O-OSCAR: A Flexible Object-Oriented Architecture for Schedule Management in Space Applications

Amedeo Cesta, Angelo Oddi, Angelo Susi
IP-CNR, National Research Council of Italy
Viale Marx 15, I-00137 Rome, Italy
{cesta|oddi|susi}@pscs2.irmkant.rm.cnr.it

Abstract

This paper overviews the results of a project aimed at developing a state-of-the-art framework for intelligent constraint-based scheduling for activity management in space applications. The paper starts discussing the main features that an architecture for planning and scheduling should have to be actually used in a working environment. Crucial aspects are seen to be: the ability to built and dynamically maintain a representation of a certain domain; the ability to efficiently search for a solution in a space of possibilities, and the ability to effectively interact with users according to needs of different operative environments. The O-OSCAR software architecture is described that contributes to solve two classes of problems of increasing difficulty, a satellite scheduling problem and a resource constrained project scheduling problem for space missions.

1 Introduction

This paper describes the results of a project supported by the Italian Space Agency (ASI) [1] aimed at developing a general framework for intelligent constraint-based scheduling and activity management in space applications. The initial goal of the project consists of building a reference architecture for temporal planning and scheduling that could be flexibly configured for different space applications. Although several results of the project can be independently used in both planning and scheduling applications, during the project particular attention has been dedicated to scheduling problems that were relevant for the supporting agency.

Leading ideas for the project has been the following:

- to guarantee a complete approach to the resolution and management of a problem. This means being interested not only in developing a particular search algorithm for the problem but also in building up a framework able to support the "problem life-cycle" from the description of the domain knowledge to the presentation of different solution aspects to the users;
- to pay particular attention to the problem of plan/schedule maintenance. In particular we aim at supporting a rich query set to the solution and the definition of a number of update and modification commands on the current solution. We consider these aspects as basic starting points to allow the continuous use of intelligent scheduling systems in a work environment;
- to create an open representation able to support multiple approaches to the resolution of problems. In particular we have been interested to integrate multiple problem solving strategies in an uniform framework to allow comparisons but also to allow the use of the more appropriate approach according to the problem at hand;
- to design a software structure that allows to integrate different research results for the solution creation and management.

Specifically requirements and constraints come from space applications. In particular:

- since space missions span for several years since their original design, a major role assumes the possibility of modifying plans and schedules, as well as the details of the application domain, as soon as the steps of a mission become more mature. Attention towards the dynamic evolution of reality has been a peculiar aspect of our work.
- the explicit consideration given to aspects of user interaction and acceptance of the automated system in a working environment. A con-

[1] This paper describes research developed under a three years project titled "Stazione di lavoro per la generazione interattiva di piani per sistemi spaziali complessi" ("A workstation for the interactive generation of activity plans for complex space systems"). In November 1998 ASI has approved the continuation of the project for two further years with the title "Un toolkit per la creazione di pianificatori interattivi per sistemi spaziali complessi" ("A toolkit for the synthesis of interactive planners for complex space systems").

Proc. Fifth International Symposium on Artificial Intelligence, Robotics and Automation in Space, 1-3 June 1999 (ESA SP-440)

tinuous attention has been dedicated to the investigation of human-computer interaction aspects customized to the application domain.

A final characterization concerns our own approach to the problem. Our main interest is focussed on constraint-based approaches to scheduling problems, we heavily rely on constraint satisfaction as both a representation tool, and as a mechanism for guiding problem solving, in this way being similar to approaches described in [12; 10; 11]. A more specific feature of our work has been the interest for scheduling problems with a strong temporal structure, in particular we have considered problem where quantitative temporal constraints are defined between activities to bound minimal and maximal distances among them.

The major result of our investigation has been the software architecture named O-OSCAR (Object-Oriented SCheduling ARchitecture) that represent a carefully design library of functionalities designed to support the previous requirements in an integrated way.

2 Ingredients for a Scheduling Architecture

To develop a complete solution for a planning/scheduling problem a basic step consists in identifying exactly the basic problems to be addressed, their peculiarities, and the interrelationships between them. Figure 1 sketches the results of our analysis showing four aspects that contribute to the solution.

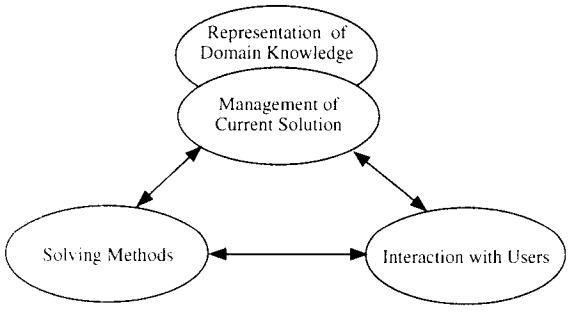

Figure 1: Functional Aspects

Two aspects are strictly interconnected:

Domain Representation Language. A key initial decision consists in defining the class of problems that is possible to address with the architecture. A Domain Representation Language allows the system developer to describe different aspects of the world that the scheduling system needs to know in order to produce a solution. Usually such languages allow the representation of classes of problems and the peculiar domain constraints.

Solution Representation and Management. Constraint-based methods are centered on the production and maintenance of a symbolic solution that relies on a number of a specialized constraint reasoners, representing different aspects of the currents context (e.g., temporal constraints, resource availability). When a change to the solution is performed by a problem solver or a user, the module taking care of solution representation checks the consistency of the change and updates its representation. The solution manager is usually endowed with a set of primitives that allow both atomic or aggregate changes, and with a set of query functionalities for knowing specific information in the solution.

It is not surprising that the basic representation language and the tools for representing the solutions represent the core part of an architecture (the part that more influences the further choices). It should be also clear that they are strictly interconnected, in fact the domain description should allow to express in a suitable way the main features of a domain but also, and more importantly, the constraints that limit finding a solution to a problem in that domain. All this features should be naturally mapped in the representation mechanism of the solution manager because the core of the constraint-based approach is an active service that automatically take care of checking/maintaining the satisfaction of the basic domain constraints.

Once done the architectural choices for realizing these two core components, a complete approach to the solution is obtained addressing the two missing aspects: adding one or more strategies to solve the problem and coping with the interaction with users. This means adding two further blocks to an architecture:

Automated Problem Solving. This is the module that makes available a portfolio of solution methods for a given class of problems (e.g., exhaustive search procedures, greedy heuristics, local search approaches). All the methods use the query and change primitives of the solution manager.

User-System Interaction. This module allows the interaction of the user with both the solution and the problem solving methods. The interaction functionalities may vary from more or less sophisticated visualization services, to a set of complex manipulation functionalities allowed to the user on the solution. A further aspect, very relevant in developing applications, consists in the possibility of adapting the interaction to the working tasks and competence of different users, in order to allow maximal productivity to each person that interacts with the scheduling system.

An advantage of having identified the basic func-

tionalities (and as a consequence the basic modules) a scheduling architecture should be endowed with stays in the possibility of focalizing the research on specific features of each part (e.g., the expressiveness for the Description-language, the efficiency and flexibility of services for the Solution-manager, the ability to controlling search for the Solver; the capability to be adaptable to different needs for the Interaction-module).

It is worth observing the key role that the solution management has in this approach (see the central placement in Figure 1). As a consequence, a major effort in our work has been dedicated to produce a flexible, configurable and efficient software system for schedule management.

3 The O-OSCAR Architecture

As said in the introduction, the project has focussed its attention on the production of an open software architecture for the solution of scheduling problems. Such a software system, named O-OSCAR (Object-Oriented SCheduling ARchitecture), is a principled kernel of functionalities that has allowed to create an open, configurable framework to be adapted to multiple contexts.

Following the distinctions introduced in Section 2, O-OSCAR mainly consists in a software system that makes available the pair ⟨*Description-language;Solution-manager*⟩ according to a class of problems. Such software system guarantee an amount of functionalities that joined with a problem solving algorithm and an interaction module allows for the development of a complete system to solve a class of problems.

As explicitly stressed in the name of the system, a main feature of O-OSCAR is the attention paid to the object-oriented design. Object-oriented techniques allow the stable implementation of specific modules with clear interfaces that can be composed to configure the software system according to the application. Moreover, the use of specialization techniques allows also an incremental refinement of different functionalities.

Figure 2 shows the general schema followed to create O-OSCAR versions for two different classes of problems.

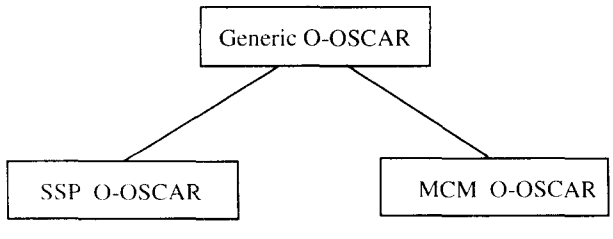

Figure 2: O-OSCAR: The Developed Software

We have designed a generic representation for schedules named *Generic O-OSCAR* that currently plays the role of Domain Description Language at the higher level of abstraction. *Generic O-OSCAR* identifies the typical aspects involved in a schedule, namely resources, activities, constraints and decisions. Having chosen a general representation allows us to interface our work directly with typical abstractions from Operations Research (see for example [13]).

The generic level has been specialized to create software architectures for two classes of scheduling problems:

- The *SSP problem* (SSP stands for Satellite Scheduling Problem). It represents scheduling domains in which resources have binary capacities, activities may have flexible temporal durations and the user may specify preferences over allocation intervals. This class of domains is quite frequent in space applications in particular in satellite allocation requests scheduling (see later the DRS request allocation problem we have studied).
- The *MCM problem* (MCM stands for Multi-Capacitated and Metric). The MCM O-OSCAR management system represents a more sophisticated problem characterized by resources whose capacities are integer numbers greater than 1 (to represent aggregate resources), and several metric temporal separation and time-window constraints may be represented. This class of problems include quite complex cases like Multi-Capacitated Metric Job-Shop [5] and Resource Constrained Project Scheduling [8]. With such an extension it is possible for example to deal with several mission planning problems having the possibility of expressing a quite realistic set of constraints over available resources.

The development of two different software systems is due to needs of the project. Focalizing on SSP has allowed us to prototype quickly a complete system to be used to make the dialogue with the supporting agency more concrete. Nevertheless SSP O-OSCAR allows to address effectively a subset of scheduling problem very frequent in space domains. The development of MCM O-OSCAR derives from the experience of SSP but has involved a major redesign to cope with more sophisticated constraints and a wider class of problems.

Both SSP and MCM O-OSCAR share the same layered software design that allows us to interface the quite general representation language with the constraint-based AI techniques we wanted to use at the lower level. In particular three layers have been defined a sequencing layer, a causal layer, a constraint layer.

The *sequencing layer* is the interface of the system with the problem solver (also called Sequencer in the

following) and the interaction module functionalities. It inherits the abstract characterization of *Generic O-OSCAR* and allow to see a schedule subdivided in resources, activities, constraints and decisions. In particular the decisions represent an association with an activity and the resources it require to be executed and it is used as an input/output parameter to return the actual solution. Of course in the SSP and MCM software systems different methods are available to allow intervention by the sequencer and the users.

The *causal layer* is the level influenced by Artificial Intelligence symbolic representation techniques. It contains a structured description of the temporal evolution of the resources and the activities (in this way it represent a "causal model" of the domain, hence the name of the layer). In particular a further internal representation entity is used, the *token*, to fully represent the association among an activity, the resources it requires, the temporal and technological constraints it should satisfy in any solution.

The *constraint layer* is the level at which both general and specialized constraint satisfaction techniques are used. This level at present contains representation capabilities for temporal constraints (in particular consistent with the quantitative time network manager described in [2; 4]), and for resource constraints (namely the possibility is given to use either the propagation algorithms described in [7] or the more procedural profile-based representations formalized in [5]). It is to be noted that this lower level is a layer that offers services to the higher levels and can be sophisticated more or less according to the requirements of the current problem.

We end this section commenting about similarities between O-OSCAR solution management capabilities and, on one side, the blackboard-based representation used in OPIS [12] and SONIA [10], and, on another side, with the temporal data-base used in HSTS [11]. Similarities with the first two systems are at the methodological level both that system being reference examples for the constraint-based approach to scheduling. The similarities with HSTS are more strict and should be more carefully analyzed. We share with that system the use of a complete temporal propagation. We differ strongly with our sequencing layer because we interface a more simple generic schedule description language (influenced by [13]) instead of the description language of HSTS [11; 3] more suitable for temporal planning problems. At the causal and constraint layers the difference starts from our attempt to deal with complex multi-capacitated problems that turned out in a representation quite different from the one currently reported for HSTS.

We continue the paper giving a short overview of the two complete systems we have built starting from SSP and MCM O-OSCAR respectively.

4 Using SSP O-OSCAR

As a first use of the SSP version of O-OSCAR we have developed a complete system to solve request allocation problem for the Data Relay Satellite (DRS) System that we had previously addressed with a more "classical" knowledge-based approach [1]. The Data Relay Satellite (DRS) System is a European Space Agency program aimed at providing a data relay service between Low Earth Orbiting (LEO) satellites and their ground terminals. Actually this program is in the last step of development, and it will be operative within 1999 (its actual name being Artemis).

The scheduling problem of DRS consists in the production of a mission plan, that allows the clients to utilize the transmission services. An high number of access requests is expected, so that their temporal extension exceeds the total transmission time available, introducing conflicts that have to be solved following some quality objectives. Given the technical characteristics of the DRS system, the crucial aspect in the production of the plan is the management of the link between the DRS and the LEO satellites, while the links between DRS and ground stations are less problematic. The first type of link imposes the satisfaction of physical constraints of the DRS's antennas, temporal constraints of the requests, and requirements of priority, commercial value and allocation preference.

An interesting aspect of the problem is represented by the requests and related constraints. All user requests specify a number of desired characteristics which include: (a) static priority associated to the request's owner; (b) technical requirements: these may include for example the band, speed of transmission and the number of channels required; (c) user flexibilities: minimum and maximum time intervals for the duration, the interval of time within which the access must be scheduled (flexibility interval) and the utility function associated with these flexibilities; (d) user preferences: preferred values for the duration and the actual access time.

Goal of the system is to generate the Detailed Assignment Plan (DAP): (a) schedules of as many access requests as possible; (b) satisfies of as many user preferences as possible; (c) gives priority to preferences of requests having a higher "relevance" coefficient. The goals are potentially conflicting: an optimization in resource use required to satisfy the first goal would imply taking full advantage of user specified flexibilities but in doing so, the preference (or utility) function given by the users may not be satisfied. The other two goals are in turn partially contrasting, since maximizing user preferences does not necessarily coincide with satisfying the requests of preferred users.

According to the technical documentation, the production of the DAP is supposed to follow an iterative process repeated three times, and that involves two

types of human operators (see the schema in Figure 3):

- *Commercial operators* at the Mission Control Center negotiate the sale of the free transmission spaces with the clients, and insert the related activities in the plan;
- *Spacecraft engineers* (called Operative users) at the Operation Control Center modify the plan inserting some special activities for the maintenance of the system operativity and requests with a special requirement of urgency.

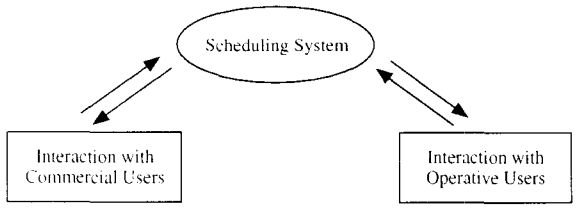

Figure 3: Users Views in DRS O-OSCAR

These two operational profiles follow different and potentially conflicting objectives (maximum satisfaction of requests vs. DRS's resources saving). Those objectives have to be integrated together in an automated scheduling system that supports decision making in this environment.

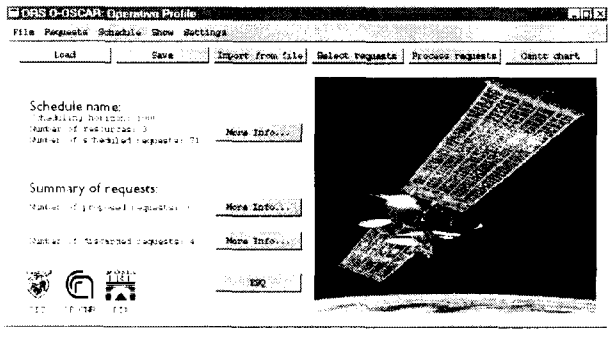

Figure 4: The interface for the operative user

The DRS Demonstrator has been built on top of the SSP O-OSCAR functionalities. According to the O-OSCAR methodology a complete system is developed starting from a core set of functionalities that are able to represent and manage a solution for a class of problems (SSP in this case). In addition to the basic functionalities for solution representation and modification, two different modules need to be built:

- A *Sequencer* that is able to produce incremental modifications on a current solution to satisfy current goals. In the DRS case a set of specialized heuristics is able to produce assignment plans in a time compatible with the duration of activities in the operative environment.
- An *Interaction Module* that allows multiple users to use the planning facilities of O-OSCAR extracting services according to the working tasks. In particular two interface profiles have been defined one for the tasks of Commercial operators and one for the tasks of Operative users (a picture is shown in Figure 4).

A peculiar characteristic of O-OSCAR is its ability to support dynamic modification to the schedule after producing a solution: it is possible to introduce a single new activity in the schedule, remove activities to serve a maximal priority one, etc.

5 Using MCM O-OSCAR

Having demonstrated the potentiality of the O-OSCAR architectural approach we have worked at producing a framework able to cope with a wider class of problems. It is worth remarking that extending O-OSCAR to cope with MCM problems allows to model temporal constraints like "observation temporal windows" very peculiar in space exploration and science, and resource constraints like "amount of energy" and "workforce" that are common in modeling the ground preparation of space missions and in the managing of space instruments.

A quite complex example of the new range of functionalities given by MCM O-OSCAR is represented by the so-called RCPSP/max problem (Resource Constrained Project Scheduling Problem with Time Windows, or with Generalized Precedence Relations). In such problem a set of activities are connected by a temporal structure that represent a project to be completely executed to solve the problem. Each activity requires different sizes of certain resources to be executed and should satisfy a number of temporal constraints with respect to other activities. The distance separating two activities may satisfy minimal and maximal duration constraints. Domain resources have a capacity greater then one.

Again, attention has been given to the possibility of incremental constructing the solution, to the ability of modifying something when a schedule exists, etc. Around the basic representation and management functionality we have built a complete system following the O-OSCAR methodology. Two modules have been added:

- A *Sequencer*. To cope with Project Scheduling problems we have built a multi-strategy module that allow the integration of several resolution procedures. In particular we have integrated state-of-art branch and bound [8] and heuristic [9] algorithms from the Operation Research community and our original constraint-based algorithm called ISES [6]. In this way we are able to test multiple approaches to the same problem but also to use the more appropriate algorithm according to the dimension of the problem.

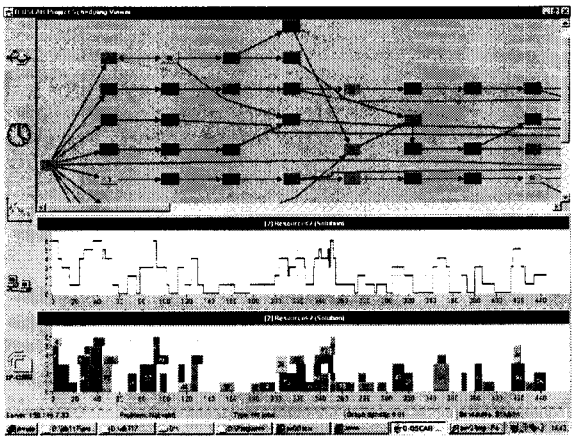

Figure 5: The PS O-OSCAR Viewer

- An *Interaction Module*. In this case a complex direction has been successfully attempted: the development of a client-server architecture and a Java client that interacts with the scheduling system through a specialized communication protocol. The result is a quite sophisticated interface a snapshot of which is shown in Figure 5.

The result of this effort is the PS O-OSCAR system that at present is able to effectively solve recognized benchmark problems and has also been officially demonstrated to ASI. It is worth remarking that also in the case of PS O-OSCAR, the functionality of dynamic modification of the schedule (that was a peculiar aspect in the DRS demonstrator) has been reproduced in this more complex scenario.

6 Conclusions

This paper has described the main aspects of O-OSCAR a scheduling architecture for plan production and management. Two different systems relative to different space applicatiosn have been successfully developed with it. Interesting features of O-OSCAR are the complete approach to the scheduling problem, the possibility of adding dynamic modifications to a current solution, the possibility of usefully integrating different solution strategies in the same system. A further aspect, very peculiar in our work, is the attention given to interaction with the user in the solution management process and the adaptation of such interaction to the user tasks and competence.

Acknowledgments

This research is supported by ASI (Italian Space Agency) and is part of a joint effort among Dipartimento di Informatica e Sistemistica dell'Universita' di Roma "La Sapienza", Dipartimento di Informatica e Automazione della Terza Universita' di Roma, and IP-CNR, Consiglio Nazionale delle Ricerche, Roma. Thanks to Luigia Carlucci Aiello (Project Coordinator), Marta Cialdea Mayer and Marco Schaerf for creating a stimulating environment for the project. Special thanks to several people that contributed to the O-OSCAR development, namely Paolo Bazzica, Gianni Casonato, Gabriele Giuseppini, Attilio Mainetti and Fabrizio Peroni.

References

[1] M. Adinolfi and A. Cesta. Heuristic Scheduling of the DRS Communication System. *Engineering Applications of Artificial Intelligence*, 8:147–156, 1995.

[2] R. Cervoni, A. Cesta, and A. Oddi. Managing Dynamic Temporal Constraint Networks. In *Artificial Intelligence Planning Systems: Proceedings of the Second International Conference (AIPS94)*, 1994.

[3] A. Cesta and A. Oddi. DDL.1: A Formal Description of a Constraint Representation Language for Physical Domains. In M. M.Ghallab and A. Milani, editors, *New Directions in AI Planning*. IOS Press, 1996.

[4] A. Cesta and A. Oddi. Gaining Efficiency and Flexibility in the Simple Temporal Problem. In *Proceedings of the Third International Workshop on Temporal Representation and Reasoning (TIME-96)*, 1996.

[5] A. Cesta, A. Oddi, and S. Smith. Profile Based Algorithms to Solve Multiple Capacitated Metric Scheduling Problems. In *Proceedings of the Fourth Int. Conf. on Artificial Intelligence Planning Systems (AIPS-98)*, 1998.

[6] A. Cesta, A. Oddi, and S. Smith. An Iterative Sampling Procedure for Resource Constrained Project Scheduling with Time Windows. In *Proceedings of the 16th Int. Joint Conference on Artificial Intelligence (IJCAI-99)*, 1999.

[7] A. Cesta and C. Stella. A Time and Resource Problem for Planning Architectures. In *Proceedings of the Fourth European Conference on Planning (ECP 97)*, 1997.

[8] B. De Reyck and W. Herroelen. A Branch-and-Bound Procedure for the Resource-Constrained Project Scheduling Problem with Generalized Precedence Relations. *European Journal on Operations Research*, 1998. to appear.

[9] B. Franck and K. Neumann. Resource Constrained Project Scheduling Problems with Time Windows – Structural Questions and Priority-Rule Methods. Technical Report WIOR-492, Universität Karlsruhe, 1998. (Revised November 1998).

[10] C. Le Pape. Scheduling as Intelligent Control of Decision-Making and Constraint Propagation. In M. Zweben and S. M. Fox, editors, *Intelligent Scheduling*. Morgan Kaufmann, 1994.

[11] N. Muscettola. HSTS: Integrating Planning and Scheduling. In M. Zweben and S. M. Fox, editors, *Intelligent Scheduling*. Morgan Kaufmann, 1994.

[12] S. Smith. OPIS: A Methodology and Architecture for Reactive Scheduling. In M. Zweben and S. M. Fox, editors, *Intelligent Scheduling*. Morgan Kaufmann, 1994.

[13] G. Wolf. Schedule Management: An Object-Oriented Approach. *Decision Support Systems*, 11:373–388, 1994.