changes made on the configuration database are automatically reflected by the PGA into the planning problem instance (PDDL Problem Description file). This concept of automatically editing the planning process input is a practical way of providing configurability to the planning task, facilitating the functions of the satellite operation staff.

References

[1] European Cooperation for Space Standardization (ESA-ESTEC), "Space Engineering. Ground Systems and Operations – Part 1: Principles and Requirements", ECSS-E-70 Part 1A, ESA Publications Division, Netherlands, 2000.

[2] Fikes, R., and Nilsson, N., "STRIPS: A New Approach to the Application of Theorem Proving to Problem-Solving", *Artificial Intelligence*, Vol. 3–4, No. 2, 1971, pp. 189–208.

[3] Fox, M., and Long, D., "PDDL 2.1: An Extension to PDDL for Expressing Temporal Planning Domains", *Journal of Artificial Intelligence Research*, Vol. 20, 2003, pp. 61–124, available at: http://www.jair.org/vol/vol20.html.

[4] Edelkamp, S., and Hoffmann, J., "PDDL 2.2: The Language for the Classical Part of the 4th International Planning Competition Technical Report", Technical Report 195, Albert Ludwigs Universität, Institüt fur Informatik, Freiburg, Germany, 2004.

[5] Goldman, R. P., "Adapting Research Planners for Applications", *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling*, Association for the Advancement of Artificial Intelligence (AAAI), Whistler, British Columbia, Canada, June, 2004, pp. 279–286.

[6] Caire, G., Coulier, W., Garijo, F., Gomez, J., Pavon, J., Leal, F., Chainho, P., Kearney, P., Stark, J., Evans, R., and Massonet, P., "Agent Oriented Analysis using MESSAGE/UML", *Lecture Notes in Computer Science*, Vol. 2222, January 2002, pp. 119–135.

[7] Zambonelli, F., Jennings, N. R., and Wooldridge, M., "Developing Multiagent Systems: The Gaia Methodology", *ACM Transactions on Software Engineering and Methodology*, Vol. 12, No. 3, July 2003, pp. 317–370. doi: 10.1145/958961.958963

[8] Iglesias, C. A., and Garijo, M., "Agented-Oriented Methodologies," *The Agent Oriented Methodology MASCommonKADS*, edited by B. Henderson-Sellers and P. Giorgini, IDEA Group Inc., Hershey, PA, USA, 2005, pp. 46–78.

[9] O'Malley, S. A., and DeLoach, S. A., "Determining when to Use an Agent-Oriented Software Engineering Paradigm", *Lecture Notes in Computer Science*, Vol. 2222, January 2002, pp. 188–205.

[10] Cardoso, L. S., Ferreira, M. G. V., and Orlando, V., "Aplicação da Tecnologia de Agentes de Planejamento em Operações de Satélite", M.Sc. Dissertation, Applied Computing Postgraduate Program, Brazilian National Institute for Space Research, São José dos Campos, Brazil, 2006.

[11] Xu, R., Cui, P., Xu, X., and Cui, H., "Multi-Agent Planning System for Spacecraft", *Proceedings of the Second International Conference on Machine Learning and Cybernetics*, IEEE Computer Society, Xi'an, China, November, 2003, pp. 1995–1999.

[12] Coddington, A., "A Continuous Planning Framework with Durative Actions", *Proceedings of the Ninth International Symposium on Temporal Representation and Reasoning*, IEEE Computer Society, Manchester, UK, July, 2002, pp. 108–114.

[13] Cesta, A., Cortellessa, G., Fratini, S., Oddi, A., Denis, M., Donati, A., Policella, N., Rabenau, E., and Schulster, J., "Mexar2: AI Solves Mission Planner Problems", *IEEE Intelligent Systems*, Vol. 22, Issue 4, July–August 2007, pp. 12–19. doi: 10.1109/MIS.2007.75

[14] Cesta, A., Cortellessa, G., Fratini, S., Oddi, A., and Policella, N., "Mexar2: An Operational Tool for Continuous Support to Mission Planning", *Proceedings of the Fifth International Workshop on Planning and Scheduling for Space*, Space Telescope Science Institute, Baltimore, MD, USA, October, 2006, available at: http://www.stsci.edu/institute/conference/iwpss/ProceedingsArchive.

Roy Sterritt
Associate Editor