Describing Planning Domains Through an Object-Oriented Model

Abstract

The National Institute of Space Researches (INPE) has been working and having great success in it's space operations. However, INPE has been finding several challenges when performing these missions. Among them, it is the high operational cost, because these missions are executed manually. This problem has been theme of several works and congresses where the automation solution has been proposed.

A technique that has been used, to create the planning of the spacecraft or satellites activities and finds the objectives of space missions, is the Planning and Scheduling of Artificial Intelligence (AIPS).

This article describes a work in progress, construct a meta-model oriented object enough generic to describe any domain related to space missions. This work also proposes an alternative language of representation to planning problems. Therefore, it will allow more flexibility when defining domains by definitions of components in the model.

To establish a strategy is aimed here to represent planning problems. The construction of a framework that starts from a definition created through the model and supplies this representation in any language of existent description, like PDDL and OCL.

This model contemplates the necessary requirements for planning applications. For this approach to be valid as a representation of the knowledge, the model proposed demand to be sufficiently wide representing a wide variety of problems, but restrictive enough to allow efficient algorithms to operate on him. It is also intended, to allow the modeling of problems that involve time and restrictions of resources in the generation of plans; to model planning problems involving events exogenous (as the access or output of eclipse zones, in the orbit) and to map the behavior of the actions through the predicates.

Planning and Scheduling

According to [1] to plan is the abstract and deliberative process of choice and organization of actions advancing expected effects. That process has as mission to reach, in the best possible way, the pre-established objectives. Scheduling is define by **Erro! Fonte de referência não encontrada.** et al. (1993) as "the process of allocating time and resources to tasks in a plan, still satisfying a series of domain restrictions".

The basic idea of the planning process is to simulate the behavior of an environment in which a sequence of actions (also called in the existent literature as tasks, activities or operators) is applied. The characteristics of the environment are represented in a logical way by a group of states, and each applied action to the environment can modify one or more of the states. The goal is to find the correct sequence of actions that ease the environment of the initial state to a wanted state, or state-objective.

For that, the planner should possess a model of the environment with representations of the states, a group of applicable actions to the model - each possessing pre-conditions for the execution and the description of the effects - and the initial and objective states [2].

A plan is generated from a planning process that is based on an initial state, and starts to select actions, among a group specified of actions in the domain taking precondition consideration, trying to reach the established objective. After the action's execution the effects are applied on the model. If the obtained state does not reach the expected goal, the process proceeds, until that the same is reached. The group of actions executed with success becomes the plan.

With the great use of the techniques of Planning, the Engineering of Knowledge started to have a great importance in the conception of the problems of the space area. The specification, modeling, and analysis of the planning domain become essential for the best understanding and classification of the domains and planning problems.

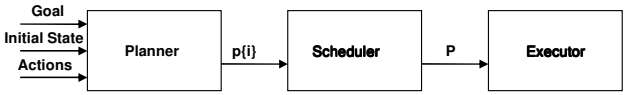

Figure 1. Agent of Classic Planning
SOURCE: Feber (1999), p. 152

Knowledge Enginnering for AIPS

The Engineering of Knowledge studies the beginnings and methods for the development and construction of Systems Based in Knowledge (Knowledge Based Systems – KBS[3].

KBS's are softwares that are based in data stored in a knowledge base, solve problems usually being used in artificial intelligence inference machines, tends as auxiliary objective and/or to substitute human specialists in your tasks. The Figure 2 presents a simplified architecture of a KBS.

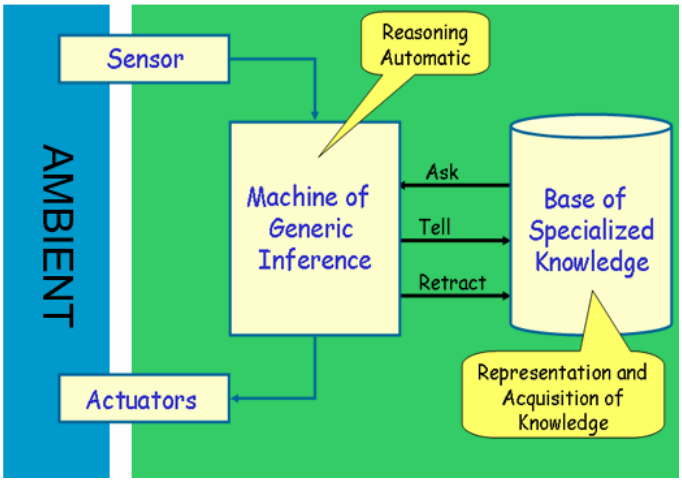

Figure 2. Simplified architecture of a system based on knowledge.

In this architecture we have the Base knowledge that contains, for instance, the representations of actions and events of the domain, as well as the rule; and the Machine of Generic inference (Responsible for evaluating, chaining and determining when this knowledge should be used), responsible for the automatic reasoning on the Knowledge Base and the facts for the resolution of problems.
KBS are used by a vast range of applications, among them are the applications based in Planning.

From your origin, the Automatic Planning is, in fact, based on knowledge. That is, the planning process involves the manipulation of knowledge of complex phenomena, such as the actions [1].

The first step when representing knowledge about a world (I) is to conceptualize it in terms of its object, functions, and relations [4].

Some works, like Lee McCluskey (2002), explains some defined terminologies in Automatic Planning. The knowledge associated to a certain real application of planning is denominated ***domain***. Any form of symbolic representation of part of the domain can be called description of the domain. Specification of the domain is an abstract description in a definitive and closed domain. It is waited that a specification of the domain is complete and you need. I model of the domain is the specification of the domain in the operational form, containing explicit details of the properties and of the dynamics of the domain, appropriate to be processed by a planning mechanism. Present phenomena in the model of the domain owe corresponded to those presents in the real domain [5].

The model of the domain and your representations of problems, they are necessary for the planners to produce the plan. Being like this the Engineering of Knowledge comes to be of addition importance for the area of Planning.

Languages of Modelling of Domains

The representation languages of planning problems are explicit representations of states and actions that turn possible the derivation of effective heuristics and the development of powerful and flexible algorithms for the resolution of problems [6]. Still according to Russel (2004), the representation of planning problems - states, actions and objectives - should make it possible to create planning algorithms to take advantage of the logical structure of the problem. The solution is to find a language to be sufficiently expressive to describe a wide variety of problems, but restrictive enough to allow algorithms to operate on it.
The first planning system was announced in 1971, called STRIPS - *Stanford Research Institute Problem Solver,* however the representation of actions and states based in literals, that was known also as predicates, and the model of actions where the per-conditions and effects of a action represent the dynamics of the execution of this in the domain, had very more influence than your algorithmic approach.
But several restrictions imposed by the representation of STRIPS that were chosen, trying to turn the algorithms simpler and more efficient, without turning very difficult to describe the real (Russel, 2004) problems, they ended up motivating the creation of several variants, and even new languages as ADL - Action Description Language [8] and PDDL, that followed the same paradigm, however less restricted.
The Action Description Language [8], created in 1989 trying to get a less rigid representation than STRIPS, allowing to describe more complex models, permitting to describe more complex models. Like this, ADL added the separation in per-conditions, conditional in effects and universal quantification, in the per-conditions and in the effects.
Up to 1998, a standard format to represent the definition of problems among the planning community, didn't exist. With that, the planners WENT were developed using different access types with variable formats. So, the responsible committee for organization of the first International Competition of Planning (IPC) made available the language PDDL. Planning Domain Description Language for description of planning domains was created to be used in the competition of that same year IPC/98 [9][10]. It passed to be a possible way to compare planners' efficiency WENT to solve real problems.
Some main features of the PDDL are:

- The PDDL is a representation directed to the action of the domain;
- The actions represented in PDDL are based on action of model STRIPS [11] where the daily pay-conditions and effect of an action represent the dynamics of execution in the domain;
- It possesses features of language ADL that include the representation of conditional

effect in the actions, as well as universal qualifiers and quantifiers.

- Definition of restrictions;
- Specification of composite hierarchic actions for sub action [9];
- Had to the fact of the language to be default it is possible that one exactly problem represented in PDDL is treated by various different planners (ability of problems between planning agents).

The knowledge of the PDDL is compound by two distinct archives: the one that describes the domain and specific; and the other that describes the problem. Description of the domain is valid only for diverse problems that are composites for descriptions of the initial state and goals to be reached.

Its definition can be found completely in the specification of the version 2.1 [12] or in posterior versions as the PDDL 2.2 [13] and PDDL 3.0 [14].

Implemented and in PROLOG and with the idea of representing models of dominions' as a set of objects citizens to some restrictions, the proposal of McCluskey (2000), being a representation based on objects, if they still show restricted for describing the model as a set of, predefined.

The OCL has as great contribution to allow a better structure and greater representation of the knowledge on the shaped world being. But, for still being based on description for predicates, the language is dependant of the explicit representation of any change in the states of the model, accurately as in STRIPS or PDDL. Of any form, the OCL represents a great pacing in route to the description of dominions' of form total guided objects [2].

Generic Object-Oriented Model

The domain's modeling of planning is normally costs for the lack of method and directed principles of modeling to these types of domains. The modeling processes normally are guided by these methods. Methods and procedures of modeling for planning are still appearing, as this work.

To correctly get and to represent the knowledge in planning applications are not a simple task. The aspects of the knowledge in these applications are recognized as being the biggest difficulty in the design. In aerospace and military experiences [15] [16] [17], they had pointed that the aspects of the Engineering of Knowledge are the ones that require more attention. These experiences and the restrictions imposed for the created forms of representation until today, motivated the development of this work that consists of creating Generic Object-Oriented Meta-Model for Planning Domain Representation (GOOMETA-PDR).

In this article the modeling of the static structure of the GOOMETA-PDR will be presented, through a diagram of classrooms presented in Figure 3.

This boarding takes in consideration the practical good methods and of modeling proceeding from the software development, however it expects with the GOOMETA-PDR to describe any domain and problem of the planning area, being able with this to take off advantage of the benefits of the Prompt the Objects. From this model we can have planners who use design patterns for the implementation of the most varied planning algorithms, thus earning the great flexibility in the adoption of planners.

Aiming at to allow to greater flexibility in the definitions of the domain, the model is being constructed taking in consideration the real necessities of a real problem of planning.

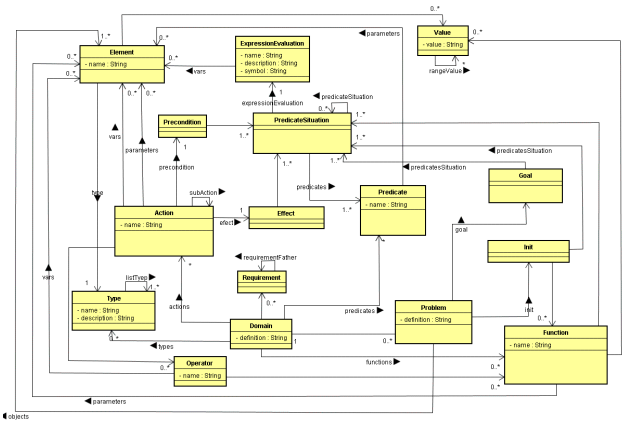

Figure 3. Generic Object Oriented Meta-Model Planning Domain Representation

The model is composed by 16 class and your respective relationships, you follows the description of each class:

- The class ***Type*** is the abstraction of the types that can be created in a domain, former: Integer, Satellite, Region, etc...
- ***Value*** represents a value that will be configured in the domain so much for a function as for an element, the classes ***Value*** has a self-relationship that makes possible the configuration of one range of values.
- Element abstracted the objects that will be part of the problem, I also abstracted the parameters of the predicates and functions and necessary variables in the description. ***Element*** has a type and you can have values (***Value***).
- ***ExpressionEvaluation*** as the own name already says represent evaluation expressions, these expressions will be applied to predicates and they will also be able to have applied variables the evaluation, situation current one in descriptions of the satellite domain, This class has three attributes one for the description of the name of the expression, one for the description and one for a symbol for this description. Example of

expressions that can be created: > *(larger), Not (Denial), And, Or, etc...*

- ***Predicate*** abstracted the representation of the predicates of a domain, that class have a list of parameters that will be composed by defined elements in the domain.
- ***Predicate*****Situation** class that I abstracted the situation of a predicate, it is composed by one or more predicates, for an instance of ***ExpressionEvaluation***, and you contain a self-relationship that makes possible the concatenation of situations for a predicate.
- ***Action*** represents the actions of the domain, this contemplates the specification of "Sub Actions" through a self-relationship, they are also composed by two lists of elements one that represents the parameters and other the variables of each specified action. Each action specifies, you possess an effect and a precondition that they are represented respectively by the class Effect and Precondition, and these are composed by a list of Predicate Situation.
- ***Operator*** this class has purpose to represent necessary extensions for specific actions of each domain, for instance: time and resource. Making possible like this the representation of the most several domains of the planning area. Besides her to possess a list of elements, she has a list of ***Function's*** being possible the representation of dynamic data for each created abstraction.
- ***Function*** abstracted possible functions to be part of each domain, Function has a list of values that make possible the configuration of initial values or a range of values, also possesses a list of elements that represent the parameters of these functions. In this, it will be able to inserted rules being used of a group of Predicate Situation.
- ***Init*** representation of the initial state of a problem, a list of Function's contemplates functions to be executed in I begin him/it of the planning, the represented state is the composition of evaluation expressions on a predicate.
- ***Goal*** responsible abstraction for representing the goal to be reached by a planner, such goal is represented as the initial state, described in the class Init.
- ***Problem*** has for purpose to represent the description of the problems for a fact control, this is composed by an instance of the class Init, an instance of the class Goal and also a list of elements that will represent the objects to be used by the planner.
- ***Domain*** represents the description of the planning control, composed by a list of Requirement's and each requirement has a solemnity relationship that makes possible the reuse of defined rules, ***Domain*** is composed also by a list of actions, you possesses a list of predicates, also several Type's can be declared for a control and a group of Function's. A control can have several problem instances.

To demonstrate the representation of the data of a real domain a matching will be made enters a description of domain in PDDL and an object diagram. Figure 4 presents an example of domain in PDDL, representing states and relative actions to the operation of a satellite. Figure 5 demonstrate part of the object diagram representing the same description of domain presented for Figure 4, using them of the GOOMETA-PDR.

```
(define (domain satellite)
 (:requirements :strips :equality :typing :fluents :durative-actions)
 (:types satellite direction instrument mode)
 (:predicates
             (on_board ?i - instrument ?s - satellite)
             (supports ?i - instrument ?m - mode)
             (pointing ?s - satellite ?d - direction)
             (power_avail ?s - satellite)
             (power_on ?i - instrument)
             (have_image ?d - direction ?m - mode) )
 (:functions (slew_time ?a ?b - direction))

 (:durative-action turn_to
   :parameters (?s - satellite ?d_new - direction ?d_prev - direction)
   :duration (= ?duration (slew_time ?d_prev ?d_new))
   :condition (and (at start (pointing ?s ?d_prev)) )
   :effect (and  (at end (pointing ?s ?d_new))
                 (at start (not (pointing ?s ?d_prev)))))

 (:durative-action switch_on
   :parameters (?i - instrument ?s - satellite)
   :duration (= ?duration 2)
   :condition (and (over all (on_board ?i ?s))
                   (at start (power_avail ?s)))
   :effect (and (at end (power_on ?i))
                (at start (not (power_avail ?s)))))

  (:durative-action switch_off
   :parameters (?i - instrument ?s - satellite)
   :duration (= ?duration 1)
   :condition (and (over all (on_board ?i ?s))
                   (at start (power_on ?i)))
   :effect (and (at start (not (power_on ?i)))
                (at end (power_avail ?s))) )

  (:durative-action take_image
   :parameters(?s- satellite ?d - direction ?i - instrument ?m - mode)
   :duration (= ?duration 7)
   :condition (and    (over all (on_board ?i ?s))
                      (over all (supports ?i ?m) )
                      (over all (power_on ?i))
                      (over all (pointing ?s ?d))
                      (at end (power_on ?i)))
   :effect (and (at end (have_image ?d ?m)))))
```

Figure 4. Description Domain of a Satellite in PDDL SOURCE: adapted of Cardoso (2006), p. 65

The diagram presented in Figure 5 demonstrates the possibilities that the GOOMETA-PDR allows how much the freedom of object definition, types, predicates, action and until new concepts using itself of the abstraction made for the ***Operator*** classroom. Another interesting possibility that the GOOMETA-PDR offers is the construction a vast gamma of interfaces in such a way for human user how much for the integration of systems and also the possibility of the diverse construction of framework' s in such a way

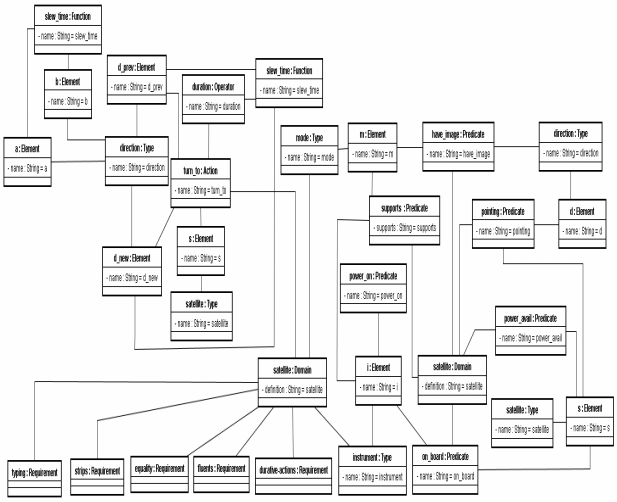

Figure 5. Part of the object's diagram representing the Description Domain of a Satellite

aiming at the planning as the description of dominions' and problems as conversion for the diverse existing languages, thus making possible the use of implemented planners already. Figures 6 and 7 present an interface archetype human being-computer for description of problem and part of the description presented in the Figure the 4 and 6 in XML using GOOMETA-PDR, respectively.

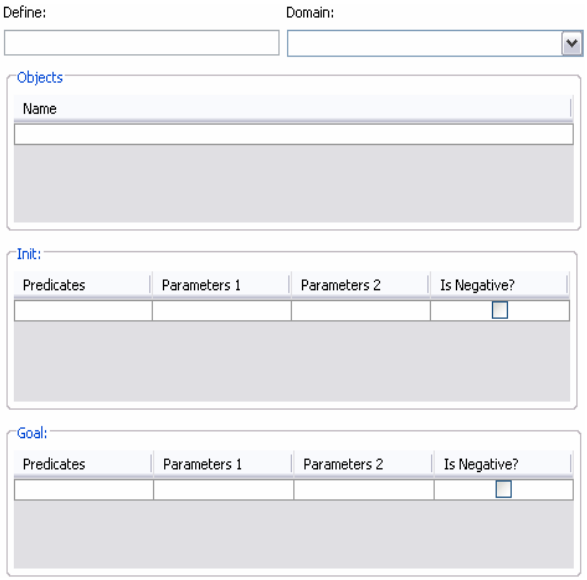

Figura 6. Prototype of interface human-computer for Description of a Problem.

```
<?xml version="1.0" encoding="UTF-8"?>
<domain definition="satellite">
    <requirements>
        <requirement name="typing"/>
        <requirement name="strips"/>
        ...
    </requirements>
    <types>
        <type name="satellite" description="..."/>
        <type name="direction" description="..."/>
        <type name="instrument" description="..."/>
        <type name="mode" description="..."/>
    </types>
    <prdicates>
        <predicate nameF="on_board"/>
            <parameters>
                <Element name="i">
                    <type name="instrument"/>
                </Element>
                <Element name="m" type="mode"/>
            </parameters>
        </predicate>
        ...
    <prdicates>
    <functions>
        <function name="slew_time">
            <parameters>
                <Element name="a"type="mode"/>
                <Element name="b" type="mode"/>
            </parameters>
        <function>
    </functions>
</domain>
```

Figura 7. Description Domain of a Satellite in XML using GOOMPDR.

Conclusion

This article describe a working in progress that proposes oriented meta-model to generic objects for the description of any domain related to space missions.

The approach here described with other two works [6] and [2], is part of an initiative of INPE of automating the operations of your satellites.

The model proposed in this labor brings the planning concepts and scaling of the area to the systems of operation of satellites, together with concepts of Orientation to Objects.

Beyond assisting the generation of plans for space operations, you will also contribute in the representation of the knowledge of the planning domain, facilitating like this the correct mapping of the necessary input data for the generation of the definition domain.

The proposed model will also assist in the representation of the knowledge of the planning domain, thus facilitating the correct mapping of the necessary data of entrance for the generation of the definition of the domain.

This approach opens other possibilities in the development of applications of planners also facilitating reuses it and a bigger adhesion to the AIPS as technique in space missions, as much in the INPE how much in all the space community.

References

[1] T. S. Vaquero. itSIMPLE: Ambiente integrado de Modelagem e Análise de Domínios de Planejamento Automático. Dissertação (Mestrado) – Escola Poltécnica da Universidade de São Paulo, São Paulo, 2007.

[2] F. N. Kucinskis. Alocação de Recursos para Experimentos Científicos com Replanejamento Automatizado a Bordo de

Satélites. Instituto Nacional de Pesquisas Espaciais(INPE), São José dos Campos, Março, 2007.
[3] R. Studer, V. Benjamins, and D. Fensel. Knowledge engineering: Principles and methods. IEEE Transactions on Data and Knowledge Engineering. 1998.
[4] N. J. Nilsson. Artificial Intelligence: a new synthesis. Morgan Kaufmann Publishers, Inc. 1998.
[5] T. L. McCluskey. Knowledge Enginnering: Issues for the AI Planning Community. Proceedings of the AIPS-2002 Workshop on Knowledge Tools and Techniques for AI Planning, Toulouse, France, April 2002.
[6] L. S. Cardoso. Aplicação da Tecnologia de Agentes de Planejamento em Operações de Satélites. Dissertação de Mestrado em Ciência da Computação. INPE, 2006.
[7] S. Russell and P. Norvig. Inteligência Artificial, Tradução da 2a. Edição, Editora Campus, 2004.
[8] E. Pednault. ADL:Exploring the Middle Ground between STRIPS and the Situation Calculus, in Proceedings of KR-89, pp. 324-332.
[9] D. McDermott. The PDDL Planning Domain Definition Language. The AIPS-98 Planning Competition Committee. 1998.
[10] D. McDermott. PDDL2.1 – The Art of the Possible?. Commentary on Fox and Long. Journal of Artificial Intelligence Research. 2003.
[11] R. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 1971.
[12] M. Fox and D. Long. PDDL 2.1: An Extension to PDDL for Expressing Temporal Planning Domains. Journal of Artificial Intelligence Research, 2003.
[13] S. Edelkamp and J. Hoffman. PDDL 2.2: The Language for Classical Part of the 4th International Planning Competition. Techinical Report, Fachbereich Informatik and Institut für Informatik. Germany. 2004.
[14] A. Gerevini and D. Long. Plan Constraints and Preferences in PDDL3 – The Language of Fifth International Planning Competition. Technical Report, Department of Eletronics for Automation, University of Brescia, Italy, August 2005.
[15] D. Wilkins. Using the SIPE-2 Planning System: A Manual for SIPE-2, Version 5.0 SRI International, Artificial Intelligence Center. 1999.
[16] A. Tate, B. Drabble and J. Dalton. O-Plan: a Knowledge-Based Planner and its Application to Logistics. AIAI, University of Edinburgh. 1996.
[17] N. Muscettola, P. P. Nayak, B. Pell, and C. B. Williams. Remote Agent: To Boldly Go Where No AI System Has Gone Before. Artificial Intelligence, 1998.