O-Plan: a Common Lisp Planning Web Service

Austin Tate and Jeff Dalton
Artificial Intelligence Applications Institute
School of Informatics
Appleton Tower, Crichton Street, Edinburgh EH8 9LE, UK
Tel: +44 131 650 2732

{a.tate, j.dalton}@ed.ac.uk

http://www.aiai.ed.ac.uk/project/oplan/

ABSTRACT

O-Plan is an Artificial Intelligence Planning System written in Common Lisp that has been developed at the University of Edinburgh over the period 1983 to 1999. It is used in a wide variety of applications. It has been deployed as a planning service accessible over the World Wide Web since 1994 and has provided an HTTP interface to users and programs since 1997. It continues to be accessible as a feature-rich planning service within its successor – I-X – that provides for cooperative multi-agent planning, command and workflow execution support.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search – *Plan execution, formation and generation*

General Terms

Design, Experimentation.

Keywords

Artificial Intelligence; Planning; Web Service; Common Lisp.

1. INTRODUCTION

O-Plan (Currie and Tate, 1991) is a comprehensive planner, written in Common Lisp, that uses artificial intelligence planning techniques. Its is based on earlier work on Nonlin (Tate, 1977) which is a hierarchical planning system able to generate plans as partially-ordered networks of activities and which can represent and check a variety of constraints for time, resources, etc.

O-Plan is packaged to run as a planning service over the World Wide Web (Tate et al., 2003). It was first used over the web via CGI scripts in 1994, and since 1997 has provided an HTTP interface (also written in Common Lisp) to users via browsers and other programs. It is therefore a very early example of a rich service accessible over the web.

O-Plan has been followed by I-X (Tate, 2000; Tate, 2003), which provides a more general approach to support mixed-initiative multi-agent cooperative synthesis tasks such as design, planning, configuration, etc. O-Plan continues to be available as a planning service on the web, and can be utilized as such a service via I-X.

A wide variety of AI planning features are included in O-Plan:

- Domain knowledge elicitation
- Rich plan representation and use
- Hierarchical Task Network Planning
- Detailed constraint management
- Goal structure-based plan monitoring
- Dynamic issue handling
- Plan repair in low and high tempo situations
- Interfaces for users with different roles
- Management of planning and execution workflow

O-Plan has been used in a variety of realistic applications. Examples include:

- Air Campaign Planning (Tate et al., 1998)
- Noncombatant Evacuation Operations (Tate, et al., 2000b)
- Search & Rescue Coordination (Kingston et al., 1996)
- US Army Small Unit Operations (Tate et al., 2000a)
- Spacecraft Mission Planning (Drabble et al., 1997)
- Construction Planning (Currie and Tate, 1991 and others)
- Engineering Tasks (Tate, 1996)
- Biological Pathway Discovery (Khan et al., 2003)
- Unmanned Autonomous Vehicle Command and Control

O-Plan's design was also used as the basis for Optimum-AIV (Arup et al., 1994), a deployed system used for assembly, integration and verification in preparation of the payload bay for flights of the European Space Agency Ariane IV launcher.

More details of O-Plan technology are available at http://www.aiai.ed.ac.uk/project/oplan/. A range of simple and more comprehensive demonstrations of the O-Plan planning web service is available at http://oplan.aiai.ed.ac.uk.

International Lisp Conference 2003, October 12-25, 2003, New York, NY, USA, © 2003, AIAI, The University of Edinburgh.

2. SIMPLE PLANNING SERVICES

One example of a simple application of the O-Plan Planning Web Service is to provide a Unix Systems Administrators Script writing aid. This shows a very simple packaged use of planning technology accessible over the web. A form can be filled in stating the requirements on mapping physical to logical Unix disk volumes. The planner can generate a suitable script for this task. The maintenance of such scripts in Unix was proving problematic for a Unix system vendor to maintain. The application shows how AI planning can be used in cases where the basic ingredients used are not very numerous, but the way they can be combined varies.

Other simple "single-shot" plan generation applications often use a similar type of form-filling style of interface on the web.

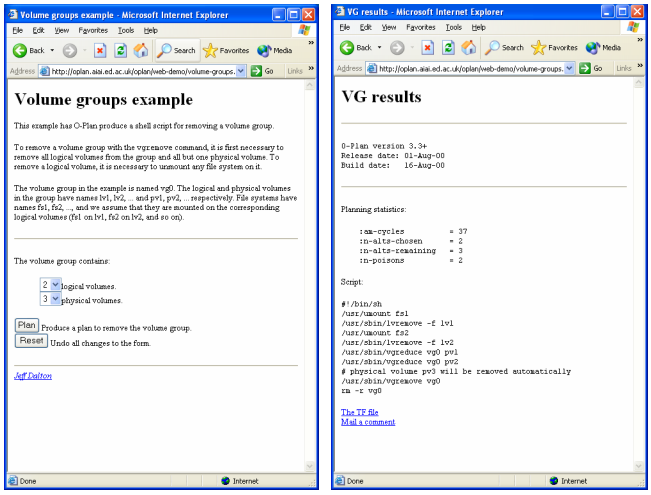

Figure 1. O-Plan generating a Unix system script.

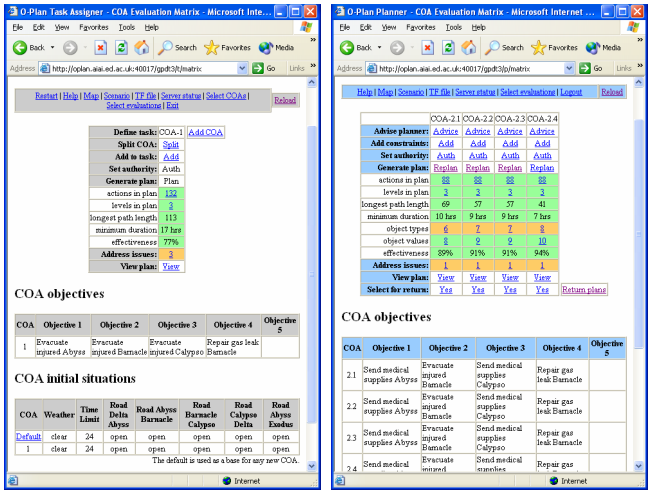

Figure 2. O-Plan with multiple cooperating web users.

3. MULTI-USER PLANNING SERVICES

A more comprehensive web interface is available for the O-Plan planning service that provides a table with columns showing options for various Courses of Action (COAs) and rows showing both the process steps involved in generating the plans and a set of evaluations of the plan options. This matrix interface can be used in a generic form but also has specialized versions for particular applications, such as one that creates plans for disaster recovery and emergency response. The interface allows multiple users in different roles to work with O-Plan in a mixed-initiative fashion. User roles can be as commander/task assigner or planner. The resulting "Open Planning Process Panel" (O-P^3) (Tate et al. 1999; Levine et al., 2000) supports the coordination of the planning process between users and the automated planning agent in the development of multiple options at various stages of generation and evaluation.

In work with the US Army we have identified the stages in the overall Small Unit Operations (SUO) command, planning and execution process at US Army company level, from receipt of mission through to a successful outcome and after-action activities (US Army, 1999). Within this process there are opportunities for a range of planning and decision aids, all facilitated by a common approach to representing the objectives and plans involved. The SUO application uses O-Plan running as a web service with a course of action/evaluation matrix process panel interface (Tate et al., 2000b) and addresses all the phases of a small unit operation., and the whole lifecycle of the generation and use of plans:

- Domain and initial plan representation
- Deliberative initial planning and generation of multiple options
- Plan execution monitoring and dynamic repair of plans
- Tailored interfaces for various user roles including process and workflow support.

4. OPEN PLANNING ARCHITECTURE

An important property of the O-Plan design, and one that is greatly facilitated by Lisp, is that O-Plan is a framework that allows key components to be plugged in. O-Plan finds a plan by exploring a search space of partial plans. "Issues" represent the missing parts of a partial plan. These say, for instance, that an action has not yet been expanded into subactions or that a condition needs to be satisfied. O-Plan's top level is a controller that repeatedly selects an issue and invokes a "knowledge source" (also known as an "issue handler") that is able to resolve that type of issue, until either no issues remain or the remaining issues are ones O-Plan has not been authorized to handle.

Knowledge sources can make decisions about what to put in the plan and what parts of the search space to visit. They build a plan by adding nodes to a partially ordered network of actions and by adding constraints that represent pre- and post-conditions on actions, time limitations, resource usage, and other things. Constraint managers determine whether constraints can be satisfied in the plan and can tell knowledge sources of possible ways to satisfy them.

This is implemented in a way that allows knowledge sources and constraint managers to be added as desired, and indeed for the entire set of knowledge sources and constraint managers to be replaced. In that way, the same architecture and controller can be, and have been, adapted to tasks other than planning - tasks such as plan execution monitoring (Reece and Tate, 1994) and scheduling (Beck and Tate, 1995).

Other types of pluggable component are plan evaluators, plan and world-state viewers, plan "sanity checkers", and, when O-Plan is used over the Web, handlers for HTTP requests.

Plug-ins can be as simple as defining a function and registering it with the appropriate part of the system. For instance, knowledge sources are registered with the controller and constraint managers with a database manager.

Other cases are more complex. A constraint manager is a CLOS object, and there are a number of mixin classes that allow simpler managers to be written. It is also possible to define the syntax of a new type of constraint by adding a constraint parser to the recursive-descent parser for O-Plan's Task Formalism (TF) language (Tate, et al., 1998)

5. PLAN OPTIONS

An option is a point in O-Plan's search space (and hence is a partial plan) that has been given a name to allow an external agent to refer to that point while using O-Plan. This is what allows the matrix interface to manage a number of different plans, and for the user to switch freely between them. The user can add new actions to a plan, add constraints, or give O-Plan the authority to expand the plan to deeper levels of subactions. An option can also be split into two children, so that they can be developed in different ways. Moreover, it is possible to simulate the execution of one plan while still developing others.

Options are another feature that has been greatly facilitated by Lisp. They are implemented using a context layering mechanism that adds an extra, implicit parameter - the current context - to object accessors (Tate, 1984)[1]. Contexts form a tree, with a child seeing by default the same values as its parent. When O-Plan has to choose between different ways to resolve an issue while developing a plan, it moves to a new child context, saving the parent as a backup point, so that it can later return and try one of the other possibilities. Thus each partial plan that O-Plan develops corresponds to a context, and options are in effect tags on some of those contexts. Lisp macros make it easy to define accessor functions that use contexts without writing out the code each time, and in Lisp the resulting functions look the same to callers as ordinary accessors. The current context is the value of a dynamically scoped variable, making it easy to change contexts.

6. ACKNOWLEDGMENTS

This material is based on research within the O-Plan and I-X projects sponsored by the Defense Advanced Research Projects Agency (DARPA) and US Air Force Research Laboratory under agreement number F30602-03-2-0014 and earlier contracts. The U.S. Government, the University of Edinburgh and research sponsors are authorized to reproduce and distribute reprints and on-line copies for their purposes notwithstanding any copyright annotation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of other parties.

[1] The mechanism and algorithms used in O-Plan's context structures was later adapted by USC/ISI for use in the context/worlds facility of LOOM.

7. SUMMARY

O-Plan is a feature-rich artificial intelligence planning system written in Common Lisp and packaged to be available as a service over the World Wide Web. Its facilities cover the whole spectrum of initial planning domain knowledge modelling, through refinement of an initial outline plan or set of requirements to generate any of a number of options to meet those objectives, to the monitoring of the execution of a plan, and its repair where needed to cope with a dynamically changing environment.

8. REFERENCES

[1] Aarup, M., Arentoft, M.M., Parrod, Y., Stokes, I., Vadon, H. and Stader, J. (1994) Optimum-AIV: A Knowledge-Based Planning and Scheduling System for Spacecraft AIV, in Intelligent Scheduling (eds. Zweben, M. and Fox, M.S.), pp. 451-469, Morgan Kaufmann.

[2] Beck, H. and Tate, A. (1995) Open Planning, Scheduling and Constraint Management Architectures, in the British Telecommunication's Technical Journal, Special Issue on Resource Management, 1995.

[3] Currie, K.W. and Tate, A. (1991) O-Plan: the Open Planning Architecture. Artificial Intelligence 52(1), Autumn 1991, North-Holland.

[4] Drabble, B., Dalton, J. and Tate, A. (1997) Repairing Plans On-the-Fly, in Proceedings of the NASA Workshop on Planning and Scheduling for Space, Oxnard CA, USA, October 1997.

[5] Khan, S., Decker, K., Gillis, W. and Schmidt, C (2003) A Multi-Agent System-driven AI Planning Approach to Biological Pathway Discovery, Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS 2003) (eds. Giunchiglia, E., Muscettola, N., and Nau, D), Trento, Italy, June 9-13, 2003. AAAI Press.

[6] Kingston, J., Shadbolt, N. and Tate, A. (1996), CommonKADS Models for Knowledge Based Planning, in Proceedings of the National Conference on Artificial Intelligence (AAAI-96), Portland, Oregon, USA, August 1996.

[7] Levine, J., Tate, A. and Dalton, J. (2000) O-P^3: Supporting the Planning Process using Open Planning Process Panels., IEEE Intelligent Systems, Vol. 15, No. 6, November/December 2000.

[8] Potter, S., Tate, A. and Dalton, J. (2003) I-X: Task Support on the Semantic Web, Poster Abstract, Second International Semantic Web Conference (ISWC-2003), Sanibel Island, Florida, October 2003.

[9] Reece, G. and Tate, A. (1994) Synthesizing Protection Monitors from Causal Structure, in the Proceedings of the

Second International Conference on Planning Systems (AIPS-94), Chicago, June 1994, AAAI Press.

[10] Tate, A. (1977) Generating Project Networks, Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-77), pp. 888-893, Cambridge, MA, USA, Morgan Kaufmann.

[11] Tate, A. (1984) Functions in Context Database, Second Alvey Workshop on Large Knowledge Base Architectures, Manchester, UK, July 1984, Also AIAI-TR-1, Artificial Intelligence Applications Institute, University of Edinburgh.

[12] Tate, A., Polyak, S. and Jarvis, P. (1998) TF Method: An Initial Framework for Modelling and Analysing Planning Domains. Workshop on Knowledge Engineering and Acquisition at the Fourth International Conference on AI Planning Systems (AIPS-98), Carnegie-Mellon University, Pittsburgh, PA, USA, June 1998.

[13] Tate, A. (2000) Intelligible AI Planning, in Research and Development in Intelligent Systems XVII, Proceedings of ES2000, The Twentieth British Computer Society Special Group on Expert Systems International Conference on Knowledge Based Systems and Applied Artificial Intelligence, pp. 3-16, Cambridge, UK, December 2000, Springer.

[14] Tate, A. (2003) Coalition Task Support using I-X and <I-N-C-A>, in Proceedings of the 3rd International Central and Eastern European Conference on Multi-Agent Systems (CEEMAAS 2003), Prague, Czech Republic, 16-18th June 2003, pp.7-16, Springer Lecture Notes in Artificial Intelligence LNAI 2691.

[15] Tate, A., Levine, J., Jarvis, P. and Dalton, J. (2000) Using AI Planning Technology for Army Small Unit Operations, Poster Paper in Proceedings of the Fifth International Conference on Planning and Scheduling Systems (AIPS-2000), Breckenridge, Colorado, USA, April 2000.

[16] Tate, A., Drabble, B. and Dalton, J. (1996) O-Plan: a Knowledge-Based Planner and its Application to Logistics, in Advanced Planning Technology (ed. Tate, A.), Morgan Kaufmann, May 1996.

[17] Tate, A. (1996) Responsive Planning and Scheduling Using AI Planning Techniques - Optimum-AIV, in "Trends & Controversies - AI Planning Systems in the Real World", IEEE Expert: Intelligent Systems & their Applications, Vol. 11 No. 6, pp. 4-12, December 1996.

[18] Tate, A., Dalton, J. and Levine, J. (1998) Generation of Multiple Qualitatively Different Plan Options, in Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems (AIPS-98), Pittsburgh PA, USA, June 1998

[19] Tate, A., Levine, J., Dalton, J., and Aitken, S. (1999) O-P^3: Supporting the Planning Process using Open Planning Process Panels, in Proceedings of the AAAI Workshop on Agent Based Systems in the Business Context, AAAI-Press WS-99-02.

[20] Tate, A., Levine, J., Jarvis, P. and Dalton, J. (2000a) Using AI Planning Techniques for Army Small Unit Operations, Poster Paper in the Proceedings of the Fifth International Conference on AI Planning and Scheduling Systems (AIPS-2000), Breckenridge, CO, USA, April 2000.

[21] Tate, A., Dalton, J. and Levine, J. (2000b) O-Plan: a Web-based AI Planning Agent, AAAI-2000 Intelligent Systems Demonstrator, in Proceedings of the National Conference of the American Association of Artificial Intelligence (AAAI-2000), Austin, Texas, USA, August 2000.

[22] Tate, A., Levine, J., Dalton, J. and Nixon, A. (2003) Task Achieving Agents on the World Wide Web, in Spinning the Semantic Web, Fensel, D., Hendler, J., Liebermann, H. and Wahlster, W. (eds.), Chapter 15, pp. 431-458, MIT Press, 2003.

[23] US Army (1999) Center for Army Lessons Learned, Virtual Research Library, http://call.army.mil