O-Plan2 Technical Paper

Mixed Initiative Planning in O-Plan2

Austin Tate

Artificial Intelligence Applications Institute
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
United Kingdom

February 15, 1994

ARPA-RL/O-Plan2/TP/12 Version 1

Mixed Initiative Planning in O-Plan2

Austin Tate

Artificial Intelligence Applications Institute
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN
United Kingdom
Tel: +(44) 31 650 2732 Fax: +(44) 31 650 6513
Email: A.Tate@ed.ac.uk

Abstract

The model of Mixed Initiative Planning that can be supported by the O-Plan2 architecture is the mutual constraining of a set of alternative partial plans for some task set.

This paper describes the opportunities for mixed initiative planning within the O-Plan2 architecture. Both the user and the system can work in harmony and neither is seen as at a higher level or "in charge" as far as the architecture is concerned. Ordering and priorities can be applied to impose specific styles of authority to plan within the system. One extreme of user driven plan expansion followed by system "filling-in" of details, or the opposite extreme of fully automatic system driven planning (with perhaps occasional appeals to an user to take predefined decisions) are possible. In more practical use, we envisage a mixed initiative form of interaction in which the user and system proceed by mutually constraining the plan using their own areas of strength.

Appendices describe in more detail the use of the KS-USER knowledge source to "wrap" around the interfaces provided to the user and to ensure integrity of the system, the various user roles identified within the O-Plan2 design, details of the search space explored by O-Plan2 and other relevant information concerning mixed initiative planning within O-Plan2.

1 Partial Plans as a Set of Constraints

An O-Plan2 plan is viewed as containing a set of constraints on the possible plan elaborations that can be entertained. Users and system in a mixed initiative way jointly add (or relax) constraints in the plan as planning proceeds.

A plan conceptually has three levels

1. implied constraints (called the plan agenda)
2. plan level entities (decided upon plan components at various levels of abstraction)
3. detailed constraints (for time, resources, authorities, conditions/effects, object selections, spatial use, etc. These are associated with the plan level entities)

Explicit options (Courses of Action) may exist for plans and may share a lot of common structure.

At any time the system or an user can work on this set of constraints – normally being directed by the agenda.

2 Plan Agenda and Control

The agenda keeps everyone straight about what remains to be considered. Inconsistent plans and partially elaborated plans are possible using the agenda to represent such outstanding issues.

Focus of initiative is determined by decisions on the order in which agenda entries are processed. This ordering decision process is separate to the involvement of the user in constructing a plan.

Mixed initiative is possible. The user can delegate to the system by adding suitable agenda entries (implied constraints) for parts of the work that the system can handle best. The system can seek help from the user via the same mechanism. Or both can take a look at what they can do with the agenda. Critiquing can lead to new agenda entries to work on.

3 Plan Modification Operators – Knowledge Sources

```
plan agenda   ------------>   choose PMO
     |                            |
     |                            v
plan entities                   do PMO
     |                            |
     |                            v
plan constraints <--------   update plan
```

Users or the system can (when sanctioned or authorised) work on anything that is outstanding on the agenda. They do this through the "wrapper" of a Plan Modification Operator (PMO). This is like a knowledge source in blackboard systems. An user also interacts via a PMO wrapper to ensure that plan integrity is maintained.

All decision making processing which can alter a plan is done via plan modification operators. The recording of dependencies and who is responsible for changes to the plan is possible in such PMOs to support later plan changes and constraint relaxation. This has been done in some versions of our planners at Edinburgh.

PMOs (called knowledge sources in the O-Plan2 architecture) can run on one or more knowledge source "platforms". Concurrency is possible with multiple platforms. Real time capabilities can be assured by having dedicated platforms for a nominated knowledge source. One or more platforms can run knowledge sources to provide the planner user interface(s).

4 Plan Entities and Detailed Constraints

The user or the system made alterations to a plan are done at the plan entities layer – which expresses most of what can be thought of as the "interesting" contents of the plan, and certainly contains what the users are likely to want to work with. Whether user or system decisions lead to a change of the plan entities, they are subject to lower level constraint management at a detailed level where critiquing of the changes and inconsistency issues are raised directly with the caller or via the agenda. The constraint manager level NEVER takes any decisions, so the user and the decision making level of the system can maintain a simple view of what is going on and who is changing what.

5 User Roles

O-Plan2 identifies quite separate user ROLES with respect to planning. The discussion above relates mostly to the role we call PLANNER USER. We identify other user roles which are quite distinct. For example, an user in a system developer sense is not confused with the PLANNER USER role.

One important distinct role with respect to mixed initiative planning is the role of "user support to agenda controller" which is the place where an user can assist in deciding which agenda entries to process next (i.e., choose a PMO to process on an available knowledge source platform) and thus where the focus of initiative between the planner user and the system plan modification operators lies. This separation of roles allows a better understanding of what an user is doing and what the user's intentions are.

6 User Interfaces

We have characterised features of the User Interface for the O-Plan2 planning system and provide appropriate support for various user roles. We have developed flexible interface specifications between a plan state and users who want quite different views of the plan. Technically orientated PERT diagrams and Gantt charts, resource profiles, etc., can be the means of interaction for some. For others good domain orientated displays, maps, animations or simulations convey much more. O-Plan2 does this via a PlanWorld Viewer interface specification which allows quite disparate external viewers to be connected to the planning support tools. Map displays fit naturally into this, as do AutoCAD style systems, etc. – all with minimal work to write the adaptor code. We have already had discussions with BBN about the ARPI CPE and ways to integrate the TARGET interface and approach with that used in O-Plan2 – and this seems possible.

7 3 Level Model – Strategic, Tactical and Operational Support

The O-Plan2 architecture identifies three levels at which different types of task are performed in a command, planning and control environment.

Strategic: task characterisation: analysis and direction.

Tactical: task characterisation: synthesis.

Operational: task characterisation: modification and control.

For more "strategic" plan decisions, the agenda level and manipulation of WHAT to do rather than HOW to do it is more important. We believe that strategic "planning" does not employ the same techniques as "tactical" planning. The strategy level involves much more analysis, option comparison and direction. The tactical level is closer to what is termed generative AI planning and the resource scheduling we see in manufacturing and logistics support systems today. Similarly, the operational level has different priorities and requirements to the higher strategic and tactical levels. All three levels are needed, and the O-Plan2 architecture is designed to accommodate them all. A clear direction in the form of a 'Task Assignment" is needed from the strategic decision making level to drive the tactical level.

A clear characterisation and analysis of different "task types" has emerged from knowledge engineering. There are some ten basic task types often identified (diagnosis, interpretation, classification, planning, monitoring, learning, etc.). The KADS methodology separate these into three classes: analysis, synthesis and modification tasks. This is consistent with the separation of the roles of the three levels in the O-Plan2 architecture. It is also common practice to relate to these three levels in many organisations – whether military or otherwise.

The core O-Plan2 planner model described earlier fits into this three level (Task Assignment, Tactical Planning, Operations Support) framework as shown in the diagram below.

```
    TASK ASSIGNMENT          TACTICAL PLANNING         OPERATIONS SUPPORT


task/option analysis ---->  plan  <-------------------------- reports
authority management       agenda ----> choose PMO               |
                             |             |                 operations
                             |             |                  support
                            plan           v                     |
                          entities       do PMO  -----------> commands
                             |             |
                             |             v
                            plan  <---- update
                         constraints     plan
```

8 Focus of Initiative

In O-Plan2, there is separation of decisions on focus of initiative at 4 main points. These relate to different types of task.

- mission tasking, option analysis, authority management and direction.

  Task characterisation: analysis and direction.

  Initiative: normally manual.

  O-Plan2 Agent/Component: Task Assignment agent.

- decisions on what to work on next for the human and system components given available human planner and system computational resources.

  Task characterisation: interpretation and classification.

  Initiative: normally automatic using pre-defined priorities with the possibility of manual override.

  O-Plan2 Agent/Component: Planner agent/controller dispatching to planner agent/knowledge source platforms.

- decisions to add (or relax) constraints on important plan entities.

  Task characterisation: synthesis.

  Initiative: opportunistic with mixed automatic and manual possibilities. System support if constraints are relaxed is essential due to the potential ramifications of such change.

  O-Plan2 Agent/Component: Planner agent/knowledge sources.

- detailed constraint propagation and projection.

  Task characterisation: algorithmic.

  Initiative: normally automatic, with human assistance for speed-up.

  O-Plan2 Agent/Component: Planner agent/constraint managers.

9 Authority Management

It is important to clarify the description of authority for the planner user and system. Authority to plan to given levels of detail for certain parts (phases) of certain plan options, and permission to execute parts of plans should be given explicitly. For example, "give me a CONPLAN for the DEPLOYMENT phase of a specifically nominated COA we are discussing", or "execute the MOBILISATION phase of a specific COA we are discussing".

The O-Plan2 plan representation recognises:

- named plan options
- named plan phases
- named plan levels of abstraction

O-Plan2 research has begun to explore issues of clearer authority management and representation between agents involved in command, planning and control. The O-Plan2 Architecture and plan representation allows a simple form of authority management at present. The current O-Plan2 user interface makes allowance for later more sophisticated authority management.

10 Summary

The O-Plan2 architecture has been designed to support advanced research and prototype development for flexible next generation support systems for command, planning and control environments. This paper has shown how the current architecture already goes some way towards addressing key research and development issues to support flexible mixed initiative planning.

References

[1] Allen, J., Hendler, J. and Tate, A., *Readings in Planning*, Morgan-Kaufmann, 1990.

[2] Beck, H. *TOSCA: A Novel Approach to the Management of Job-shop Scheduling Constraints*, in Realising CIM's Industrial Potential: Proceedings of the Ninth CIM-Europe Annual Conference, Amsterdam, 12-14 May 1993, (eds, C. Kooij and P.A. MacConaill and J. Bastos), pp138-149.

[3] Choueiry, B.Y. and Faltings, B. Resource Allocation by Problem Decomposition and Temporal Abstractions, in Proceedings of the Second European Workshop on Planning (EWSP-93), Vadstena, Sweden, IOS Press.

[4] Currie, K.W. and Tate, A., O-Plan: the Open Planning Architecture, *Artificial Intelligence* 51(1), Autumn 1991, North-Holland.

[5] Tate, A., Generating Project Networks, Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-77), Cambridge, Mass., USA, 1977.

[6] Drummond, M.E., & Tate, A., *PLANIT Interactive Planners' Assistant – Rationale and Future Directions*, AIAI-TR-108, AIAI, University of Edinburgh, 1992.

[7] PLANIT Club, *PLANIT Club Final Report* published on behalf of the PLANIT Club by Systems Designers plc, Fleet, Hampshire, UK, document ref. C03209, 1987.

[8] Tate, A., Drabble, B. and R.B.Kirby, R.B., O-Plan2: an Open Architecture for Command, Planning and Control, in *Knowledge Based Scheduling* (eds. M.Fox and M.Zweben), Morgan Kaufmann.

[9] Tate, A, INTERPLAN: a plan generation system which can deal with interactions between goals, Research Memorandum MIP-R-109, Edinburgh: Machine Intelligence Research Unit, December 1974.

[10] Tate, A, Using Goal Structure to direct search in a problem solver, Ph.D. Thesis, University of Edinburgh, September l975.

[11] Tate, A., Generating Project Networks, Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-77), Cambridge, Mass., USA, 1977.

[12] Tate, A., The Emergence of "Standard" Planning and Scheduling System Components, in Proceedings of the Second European Workshop on Planning (EWSP-93), Vadstena, Sweden, IOS Press.

APPENDIX A – The KS-USER Knowledge Source in O-Plan2

The O-Plan2 architecture allows for a KS-USER knowledge source. Knowledge sources are the only places in which decisions relating to the plan entities are taken – other parts of the system being concerned with the ordering in which decisions are taken, and the management of plan states and the constraints included in them.

The KS-USER knowledge source allows a planner user to take decisions within the framework of the architecture. The user can take the initiative by asking for the KS-USER knowledge source to be activated to allow the plan to be viewed and decisions made, constraints applied, etc. Alternatively, the system can seek user input and decisions by asking the KS-USER knowledge source to seek certain kinds of input from the user. Hence both planner user and system are working in harmony and neither is seen as at a higher level or "in charge" as far as the architecture is concerned. Ordering and priorities can then be applied to impose specific styles of authority to plan within the system. One extreme of user driven plan expansion followed by system "filling-in" of details, or the opposite extreme of fully automatic system driven planning (with perhaps occasional appeals to an user to take predefined decisions) are possible. In more practical use, we envisage a mixed initiative form of interaction in which the user and system proceed by mutually constraining the plan using their own areas of strength.

O-Plan2 Design Rationale for KS-USER Knowledge Source

The KS-USER knowledge source is intended to be the single point of interaction with the O-Plan2 planner agent for the user in the role of planner user. The planner user is intended to act at the same level as other decision making components of an O-Plan2 agent (i.e., has the same properties as a knowledge source).

For integrity of the manipulation of an O-Plan2 agent's plan state, the KS-USER knowledge source must respect the O-Plan2 knowledge Source Protocol in its dealings with the Controller (for spawning alternative plan states where necessary, or for adding agenda entries into a plan state). Its O-Plan2 Knowledge Source Framework description must be accurate in describing its read/write interaction requirements (of each knowledge source stage) on the plan state through the O-Plan2 Data Base Manager. Greater levels of concurrency are possible by specifying the interaction details in as constrained a way as possible where this is known.

There are two principal ways in which the planner user will interact with the system:

mode a) user wishes to intervene

mode b) system wishes user to intervene

In addition, there is a requirement for visualisation of some aspects of the plan (via the Plan-World Viewers). This may be at the request directly of the planner user (i.e., as in (a) above but where no changes are to be made to a plan state) or maybe to serve a request from outside the agent (for example, to provide a visualisation of the plan at the request of the Task Assignment agent). So, we have a third mode of planner user support requirement for this latter service case:

mode c) planner user interface services to other agents

Earlier O-Plan1 systems (1984-1988) utilised a single KS-USER knowledge source for modes (a) and (b). The KS-USER knowledge source implemented in O-Plan2 up to version 2.1 is used for mode (c). In O-Plan2 up to version 2.1, some other user interface aspects related to mode (b) are incorporated in individual knowledge sources (such as KS-BIND). However, these were intended to be centralised in KS-USER in due course. Also, some aspects of support for mode (a) have been available via the system developer interface (the Data Base Manager Developer's menu and especially its break-in option). We now wish to demonstrate in an integrated way the proper support for mixed initiative planning within O-Plan2.

The aim will be to demonstrate the range of ways in which a planner user can interact with the system. These will show the mixed initiative properties of the O-Plan2 architecture in a realistic setting.

KS-USER Specification

KS-USER may be called in any one of three modes (indicated by an entry in the information field of the agenda entry passed to KS-USER).

mode a) User Request Mode

A button on each O-Plan2 agent control panel will allow the principal user of that agent to request interaction in their role as agent user (e.g., planner user role for the planner agent). An agent level agenda entry will be posted with USER REQUEST MODE indicated. This will lead to the activation of the KS-USER knowledge source installed in the agent.

At this level a menu of possible interaction options will be presented. The aim is to eventually provide very flexible editing of the current plan state and the ability to select from open alternatives (leaving those remaining to be handled by the controller), to re-order options available for schema choice, variable binding choice, ordering choice, etc. The immediate target is to provide support for the following:

1. Plan View
2. World View
3. Bind Variables
4. Break-in (with warning not to alter plan state improperly)
5. Quit

Bind Variables would be the only "sophisticated" part of the interface not currently available in KS-USER. This would find all open Plan State Variables (PSVs), and present these in a simple way with their current possible values and their restriction set. Perhaps some list of where the variables occurred in plan entities could also be given.

The interface would allow an user to:

1. select any open variable and to order the possible values
2. restrict any open variable (to one value or to some sub-set of values) (an alternative would be posted via the controller for the excluded choices to guarantee search space integrity).
3. commit valid changes made and quit from KS-USER
4. abort changes made and start again
5. quit from KS-USER

The choices would be made within a new "what-if" context layer such that the user could easily abort any sequence of decisions that was not useful.

It should be noted that sophisticated forms of user interface and compatible binding decision support could be possible in such an interface. We will only provide relatively simple forms in our implementation. One possible variant that would fit directly into the framework adopted would be the use of the VAD (Value-Assignment Delay) Heuristic and a supportive graphics interface for this as described in:

"Interactive Resource Allocation by Problem Decomposition and Temporal Abstractions", Berthe Y. Choueiry and Boi Faltings, AI Laboratory, Swiss Federal Institute of Technology, EPFL-Ecublens, CH-1015 Lausanne, Switzerland, Second European Workshop on Planning (EWSP-93), Vadstena, Sweden, IOS Press.

After any choice, the Plan State Variables (PSV) Manager would be allowed to propagate the consequences of the action taken, to check the immediately implied implications of the user action and to further constrain the remaining open variables.

mode b) System Request Mode

In O-Plan2, a KS-BIND agenda entry is posted to handle any outstanding PSV bindings. If the O-Plan2 control panel indicates that the user should be asked to make bindings for open variables, then when KS-BIND is activated it should delegate its job to a KS-USER agenda entry with a SYSTEM REQUEST MODE indicator for BINDING A VARIABLE and indicate the variable or variables involved. When activated, KS-USER will use the same interface as for Bind Variables under the USER REQUEST MODE described above. It may only allow the indicated variable(s) to be bound or may allow any variable that is still open to be bound (to be determined). If the planner user elects not to bind the variable(s) for which the system request was made, then a KS-BIND request with an automatic bind indicator should be posted to allow the proper termination of the knowledge source with responsibilities fulfilled.

mode c) Agent Services Mode

KS-USER may be called to service requests from outside (or possibly also inside) an O-Plan2 agent for user interface related access to the plan state via the PlanWorld Viewers. In this case the caller posts an agenda entry for KS-USER with the AGENT SERVICES MODE indicator and the specific service required. Currently we will support PLAN VIEW or WORLD VIEW from the Task Assignment agent.

APPENDIX B – User Roles in O-Plan2

User interaction with O-Plan2 can occur for a variety of purposes. Various *roles* of an user interacting with O-Plan2 are defined and are supported in different ways within the system. We consider the identification of the different roles to be an useful aid to guide future user interface support provision.

Domain Expert Role

A single user responsible for defining the bounds on the application area for which the system will act. The domain expert user may directly or indirectly specify O-Plan2 Task Formalism to define the domain information which the planner will use.

Domain Specialist Role

One or more domain specialists may define information at a more detailed level within the framework established by the domain expert. Once again, the domain specialist may directly or indirectly specify O-Plan2 Task Formalism to provide the detailed domain information which the planner will use.

Task Assignment User Role

The command user interacts only with the Task Assignment Agent to provide user requirements or commands. This is currently the top level menu for the O-Plan2 system. This user is responsible for the selection of the task which the system will try to carry out. The menu currently allows for a domain to be selected and for a selection from the task schemas within the Task Formalism for that domain to be selected. Future management of alternative plan options, plan analysis support and the provision of authority to plan or execute the plan are to be supported at this level.

Planner User Role

The planner user is the user responsible for ensuring that a suitable plan is generated to carry out the given task. This may involve the selection of alternatives, the restriction of options open to the planner and browsing on the emerging and final plan to ensure it meets the task requirements set by the task assignment user. Since the planner user can perform decision making in the planner agent, the planner user is supported by a knowledge source called KS-USER. This knowledge source can be added to the agenda for the current plan state on demand (via an user request). Since the KS-USER knowledge source normally has high priority, it will normally be called as soon as possible. The KS-USER knowledge source activation has access to the current plan state to allow for decisions on user intervention to depend on the contents of the current plan state.

Execution System Watch/Modify Role

The user may interact with the execution system to watch the state of execution of the plan and perhaps even to modify the behaviour of the execution system.

World Interventionist

If a world simulation is being used to demonstrate the O-Plan2 execution system, an user may be given facilities to intervene in the world simulation to cause events to happen and problems to occur such that execution of plans in uncertain situations can be tested.

User Support to Controller Role

The user may assist an O-Plan2 agent's controller to decide which knowledge source to dispatch to a waiting knowledge source platform or to decide on when to direct a running knowledge source to stop at a stage boundary.

User Support to Alternatives Handler

The user may assist an O-Plan2 agent's Alternatives Handler to decide which alternative to select when one is needed or to suggest an alternative is tried rather than continuing with the current plan state.

System Developer Role

The system developer has access to the diagnostic interface of the system running within each agent. This is supported by the Developer Diagnostic Interface of each O-Plan2 agent. The behaviour of this interface can be set and modified via a Control Panel which allows for the setting of levels of diagnostics using buttons, etc.

System Builder

The O-Plan2 Agent Architecture is intended to be sufficiently flexible to allow a system builder to create a system with defined behaviour. To this end, it is possible to have radically different plan state data structures, knowledge sources, domain information and controller strategies. For example, the O-Plan2 Architecture already has been used to provide a Manufacturing Scheduling System which uses a resource orientated representation for the plan state rather than the action orientated plan representation in the O-Plan2 Planner. This scheduler, called TOSCA (The Open SCheduling Architecture), also has different knowledge sources to those used in the O-Plan2 Planner.

APPENDIX C – O-Plan2 Planner Search Space Description

Characterisation of the Search Space

The O-Plan2 planner searches a space of teleologically novel solutions to the given task and only progressively expands the search space to include alternative means to satisfy the chosen teleological approach if a solution is not discovered earlier. In short we refer to this as a "teleologically novel, progressively extended search space".

The teleological approach defines the way in which conditions or resource requirements at activities within a plan are satisfied. O-Plan2 guarantees to produce at least one valid solution to a given problem if this is feasible within the constraints specified on the task and within the modelling capabilities provided by the constraint managers installed. It does this by systematically searching a lazy-generated space of solutions for the task given.

O-Plan2 does not guarantee to produce more than one such valid solution for any given teleological approach since it does not generate alternatives unless these prove necessary. It is therefore not suitable for problems in which all (syntactically different) solutions are required or in which an optimal solution is needed.

This approach to defining the search space of a planner was first introduced in INTERPLAN (Tate, 1974, 1975) and subsequently used in Nonlin (Tate, 1977) and O-Plan1 (Currie and Tate, 1991).

Search Space Node – A Plan State – A Conjunction of Constraints

A node of the search space is a partial plan (called a "plan state") which represents the conjunction of all constraints on the partial and fully elaborated plans which can be reached from that search space node without relaxing any constraint.

The systematicity of the search space is not compromised by selections of the order in which any component of the conjunct of constraints is refined. So simple or more complex opportunistic heuristics to select which constraint to refine are possible. Parallel constraint satisfaction techniques can also be utilised because of this property of a search node. The plan state structures themselves are designed to allow a large number of alternative solutions which do not affect other constraints to be built up (by "posting" a complete disjunction of alternative constraints into suitable structures within the plan state).

Search Space Arc – Plan Modification Operators

An arc of the search space represents the application of a Plan Modification Operator (PMO). An O-Plan2 Knowledge Source can branch the search and apply a single PMO in the new branch.

Relaxing the Constraints at a Node – Progressive Expansion (opening up) of the Search Space

On the search space leaves (fringe), if a PMO establishes that no further solutions are possible within the constraint set given using the current means of satisfying the teleological approach, then the PMO may "poison" that plan state (search node).

It is possible, at that time to consider whether alternative means to get to a valid solution within the defined teleological approach is possible. This is done by a special "poison handler" PMO. The poison handler contains all knowledge which the system has about alternative means to handle a given "poison cause". Only the poison handler has the authority to "extend" a poisoned plan state (search space leaf) and may do so progressively (only guaranteeing to use one of any alternate means it has at its disposal). Any relaxation of the search space whether done by the planner role user or the system should be delegated to the poison handler to allow it to maintain the integrity of the search space of the planner.

Mechanisms to Describe Disjunction in the Search Space

The designer of the O-Plan2 planner may choose from a number of mechanisms for holding disjunctions in the search space when seeking to implement the above search space definition. This can cut down on the search space that needs to be manipulated.

- Alternative Plan States.
- "Posting" Choices into separate Agenda Records (in their Information Fields).
- "Posting" Choices of the O-Plan2 Constraint Management Shared Ontological Elements in an "Or-tree" held in each separate Agenda Record (in their Information Fields).
- Progressive Expansion of Search via Poison Handler Capabilities.

APPENDIX D – Previous Work of Relevance to Mixed Initiative Planning – PLANIT

The paper "PLANIT Design Rationale and Future Directions" by Mark Drummond and Austin Tate (available as AIAI-TR-108 from AIAI, University of Edinburgh) reports on work conducted for a consortium of 27 organisations within the UK Alvey Programme in 1986-7. The PLANIT project produced an Interactive Planner's Assistant (IPA) that helped an user make use of an integrated set of knowledge rich plans, schedules and process plans for work in an enterprise.

Quotes from this paper:

> The PLANIT IPA (Interactive Planner's Assistant) can be considered as a "speadsheet" which provides a constraint network linking the various entities involved in representing a "knowledge rich" plan. This model is an useful one and with more powerful representations and operational planning capabilities the systems of the future will be based on a similar notion.
>
> Changes to any part [of a plan state] will be reflected in other constrained parts by the use of suitable constraint propagation systems.

The Plan representation and techniques used within the PLANIT IPA were based on O-Plan research and have much in common with the design principles of O-Plan2. The IPA provided a supportive interface through which a planner role user could make legitimate constrained changes to a plan and could seek the support of the system via the automatic application of Plan Modification Operators in a single step fashion. The system could then propagate the consequences of the user's chosen higher level plan modifications to more detailed constraints within the plan. This allowed the plan to be critiqued following the user driven changes.

APPENDIX E – Discussion on the C-5a Example of Mixed Initiative Planning from ARPI MIP Email

Description of the Problem – 5-Dec-93

From Drew McDermott Report on ARPI MIP Group Meeting on 1 to 2-Dec-93 at Yale.

Suppose the user has nursed a plan along, and in the final, scheduled result, C-5a transport planes are used in the early stages to transport some mundane item (K-rations, for instance). These planes are the biggest in the fleet, and would normally be reserved for transporting large, odd-shaped items. The user likes the plan the way it is, except for the use of C-5a's at this point (possibly because projected maintenance will cause a bottleneck in tank deliveries later). He or she would like to tell the system to redo the schedule, this time using alternative means to transport the K-rations in the early stages. How does that get conveyed to the computer, and how does the computer take it into account? The user would like to be able to point at the C-5a icon and drag it to some other point in the schedule; or, perhaps less vividly, to bring up a template specifying transportation resources, and cross out "C-5a" as an option at that point in the schedule.

The problem is that most human-initiated changes will leave the plan in an inconsistent or incomplete state. If, for instance, the user moves the C-5a's to a different part of the plan, the part of the plan that formerly made use of them now fails to deliver the K-rations, and the part that now uses them may call for more C-5a's than are actually available. To repair flaws like this, the user would like to rerun the modules that are responsible for worrying about incompletenesses and inconsistencies. The problem is to make sure that the modules don't just make the same mistake again.

We assume that the system doesn't simply lack information about the purpose of C-5a's. It is the job of the system *customizer* to make sure that when schedules are generated and ranked, a penalty is assessed for using C-5a's to carry K-rations. If this information is omitted, then there is no chance for the user to add it; that would amount to reprogramming the system, which the average user has no time or stomach for. He or she just doesn't speak the language of objective functions. The system must take what the user indicates and translate into the appropriate internal representation. In the case at hand, that will require it to realize that when the user edits C-5a's out of a part of the plan, the intent is to increase the penalty for using them to carry out the task of transporting K-rations. With such an interpretation, it is quite possible that after a scheduler is rerun, the C-5a's will still be used to carry the K-rations, because the new penalty is not enough to offset other penalties that the user can't see. This outcome will surely be frustrating to the user. On the other hand, if the user's edit is translated into a hard constraint on the schedule, a suboptimal schedule may result, without either party realizing it.

Comments by Austin Tate – 6-Dec-93

Relating to the C-5a example in the above discussion, it seems to be reasonable to allow for systems which know about preferences or priorities – such as preferring to use C-5as for large items – and which can themselves try to get good feasible plans within some constraint limits.

But, if that was not in the system's representations, the system need not give up. If the user did not like the system planner's proposed use of C-5as, then we could allow the planner role user (via the KS-USER knowledge source wrapper) to place constraints on the plan in order to see their consequences. Imagine the user doing the following:

- explore a new option

  in this option, restrict the resources for carrying K-rations for some part of the plan to exclude instances of plane type C-5a

- allow other options to be considered automatically if this option is infeasible

The O-Plan2 Knowledge Source Framework and Protocol is intended to allow ALL Knowledge Sources, whether system ones or a KS-USER wrapper, to be defined such that plan changes, constraints and preferences can be stated and explored. It also provides a simple high level way of splitting the search space where this proves necessary for exploring major options under consideration by the user.

It is more problematic to allow the use of C-5as to be crossed out in one particular plan option and have that eventually lead to the system adjusting preferences and priorities. If this was what was wanted, the user should think of this as crossing out the C-5a use for this SPECIFIC plan option, but also be given a supportive interface to easily be able to say that C-5as are not so good for this purpose for some reason (s). However, the learning community are making much progress, so adaptive interfaces that learn by following examples of behaviour an user wants will be possible at some point.