Technical Report

O-Plan Mixed Initiative Planning Capabilities and Protocols

Brian Drabble & Austin Tate

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

July 25, 1996

ARPA-RL/O-Plan/TR/24 Version 1

Contents

1 Introduction

The aim of this report is to describe the facilities within the current O-Plan system to provide a framework within which we can experiment with issues concerning Mixed Initiative Planning (MIP). The current framework has a number of simple mixed initiative system/user interfaces and a demonstration has been constructed which demonstrates some of them. The demonstration has also allowed a number of different modalities to be identified which would allow a richer level of interaction between different user roles and the system. The aim is now to increase the functionality of the system to incorporate all of the modalities identified. An overview of the MIP framework and the differing roles of the user is described in Figure 3.

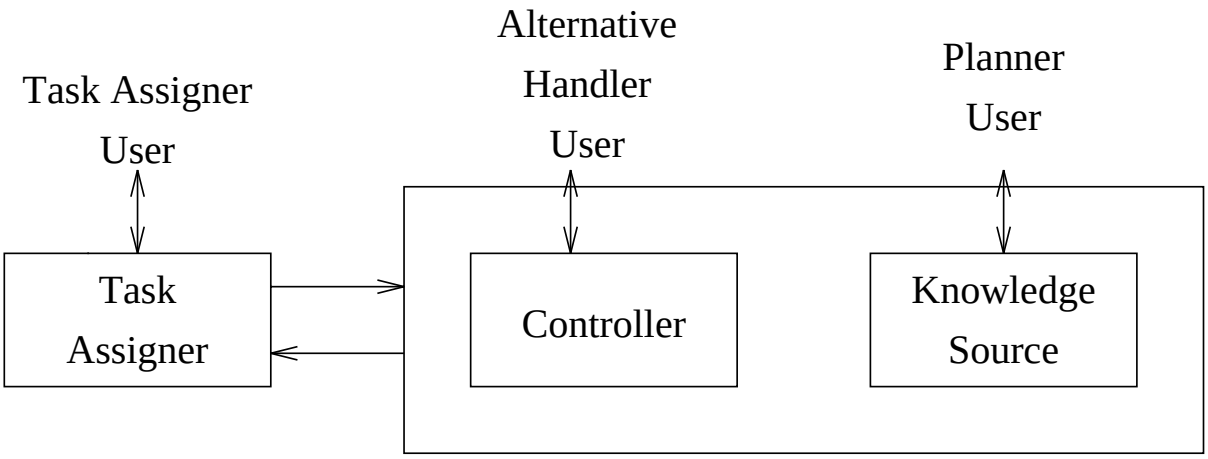

Figure 1: MIP Overview and User Roles

The structure of the report is as follows, Section 2 describes the KS-USER knowledge source which is used to provide a wrapper for the user to interact with the planner in the role of planner user. Section 3 describes the current MIP demonstration and Section 4 describes the different modalities required to support a rich MIP framework between the user and the system. Section 5 describes the problem of maintaining a correct user perspective and Section 6 describes the potential support which could be provided. The Appendix describes the different roles the user can undertake and the information and support they require.

2 The KS-USER Knowledge Source

The O-Plan 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-Plan Design Rationale for KS-USER Knowledge Source

The KS-USER knowledge source is intended to be the single point of interaction with the O-Plan 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-Plan agent (i.e., has the same properties as a knowledge source).

For integrity of the manipulation of an O-Plan agent's plan state, the KS-USER knowledge source must respect the O-Plan 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-Plan 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-Plan 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-Plan up to version 2.1 is used for mode (c). In O-Plan 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-Plan.

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-Plan 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-Plan 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-Plan, a KS-USER agenda entry is posted to handle any outstanding PSV bindings. If the O-Plan 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-Plan 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.

3 Mixed Initiative Planning Demonstration

The aim of this section is describe the MIP demonstration conducted with the current O-Plan system (version 2.3). The demonstration is based in the Pacifica Non-Combatant Evacuation Operation NEO domain and is characterised as follows:

- The scenario is that a number of persons need to be evacuated from the island of Pacifica. The persons are dispersed around the island and need to be picked up from the three main outlying cities of Abyss, Barnacle and Calypso and transported to the capital city Delta.. The evacuation of these cities can be carried out using ground transports (trucks, buses) and air-transports (helicopters). The ground transports (GTs) and the helicopters are initially located in Honolulu and need to be transported to Pacifica by means of military cargo planes (C5 and C141). The ground transports and the helicopters need to be returned to Honolulu at the end of the mission. The evacuees are transported to Honolulu by a B707 passenger plane which is at Delta airport at the start of the mission.
- There is a limited amount of fuel on the island and that is the only fuel available. The fuel reserves are as follows:
  - Diesel Fuel : 50 Gallons
  - Aviation Fuel: 200 Gallons
- The GTs use 20 gallons of diesel fuel per round trip i.e. from Delta to an outlying city and back and the helicopters use 30 gallons of aviation fuel.

The aim of the crisis planning is to evacuate the people from the outlying cities to Delta and then fly them back to Honolulu. The planner is able to find a solution to this problem automatically in 1810 cycles. However, with user intervention a specific approach to the evacuation can be taken. The approach to be used is as follows:

- Evacuate Abyss with the user specifying the ground transport method. Then have the user specify the ground transport to undertake the mission which in the demonstration is GT1.
- Evacuate Barnacle with the user specifying the helicopter method but with the system choosing an appropriate helicopter.
- Evacuate Calypso with the user specifying the ground transport method. The system is left to choose an appropriate GT (is should choose GT2 as GT1 is in use).

The purpose of the demonstration is to show how:

- the user via the KS-USER knowledge source can provide extra domain information to the planner

- the user can interact with the planner to chose the type of evacuation operation to be used, e.g. GT or helicopter and the specific resource e.g GT1, GT2, etc which will be used.
- a mixed initiative approach to the task allows it to be solved in a considerably lower number of problem solving cycles (424 as opposed to 1810).

The following script describes the steps to be followed to repeat the MIP demonstration and provides details of the interactions with the user and the choices the user was able to take. The script assumes that:

1. the O-Plan system has been started and initialised successfully
2. the user has selected the `Input TF` option from the `O-Plan Task Assignment` window and then selected `pacifica-mip` as the application domain from the list of TF files.

Full details of loading, initialising and tasking the O-Plan system can be found in the User Guide which forms part of the documentation of the O-Plan release.

During the demonstration information will be presented in a number of the windows of the O-Plan system and the user is advised to familarise themselves with the windows displayed.

- Before selecting the task make sure all user interaction options (schemas, variables and poison) are set to `ASK:`. This informs the planner that the user wishes to select schemas and variable bindings (when choices exist) and to choose the alternative plan state when the current one is poisoned, i.e. it is no longer valid.

  This can be achieved by clicking the `Intervene as User` button in the O-Plan Control Panel. The `User Intervention` menu will be presented and the user should choose the `Set modes` option. The `Set Modes` menu will be presented and the user should click on each of the modes:

  - `Set binding mode to:`
  - `Set schema selection mode to:`
  - `Set poison handler mode to:`

  In each case the option should change the mode from `ask` to `auto`. When all three modes read `auto` the modes are configured correctly and the user may choose the `OK` option. On exit from the `Set Modes` menu the user should then select `QUIT` from the `User Intervention` menu.

- Select `Set Task` from the `O-Plan Task Assignment` menu and then select `task_operation_columbus` from the task menu. The planner will begin its search for a plan. The first choice the planner must deal with is the evacuation method to move the people from Abyss to Delta. The planner prompts the user in two ways:

  - Details of the choice, the agenda it arose in and the variables involved are displayed in the `Planner User` window. An example of the `Planner User` window is given in Figure 2.

```
Possible schemas for ae-7:
  (:expand node-6 (transport abyss delta)):

  ground_transport_evacuees
  expands (transport ?loc2 ?loc3)
  with country = pacifica; loc1 = :undef; loc2 = abyss; loc3 = delta;
       gt = :undef;

  air_transport_evacuees
  expands (transport ?loc2 ?loc3)
  with country = pacifica; loc1 = :undef; loc2 = abyss; loc3 = delta;
       heli = :undef;
```

Figure 2: Example Planner User Window for schema selection

– The names of the alternative schemas are displayed in the schema choice menu.

At this point the demonstration aims to show two types on interaction with the user, choice of schema and choice of variable binding. Before choosing which schema is to be used for the Abyss evacuation the user should click the `Intervene as User` option in the O-Plan Control Panel. This will allow the user to inform the planner of the transport asset to be used to evacuate Abyss. The chosen method for evacuating Abyss is via GTs and the user should choose `ground_transport_evacuees` from the menu.

- At this point in the demonstration the system will prompt the user in response to the request initiated in the previous step. The user will be presented with the `User intervention` menu and should select the `Bind Variables` option and `Describe Open Variables` from the subsequent menu. The system will display the open variables in the `Planner User` window and the output is displayed in Figure 3.

  The user should now choose `Select a Variable to Bind` from the `Binding Options Menu`. The variable to be bound is the GT to be used to transport the evacuees from Abyss to Delta. The variable to be bound is PSV-6 and it should selected from the `Pick a Variable` menu. The current constraints for PSV-6 will be displayed in the `Planner User` Window and an example is shown in Figure 4. This states that:

  1. the variable has type `ground_transport`
  2. it was generated from the `ground_transport_evacuees` schema
  3. it is not constrained to be different from one or more other PSVs (`Not sames = nil`).
  4. it has two possible values `gt1 gt2`.

  The user should type GT1 in the `Planner User` window and then choose `Commit to Changes Made` in the `Binding Options` menu. The user may wish to explore the other

```
Open Variables:

psv-1: Names = (use1), Type = transport_use, Value = :undef,
       From schemas: ("transport_helicopters"),
       Restrictions = ?{actand ?{plan-state-vars::act-type transport_use} ?{non
?{plan-state-vars::act-type city}}},
       Not sames = nil, Possibles cache = (in_transit available);

psv-2: Names = (use2), Type = transport_use, Value = :undef,
       From schemas: ("transport_helicopters"),
       Restrictions = ?{actand ?{plan-state-vars::act-type transport_use} ?{non
?{plan-state-vars::act-type city}}},
       Not sames = nil, Possibles cache = (in_transit available);

psv-3: Names = (use1), Type = transport_use, Value = :undef,
       From schemas: ("transport_ground_transports"),
       Restrictions = ?{actand ?{plan-state-vars::act-type transport_use} ?{non
?{plan-state-vars::act-type city}}},
       Not sames = nil, Possibles cache = (in_transit available);

psv-4: Names = (use2), Type = transport_use, Value = :undef,
       From schemas: ("transport_ground_transports"),
       Restrictions = ?{actand ?{plan-state-vars::act-type transport_use} ?{non
?{plan-state-vars::act-type city}}},
       Not sames = nil, Possibles cache = (in_transit available);

psv-5: Names = (loc1), Type = air_base, Value = :undef,
       From schemas: ("ground_transport_evacuees"),
       Restrictions = ?{plan-state-vars::act-type air_base}, Not sames = nil,
       Possibles cache = (delta honolulu);

psv-6: Names = (gt), Type = ground_transport, Value = :undef,
       From schemas: ("ground_transport_evacuees"),
       Restrictions = ?{plan-state-vars::act-type ground_transport},
       Not sames = nil, Possibles cache = (gt1 gt2);
```

Figure 3: Example of the open Plan Variables

```
Selected variable:

psv-6: Names = (gt), Type = ground_transport, Value = :undef,
       From schemas: ("ground_transport_evacuees"),
       Restrictions = ?{plan-state-vars::act-type ground_transport},
       Not sames = nil, Possibles cache = (gt1 gt2);

Values:
```

Figure 4: Example of the Constraints for a PSV

options in this menu but should not bind any additional PSVs at this point. The user should not select `QUIT` from the `User intervention` menu.

- At this point in the demonstration the user has selected the evacuation method for Abyss and selected the GT which will pick up the evacuees. The system will now consider the evacuation from Barnacle and the same choice of evacuation method exists for Barnacle and existed for Abyss, i.e. to use ground transports or helicopters. The evacuation should be carried out using helicopters and the user should choose `air_transport_evacuees` from the `Pick the best schema menu`. The system will be allowed to select the appropriate helicopter for this evacuation and there is no need for the user to intervene.

- At this point the system will again prompt the user to select the evacuation method to evacuate Calypso. The method to be chosen is to use ground transports and the user should choose `ground_transport_evacuees` from the `Pick the best schema` menu. Before choosing this option from the menu the user should queue a request to intervene in the planning process. The aim is to switch the selection mode for variable binding to `automatic` to allow the system to choose variable bindings for the evacuation from Calypso [1].

- The system will use ground transports to evacuate Calpyso and will then prompt the user with the `User intervention` menu. The user should choose the `Set modes` option and then click the `Set binding mode` option. This will change the indicator to `ask` and the user should then choose `OK` to quit the menu and `QUIT` to exit the `User intervention` menu.

- At this point in the planning process evacuation methods for Abyss, Barnacle and Calypso have been chosen together with the required GT to evacuate Abyss. The system will now attempt to satisfy a number of conditions introduced by the different mission schemas. Most of these can be handled by the system but two require the user to provide the system with support in handling these conditions. The user will be prompted with a menu which indicates the condition which could not be satisfied and that the system has poisoned the

[1] The system should choose GT2 as GT1 is busy. i.e. the user selected GT1 to evacuate Abyss

plan state. The conditions involve the selection of an appropriate ground transport and having the ground transport in the correct place at the start of the mission.

In this simple demonstration the planner is unable to handle the poison by editing the current plan state and is forced to choose an alternative plan state. The user can either choose the alternative plan state or can leave the choice to the planner. In this demonstration the option will be to have the planner handle the choice and the user informs the planner of this by selecting the `Handle and continue` option from the menu. The system will prompt the user with the second failed condition and again the user should choose the `Handle and continue` option.

- The planner will now continue and satisfy the remaining conditions and bind the unbound variables. Once this is complete the plan will be finished and this is indicated by the message `planner finished` appearing on the `status` line of the Task Assignment Window.

4 Mixed Initiative Capabilities and Modalities

The MIP demonstration provides the user with a number of different ways or modalities of interacting with the planner. The following modalities and examples of the use of the modalities have been identified by using the crisis-response planning application in the Pacifica domain as a guide. The O-Plan Task Formalism domain description `pacifica-mip.tf` using task `Operation_Columbus` forms the basis for the Mixed Initiative Planning (MIP) demonstration of O-Plan – the **P-1** deliverable for the O-Plan project. Support within O-Plan version 2.3 (13-July-95) for demonstration of mixed initiative planning interactions between the user and the planning system helped with the discussions.

The modalities are related to one user role related to the task assignment agent:

- Task Assignment User Role

Two user roles related to the planner agent:

- Alternatives Handler User Role
- Planner User Role

and two components of the O-Plan planner agent:

- controller (alternatives handler part)
- knowledge source platform (knowledge sources)

The modalities and examples below are marked with a "*" if they can be demonstrated in the P-1 MIP demonstration.

- **Mode 1**:
  Knowledge Source asks for Planner Role User Involvement.
  Classification: Search space unaltered, order of choices altered.
  - * example 1: KS-BIND for object selection.
  - * example 2: KS-EXPAND and KS-ACHIEVE for action/method schema selection.
- **Mode 2**:
  Controller asks for Alternatives Handler Role User Involvement.
  Classification: Search space unaltered, order of choices altered.
  - example 1: Controller for alternatives selection preference

- **Mode 3**:
  Alternatives Handler Role User asks for User Involvement (in Controller).

  Classification: Search space unaltered, order of choices altered.

  - example 1: alternatives selection preference

- **Mode 4**:
  Alternatives Handler Role User prunes search space (in Controller).

  Classification: Search space altered, order of choices unaltered.

  - example 1: removing alternatives.

- **Mode 5**:
  Planner Role User prunes search space (in Knowledge Sources)

  Classification: Search space altered, order of choices unaltered.

  - * example 1: object selection
  - * example 2: action/method schema selection

- **Mode 6**:
  Planner Role User asks to make Choice (via KS-USER Knowledge Source)

  Classification: Search space unaltered, order of choices altered.

  - * example 1: object selection
  - * example 2: action/method schema selection

- **Mode 7**:
  Planner Role User asks for Plan Information (via KS-USER Knowledge Source)

  Classification: Search space unaltered, order of choices unaltered.

  - * example 1: Plan View (of current plan state)
  - * example 2: World View (of current plan state)
  - * example 3: Break-in for flexible plan browsing of current plan state - read only, altering data structures in a break-in implies a developer user role activity)

- **Mode 8**:
  Alternative Handler Involvement User asks for Alternatives Information (in Controller)

  Classification: Search space unaltered, order of choices unaltered.

  - * example 1: Break-in for alternatives browsing (can be shown via Agenda Manager single step provided for Developer Role User)

- **Mode 9**:
  Task Assignment asks for Plan Information (via task assignment agent to planner agent event)

  Classification: Search space unaltered, order of choices unaltered.

  - * example 1: Plan View (of current plan state)
  - * example 2: World View (of current plan state)

- **Mode 10**:
  Planner Role User specifies user/system interaction modes

  Classification: Search space unaltered, order of choices unaltered.

  - * example 1: set requirements for user or system to make object selections.
  - * example 2: set requirements for user or system to make action/method schema selections.

- **Mode 11**:
  Alternative Handler Involvement User specifies user/system interaction modes

  Classification: Search space unaltered, order of choices unaltered.

  - * example 1: set requirements for user or system to make alternative selections

5 Maintaining User Perspectives and Keeping on Track During MIP

P-1 demonstration and related studies have shown various modes of possible interaction between users playing various roles in the planning process and various system agents or components in a planner. But the current ways in which users can interact with planners are quite awkward and it is very easy for a user to lose their approach or direction within the interactions. Even when a user is clear in what they are trying to achieve, it is difficult to actually "steer" the planner in the way a user might want. It is therefore very much more difficult to interact with the system when the user is exploring alternatives and refining their own approach.

A particularly difficult thing within the current implementation of O-Plan is to coordinate the alternative plans which are being explored by the system with the plans which the user is seeking to explore. The system may use a number of alternatives in "generate and test" mode to find plans. If the alternative alters while the users are themselves dealing with one alternative or exploring a range of alternatives this becomes difficult to keep track of for the user.

The same problem was noted very early on with mixed initiative planning in the Nonlin+ planner (Tate and Whiter, 1984) which was applied to the problem of replenishment of naval vessels at sea (RAS). In that case, a user interacted witha system to choose specific strateies for sending groups of ships to safely replenish fleets while maintaining defiensive cover for the supply ships as well as the fleets. Once a user had begun to select specific ships, this was normally against an "approach" or "tactic" which the user wanted to follow through. Later decisions were dependent on earlier ones and made within the context of the chosen approach. If the system moved to an alternative plan state during the user interaction processes, the user quickly became lost.

A "jotter" for the user was introduced to Nonlin+ which allowed notes to be added by the user in a strecutured form. The contents of the jotter were associated (in a context-dependent fashion) with a specific alternative plan state. If the system altered the current plan state, then the contents of the jotter ware also reset to the state they were in the last time the user made entries in the jotter. This allowed the user to make a note of the strategy being followed in a given plan alternative, and make notes about how far they had gone in carrying out their strategy in that particular state. In this way, the cooperative user/system search for a solution could be characterised as the concurrent exploration of a number of alternative "threads" or alternative plans. Unfortunately, the user had little say over when the system chose to switch alternatives. The user interaction was therefore fragmented and not focussed on the alternatives which they viewed as most relevant or promising.

O-Plan also has a jotter-like capability which is context dependent to the plan state. In O-Plan this is called the "notepad" and entries on it are in a structured form called "notes".

As in our earlier experience with RAS on Nonlin+, in the O-Plan Pacifica P-1 Mixed Initiative Planning Demonstration, the user begins to interact with the system to "steer" it according to a defined tactic oe approach to the problem. They are choosing to evacuate the three regions in the scenario using a mix of ground and air transports which will reduce contention and allow

a rapid (though more costly than the cheapest option) return of the people to the base at Delta for air transport out from Pacifica. Once the user is following this tactic, it is technically awkward in the current (version 2.3) O-Plan planner for the user to maintain a perspective of which alternative plan state they are working in at any moment, and what emans can be taken to progress their chosen approach. And this is when the user is following a clear tactic. Its much more difficult when the user is also exploring options and deciding on the tactic in the first place.

The current O-Plan planner (version 2.3) providesa range of quite technical features to demonstrate a range of mixed initiative modes but these are scattered throughout the user interfaces and interaction facilities of the planner. Crucially, there is no means for the user to specify that the system should work with (or present to the user perspective at least) a particular plan state. The current implementation of the O-Plan planner (version 2.3) does not differentiate between:

1. "important" user orientated plan state "options",
2. "alternative" plan states generated during problem solving which represent the main decision points about alternative ways to sole the problem,
3. simple search space "contexts" used in a "generate and test" fashion by the planner to seek solutions to a localised problem.

Recent discussions about the O-Plan design have been trying to more clearly chararacterise these different ways to reference and explore alterative plan states within the O-Plan planner and in conjunctionm with users playing various roles in the planing process.

6 Identification of New MIP-related Requirements for O-Plan

Lessons learned from the Pacifica P-1 demonstration and consideration of MIP modes and styles of interaction have been joined with requirements emerging from the planned Technology Integration Experiments (TIEs) between ISX, Rochester and AIAI which concern mixed initiative planning and links between a "task assignment agent" and a "planning agent". This has helped us identify the following important building blocks for more flexible mixed initiative support in O-Plan in future:

1. Clearly differentiate user perspective "options" from internally generated and manipulated alternative plan states.
2. Provide means for the user and system to coordinate their perspective on which option is being developed at any one time (from the user perspective at least).
3. Allow a user "tactic" or "approach" to be developed followed within nominated options.
4. Acknowledge that once a tactic or approach is being followed, that aids are necessary to support the user in enacting or adjusting their chosen tactic or approach within an option. This will include "compound" dependent activities or decisions.

Appendix

User interaction with O-Plan can occur for a variety of purposes. Various *roles* of an user interacting with O-Plan 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-Plan 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-Plan 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-Plan 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-Plan 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-Plan 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-Plan 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-Plan 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-Plan 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-Plan 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-Plan Planner. This scheduler, called TOSCA (The Open SCheduling Architecture), also has different knowledge sources to those used in the O-Plan Planner.