O-Plan

Demonstration Guide

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

July 13, 1995

Version 2.3

Acknowledgements

The O-Plan project began in 1984. Since that time the following people have participated: Colin Bell, Ken Currie, Jeff Dalton, Roberto Desimone, Brian Drabble, Mark Drummond, Anja Haman, Ken Johnson, Richard Kirby, Glen Reece, Arthur Seaton, Judith Secker, Austin Tate and Richard Tobin.

Prior to 1984, work on Interplan (1972–4) and Nonlin (1975–6) was funded by the UK Science and Engineering Research Council and provided technical input to the design of O-Plan.

From 1984 to 1988, the O-Plan project was funded by the UK Science and Engineering Research Council on grant numbers GR/C/59178 and GR/D/58987 (UK Alvey Programme project number IKBS/151). The work was also supported by a fellowship from SD-Scicon for Austin Tate from 1984 to 1985.

From 1989 to 1992, the O-Plan project was supported by the US Air Force Rome Laboratory through the Air Force Office of Scientific Research (AFOSR) and their European Office of Aerospace Research and Development by contract number F49620-89-C-0081 (EOARD/88-0044) monitored by Northrup Fowler III at the USAF Rome Laboratory.

From 1992 to 1995, the O-Plan project was supported by the ARPA/Rome Laboratory Knowledge Based Planning and Scheduling Initiative through the US Air Force Rome Laboratory through the Air Force Office of Scientific Research (AFOSR) and their European Office of Aerospace Research and Development by contract number F49620-92-C-0042 (EOARD/92-0001) monitored by Northrup Fowler III at the USAF Rome Laboratory.

Additional resources for the O-Plan and O-Plan projects have been provided by the Artificial Intelligence Applications Institute through the EUROPA (Edinburgh University Research on Planning Architectures) institute development project.

From 1989 to 1993, research on scheduling applications of the O-Plan architecture was funded by Hitachi Europe Ltd. From 1989 to 1992, the UK Science and Engineering Research Council (grant number GR/F36545 – UK Information Engineering Directorate project number IED 4/1/1320) funded a collaborative project with ICL, Imperial College and other partners in which the O-Plan architecture was used to guide the design and development of a planner with a flexible temporal logic representation of the plan state. A number of other research and development contracts placed with AIAI have led to research progress on the O-Plan prototype.

Contact Information

The O-Plan project team can be contacted as follows:

O-Plan Team
Artificial Intelligence Applications Institute
The University of Edinburgh
80, South Bridge
Edinburgh EH1 1HN
United Kingdom

Tel: (+44) 131 650 2732
Fax: (+44) 131 650 6513
Email: oplan@ed.ac.uk

Created: November 29, 1991 by Brian Drabble
Last Modified: July 13, 1995 (10:06) by Brian Drabble
Printed: July 13, 1995

Contents

1 Introduction

This document describes a set of demonstration applications which form an introduction to the O-Plan system. The applications range in complexity from simple block stacking problems to space station construction problems. Each of the demonstrations shows a different aspect of the planning problem from simple action expansions to complex goal directed reasoning.

The demonstration examples are stored in a single directory referred to as TF. The demonstration TF directory is in $OPLANDIR`/demo/tf`.

The demonstrations themselves are as follows:

- Block stacking
  These are a set of TF files which deal with various block stacking problems such as the Sussman Anomaly.

- Blue Badges in Boxes
  Contains a TF file which describes a conditions satisfaction problem devised by Drew McDermott. There are two operators putin(container,object) and spray(container,colour) where a side effect of spraying a container colours the current contents of the container (if any). The task is to get a blue badge in a box and carry out the action of spraying the box red during the plan.

- House Building
  These are a set of TF files for house building problems. The different files deal with time constraints, alternative building methods and contractors, etc.

- Missionaries and Cannibals
  This is the standard missionaries and cannibals problem in which the task is to move three missionaries and three cannibals from the right bank of a river to the left bank via a single boat. The constraint is that the cannibals should *never* outnumber the missionaries on either bank (otherwise the cannibals will eat the missionary!).

- Pacifica
  Pacifica is an imaginary NEO (Non-combatant Evacuation Operations) Scenario developed by AIAI and provides a test domain for transportation logistics problems. Pacifica is an island state in the Pacific and the tasks consist of moving a number of evacuees from remote parts of the island to a central evacuation point and then flying them out. This example can be used to demonstrate the AutoCAD user interface to O-Plan.

- Satellite Control
  This contains a TF file for a satellite command and control problem. The satellite has various experiments whose data must be captured and transmitted to earth, taking into account time and resource constraints.

- Space Platform Construction
  This contains a TF file for a very simple space platform construction problem. This example can be used to demonstrate the AutoCAD user interface to O-Plan.

- Spanning Problem
  This contains a TF file for a simple problem involving condition satisfaction. The problem show that certain solutions will be missed by the planner if it chooses to achieve conditions from the start of the expansion which introduced them rather than from any point in the plan state.

Each of the following sections deals with a particular application domain. At the end of each section there is a table which describes:

1. the TF files containing descriptions of the chosen domain
2. the task schemas which specify the tasks which can be solved within the chosen domain
3. a set of comments concerning the points of interest within the demonstration.

2 Block Stacking

Block stacking puzzles are often used as a basic test domain for AI planning systems. In this particular demonstration the planner will be asked to solve the following type of problem:

```
                        -----
                       |     |
                       |  A  |
   -----               |_____|
  |     |              |     |
  |  C  |              |  B  |
  |_____|   -----      |_____|
  |     |  |     |     |     |
  |  A  |  |  B  |     |  C  |
  |_____|  |_____|     |_____|

   Initial State      Goal State
```

The planner is to construct a plan to change the initial state into the goal state. This is a simple problem because the planner has only a few simple plan operators which rely on the following assumptions:

1. the table on which the blocks sit can have as many blocks on top of it as desired
2. the hand which moves the blocks can only move one block at a time
3. a block can only be moved if it has a clear top i.e. no other block is on top of it
4. any block which is the destination of a move must have a clear top beforehand

A full list of file names and task schema names are provided in the following two tables. The file `blocks-1` contains a single `puton` schema with which the planner must construct a plan. The file `blocks-2` contains the schemas: `puton`, `makeon` and `makeclear` from which a plan must be constructed.

Filename	Task Schemas	Comments
blocks-1	task_stack_ABC	Refer to following table: Problem 1
	task_stack_ABC_2	Refer to following table: Problem 2
	task_stack_CBA	Refer to following table: Problem 3
	task_stack_BAC	Refer to following table: Problem 4
blocks-2	task_stack_ABC	Refer to following table: Problem 1
	task_stack_ABC_2	Refer to following table: Problem 2
	task_stack_CBA	Refer to following table: Problem 3
	task_stack_BAC	Refer to following table: Problem 4

Problem Number	Initial State	Goals
Problem 1	(on c a) (on a table) (on b table) (cleartop c) (cleartop b)	(on a b) (on b c)
Problem 2	(on a b) (on c table) (on b table) (cleartop c) (cleartop a)	(on a b) (on b c)
Problem 3	(on c a) (on a table) (on b table) (cleartop c) (cleartop b)	(on c b) (on b a)
Problem 4	(on c a) (on a table) (on b table) (cleartop c) (cleartop b)	(on b a) (on a c)

3 Blue Badges in Boxes

This application shows the development of a plan for a condition satisfaction problem devised by Drew McDermott. There are two operators `putin(container,object)` and `spray(container,colour)` where a side effect of spraying a container colours the current contents of the container (if any). The task is to get a blue badge in a box and carry out the action of spraying the box red during the plan.

A full list of file names and task schema names are provided in the following table.

Filename	Task Schemas	Comments
bbb	blue_badge_in_red_box	The initial conditions state that badge is blue and there is nothing in the box
	red_badge_in_red_box	The initial conditions again state that the badge is blue and there is nothing in the box

4 House Building

This application shows the development of a plan for the construction of a typical family house. The plan is developed in a top down manner with high level actions such as `install_services` and `decorate` being inserted first and then expanded only when necessary. This avoids the planner having to consider the detailed levels of the plan before the more important high level actions have been sketched out. The different house building tasks show how time and resources can be modelled within the domain. In the case of resources (house-5) the house can be built of bricks, sticks or straw but must keep within an overall budget specified as a number of pounds.

A full list of file names and task schema names are provided in the following table.

Filename	Task Schemas	Comments
house-1	task_build_house	A simple family house
house-1-inc-1	task_build_house_2	This provides an alternative `install_services` schema for the house building domain. The file requires **house-1.tf** to be preloaded beforehand.
house-2	task_build_house	Contains TF which shows how interactions occur between schemas and how schemas which provide alternate ways of achieving a task can be specified. The interaction can be seen in the KS-PLATFORM window as a failure to expand the first install services schema chosen and as a result the second install services schema is chosen.
house-3	task_build_house	A larger house building example that in **house-1.tf** or **house-2.tf**. The plan constructed requires more condition satisfaction and interaction removal than in the previous house building problems.

Filename	Task Schemas	Comments
house-4	task_build_house	A house building example similar to **house-1.tf** except that it includes durations for actions and time windows in which specified actions must take place.
	task_build_house_to_time_0	build house within a set time period on difficult ground, no alternatives possible
	task_build_house_to_time_1	build house within a set time period on ready ground, 1 alternative possible (with standard kitchen)
	task_build_house_to_time_2	build house within a set time period on ready ground, 2 alternatives possible (one with standard kitchen, one with luxury kitchen)

Filename	Task Schemas	Comments
house-5	task_build_house	A simple family house with no restrictions on time and resources. The resources available are money, sticks, straw and bricks.
	task_build_secure_house	Builds a family home for a cost between 0 and 2000 pounds which is also proof against wolves. (Guess who lives in this house?).
	task_build_cheap_house	Builds a home for a cost between 0 and 500 pounds.
	task_build_cheap_secure_house	Builds a home for a cost between 0 and 500 pounds which is also proof against wolves.

5 Missionaries and Cannibals

This application shows the development of a plan for the standard missionaries and cannibals problem. The aim of the demonstration is to show the numerical reasoning capability of the O-Plan system. The problem consists of moving 3 missionaries and 3 cannibals from the right bank of a river to the left bank via a single canoe. The constraint on the problem is that at any time the number of cannibals must not outnumber the number of missionaries. If this is the case then the cannibals will eat the missionaries!

The problem is represented in two ways:

1. The mathematics of calculating the number of missionaries and cannibals is carried out using successor arithmetic based on matching. The task definition enumerates all the the possible states of the problem both safe and unsafe (number missionaries < number cannibals) in order to prevent looping in the search for a solution.
2. The mathematics of calculating the number of missionaries and cannibals is carried out using `compute` functions which call predefined functions, e.g. add, subtract, etc.

A full list of file names and task schema names are provided in the following table.

Filename	Task Schemas	Comments
m-and-c-1.tf	mc_problem	The starting state of the problem is that the three missionaries and three cannibals are on the left back of the river along with the canoe.
m-and-c-2.tf	mc_problem	The starting state of the problem is that the three missionaries and three cannibals are on the left back of the river along with the canoe.

6 Pacifica

This application shows the development of a plan for Non Combatant Evacuation Operation (NEO) from a hypothetical island named Pacifica. The Pacifica Non-combatant Evacuation Operations Scenario (Pacifica NEO) is being used to demonstrate various concepts related to reactive execution of plans. Though this scenario is hypothetical the objectives and issues addressed are realistic. Like the scenario itself the data used is also hypothetical. However, it should be sufficiently realistic. Publicly available United States Transportation Command Operations (USTRANSCOM) documents (see [2, 3]) were used as guides to determine some of the factors used in this scenario.

Pacifica is an island state located in the Pacific Ocean within long distance flying time of Honolulu, Hawaii. It has a very interesting coastline, but remains shrouded in mystery due to its inaccessibility over the centuries. Only in the last century has it been inhabited though some areas of the island remain largely unexplored and are unmapped. In 1976, the United States opened diplomatic relations with the country by establishing an Embassy in the capital city of Delta. A map of Pacifica can be found in Figure 1.

Pacifica NEO Scenario

Recently, civil unrest has broken out in Pacifica. A number of US nationals need to be evacuated from selected points on the island by means of ground or air transports. Initially, both a C5 and a B707 along with the ground and air transports are located in Honolulu. From this initial situation a plan is developed which moves the required resources from Honolulu to Delta, Pacifica, then transports US Nationals via ground or air transports from their present locations to Delta as the Point-of-Embarkation (POE) for the evacuees and finally, airlifts all nationals to Honolulu. The recovery of aircraft and ground transports airlifted to Pacifica must also be completed at the end of the operation.

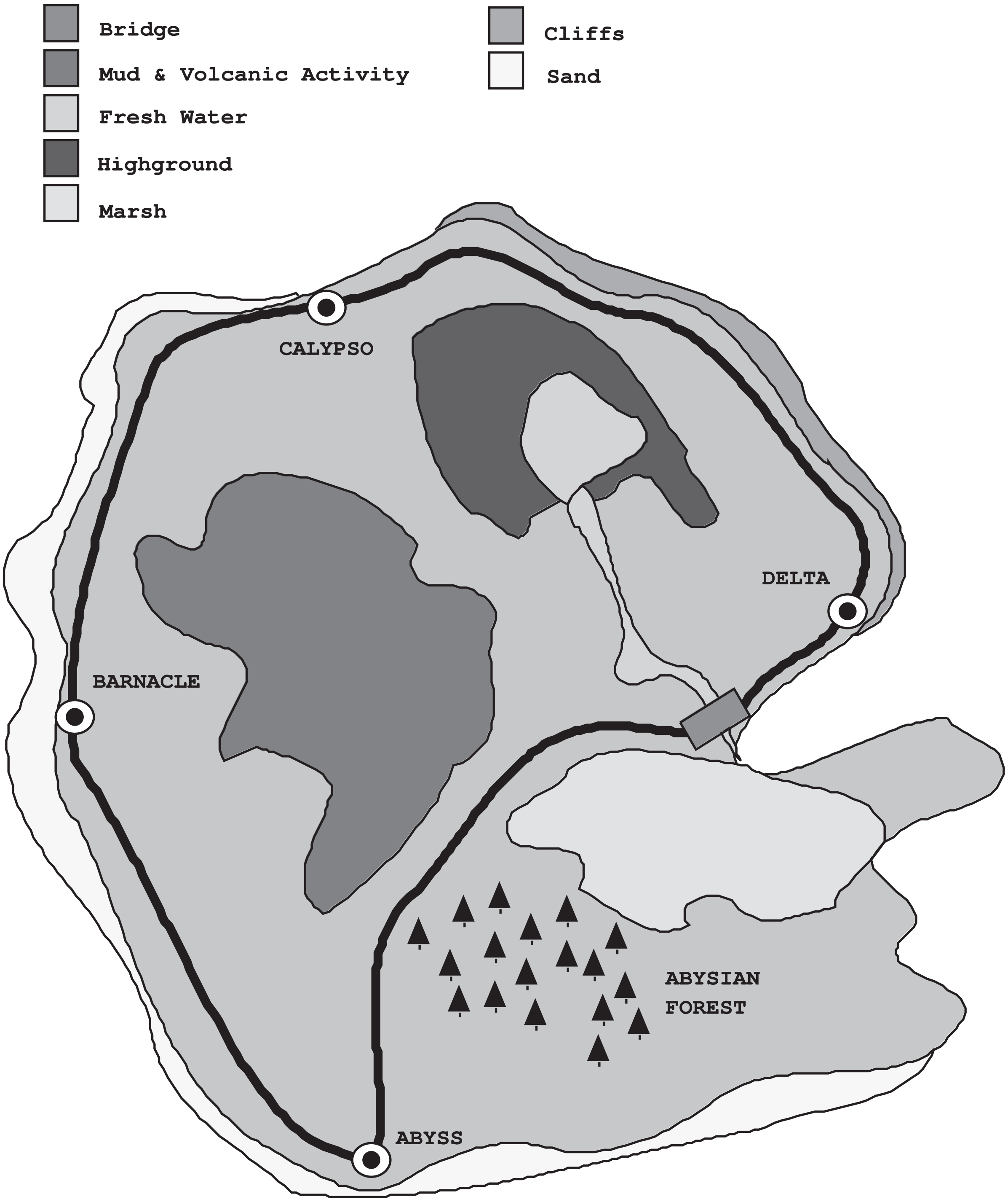

Figure 1: Island State of Pacifica

This section describes several factors which must be addressed in transportation logistics type problems and data which is used in the scenario. These factors are by no means a complete set of the factors which must be addressed in a real transportation logistics problem however, they are sufficient to demonstrate various concepts required for studying Command, Planning, and Control issues.

Aircraft

The Pacifica NEO utilises three types of aircraft resources. These are the C5 and C141 cargo transports and a B707 passenger jet. Their characteristics are described here.

Aircraft data used in the scenario is shown in the tables of this section. Table 1 describes passenger/cargo capacity, range, and landing requirements. "Range" data calculations include assumptions regarding the weight of reserve fuel, aircraft operating weight, the weight of fuel to an alternative destination, and possibly others. The "Runway Required" figures are for takeoff and indicate peacetime assumptions where applicable. These tables also reflect assumptions such as weather conditions, sea level, operating weight, and others.

Type	Passengers without cargo	Passengers with cargo	Range	Runway Required
C5	329	30	6238	9150
C141	275	15	5400	8200
B707	311	10	8427	6831

Table 1: Capacity, Range, and Landing Requirements

Type	Block Speed	Onload	Enroute	Offload
C5	6:30	1:45	1:15	1:30
C141	5:45	1:55	1:10	1:20
B707	5:00	0:30	1:00	0:45

Table 2: Speed and Turnaround Data

Table 2 describes speed and turnaround time data. Block speed is an estimate of the average speed of an aircraft that takes into consideration the amount of time it takes to take off, attain cruising altitude, descend at a destination, land, taxi and be parked in the "blocks" at the destination. Turnaround time consists of three separate times: onload time (the time to load the aircraft), enroute time (the time to refuel), and offload time (the time to unload the aircraft). It is assumed that the onload/offload times given are for fully loaded aircraft.

Ground Transportation

The scenario uses two instances of a single type of ground transport. Borrowing from the terminology from the aircraft types used in the scenario, the ground transport type data is shown below in Table 3.

Type	Onload	Enroute	Offload	Capacity	Range	Fuel Load
Ground Transport	0:20	0:15	0:20	50	348	30

Table 3: Ground Transport Data

Air Transportation

The scenarios use two instances of a single type of air transport, i.e. helicopters for in theatre movements. The planner has the option of using either ground transports or air transports depending on the fuel available and the time constraints imposed.

The type data for the air transports is shown below in Table 4.

Type	Onload	Enroute	Offload	Capacity	Range	Fuel Load
Ground Transport	0:20	0:30	0:20	50	500	1200

Table 4: Air Transport Data

Airport Characteristics

The ability of aircraft to land at different airports is determined by a number of factors, two of which are the length of the runway and the weight of the payload.

Travel Distances

The Table 5 gives distance and travel time data used in the scenario.

From	To	Transport Type	Distance (miles)	Travel Time
Delta	Abyss	GT	85	1:30
Delta	Barnacle	GT	160	2:15
Delta	Calypso	GT	60	4:45
Delta	Abyss	AT	45	0:50
Delta	Barnacle	AT	90	1:15
Delta	Calypso	AT	25	0:45
Delta	Honolulu	C5	2915	6:30
Delta	Honolulu	B707	2915	5:00

Table 5: Point–to–Point Travel Data

The following table describes the domains descriptions for Pacifica together with the different tasks which can be set.

Filename	Task Schemas	Comments
pacifica-1.tf	Operation_Blue_Lagoon	Transportation provision uses two Ground Transports
	Operation_Castaway	Transportation provision and removal from POE not ordered with respect to ground transportation arrangements.
	Operation_Paradise	Requires GT3 in type ground_transports to allow its proper use. However, it will work without this transport. The scenario uses three transports with one available already at Delta.
pacifica-2.tf	Operation_Columbus	Transport provision uses two Ground Transports and two helicopters. The transports require diesel fuel and aviation fuel respectively to travel between points on the island. The fuel available forces the planner to choose 2 Ground Transports and one helicopter.
pacifica-3.tf	Operation_Columbus	Transport provision uses two ground transports and one helicopter. The ground transports, air transports, cargo planes (C5, C141) and the passenger plane B707 require appropriate fuel either, diesel or aviation. The fuel is stored in tanks at Delta airport and further aviation fuel can be brought in via a KC-10 tanker aircraft. Other resources such as runways and parking areas are also modelled in this problem.
	Operation_Columbus_Mixed	This uses the same transport provision as Operation_Columbus put imposes tighter temporal constraints on the mission.
	Operation_Columbus_GTS_Only	The transport provision uses only two ground transports. There is sufficient time and diesel fuel to allow the mission to be carried out.
	Operation_Columbus_ATS_Only	The transport provision uses only two helicopters. There is sufficient time and aviation fuel to allow the mission to be carried out.
pacifica-4.tf	Operation_Columbus	Transport provision uses two ground transports and one helicopter. The ground transports are used for the evacuations and the helicopter is held in reserve should a problem occur with one or more of the ground transports. The file also contains a number of "repair plans" which can be used to repair the plan as a result of unexpected failures, e.g. blown tyres or a blown engine.

6.1 Resource Based Demonstration

Using the PRECiS domain as the base, the O-Plan second year demonstration showed the benefits of using a rich model of resources in an activity planner framework. A number of different resources types were modelled and these are as follows:

- **consumable-strictly**:
  Aviation and diesel fuel in the tanks at Delta and the evacuees at Abyss, Barnacle and Calypso.
- **consumable-producible-by-agent**:
  Aviation fuel brought by the KC-10 Tanker aircraft from Honolulu when there is insufficient fuel at Delta to refuel the B707 passenger aircraft prior to departure.
- **consumable-producible-by-outwith-agent**:
  This resource is not handled as expected due to O-Plan not having the required event handling ability.
- **resusable-nonsharable**:
  The ground transports, helicopters, C5, C141, KC-10 and B707.
- **resusable-sharable-independently**:
  The taxi ways at both Honolulu and Delta airbases.
- **reusable-sharable-synchronoulsy**:
  The runways at both Honolulu and Delta airbases

The demonstration was designed to show the ways in which a rich model of domain resources e.g. trucks, aeroplanes, fuel, runways, etc, could be encoded and used within an activity planner. The demonstration also provides a check on the system's ability to reason with numbers and numerical ranges, e.g. calculating the amount of fuel a transport (ground or helicopter) requires to undertake an assigned trip and to ensure the fuel is available.

The mission to be undertaken is to evacuate a number of people from the outlying cities of Abyss, Barnacle and Calypso back to the capital city Delta. The evacuation is achieved by means of a mixture of ground transports (GTs) and helicopters (ATs) each of which require diesel and aviation fuel respectively. The people are flown off the island via a B707 passenger plan parked at Delta airport. The B707 requires to be refueled prior to leaving Delta and fuel can be obtained either:

- locally from the aviation fuel tanks at Delta airport or
- from a KC-10 refuelling aircraft sent from Honolulu should local supplies prove to be insufficient. If the KC-10 aircraft is requested it must be returned to Honolulu after refuelling the B707.

The GTs and ATs are flown into Delta from Honolulu via C5 and C141 transport aircraft and must be returned to Honolulu after the evacuation is complete. The airport at Delta has a single operational runway and limited safe parking (due to terrorist activity) which requires the take off and landing of the various aircraft to be coordinated.

The scenario and the missions which can be undertaken are described in file `pacifica-3.tf`, detail of which are given in the previous table. Full details of the demonstration together with the results obtained can be found in [4].

6.2 Command, Planning and Control Demonstration

The PRECiS domain was used for the O-Plan third year demonstration which showed O-Plan solving a number of tasks from a command, planning and control scenario. The aims of the demonstration were to show:

- O-Plan reacting to changes in the environment and identifying those parts of the plan which were now threatened by these changes.
- O-Plan reacting to changes in the overall task by integrating new plan requirements into the plan.

In both these cases the changes were to be made to an ongoing and executing plan.

The types of changes explored in this demonstration include failures of trucks due to blown engines and tyre and the inclusion of new new objectives, e.g. pick up an extra group of evacuees. The PRECiS domain used for the demonstration has been deliberately simplified to allow a number of different aspects to be explored while keeping the plan to a manageable size. This is for viewing purposes only so that the user can follow what is happening in the demonstration. However, while being a simplification, the types of problem encountered and the solutions proposed by the planner are of definite relevance to military crisis action planning. Larger and more complex plans are available in other Pacifica domains, e.g. `pacifica-3.tf` and these are described in the previous sections.

The demonstration shows three O-Plan agents: Task Assignment, Planner and Execution Agent, together with the World Process which is used as the simulator for the real world. For demonstration purposes only, each of these agents is represented as one or more processes within a single Lisp process. The O-Plan third year demonstration can be repeated using s specialised script instead of the generic script provided in the User Guide. On a number of occasions the user is advised to choose a particular option, e.g. indicating that a given event has executed successfully. This is to ensure that the demonstration proceeds along a defined path. After running through the demonstration a few times it is recommended that the user explore other options which will generate alternative plans and solutions. The scenario and the missions which can be undertaken are described in file `pacifica-4.tf` and the file `README-PACIFICA-4` provides details on repeating the third year demonstration. Both of these files can be found in the default O-Plan Task Formalism directory (refer to the Installation Guide for Details).

7 Satellite Control

This application shows the development of a plan for the control of a simple satellite we have called EUSAT (Edinburgh University Satellite). This satellite is based on the actual University of Surrey's successful UoSAT series of satellites. Earlier research into the application of task planning and scheduling at Edinburgh has included work on defining a task formalism description for O-Plan1 for a spacecraft similar to UoSAT-II but omitting confidential information (which we called BOGUSAT) [6]. This was further extended in the T-SCHED scheduling system [5] which took a scheduling perspective as opposed to a task planning view as in O-Plan1 and generated actual on-board computer Diary commands. The O-Plan project EUSAT model uses the same spacecraft model as BOGUSAT.

A communications wiring harness diagram for EUSAT can be found in Figure 2.

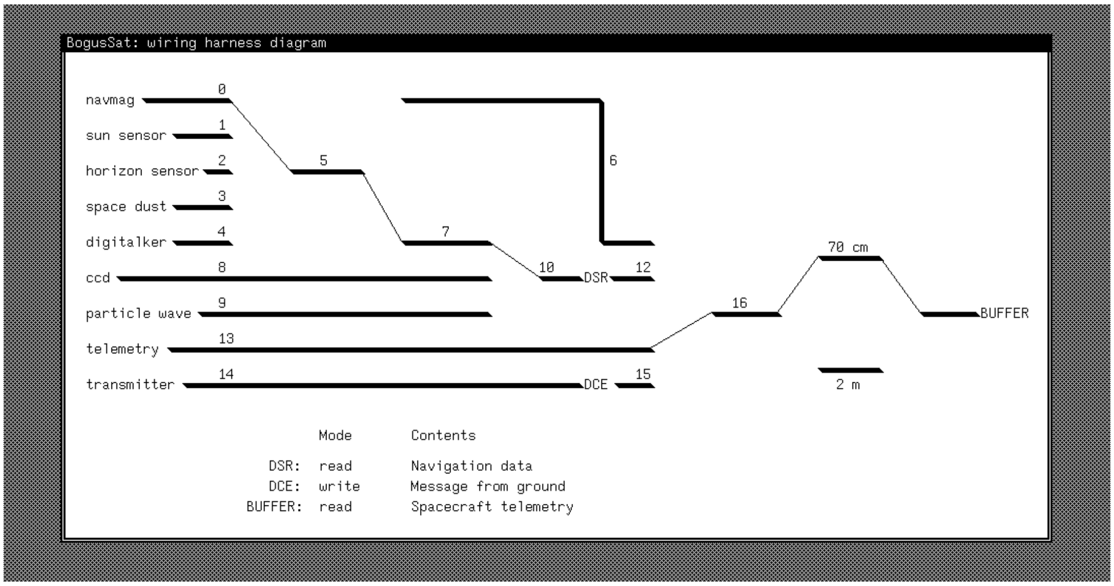

Figure 2: Communications Wiring Harness of the EUSAT satellite

The experiments of the spacecraft are drawn on the left side of the harness and include:

1. Navigational Magnetometer (NAVMAG)
2. Sun Sensor
3. Horizon Sensor
4. Space Dust Analyser
5. Digital Voice Recording (DIGITALKER)

6. Charge Coupled Device (CCD)
7. Particle Wave Experiment

The experiments are connected via a series of *switches* to a tape recorder (DSR) and then to either a `70cm` or `2m` antenna for transmission to the ground. Alternatively some experiments can be connected *directly* to an antenna through `line6` instead of passing through the DSR. One of the experiments, called the DigiTalker, allows for a message to be loaded into a tape recorder (the DCE) from the ground and subsequently re-transmitted at a later time back to the ground. As well as the series of experiments, the satellite must also send telemetry data to the ground.

The movement of data from an experiment to an antenna is modelled as a set of switch settings. Each switch has a valid set of inputs and outputs and these are described as follows:

Switch No	Inputs	Outputs
1	line0 line1 line2 line3 line4	line5
2	line5	line6 line7
3	line7 line8 line9	line10
4	line6 line12 line13 line15	line 16
5	line16	antenna70cm antenna2m
6	antenna70cm antenna2m	ground buffer

A task given to O-Plan describes the requirements for work in a typical day in the life of the spacecraft.

1. **monitor_spacecraft_health**: Send current telemetry data to ground.
2. **capture CCD**: Collect data from the CCD and send it to the ground via the DSR.
3. **capture p_w**: Collect data from the PARTICLE WAVE EXPERIMENT and send it to the `2m` antenna either directly or via the DSR.
4. **capture space_dust**: Collect data from the SPACE DUST ANALYSER and send it to the `2m` antenna either directly or via the DSR.
5. **DCE_communicate**: Receive and re-send a message from and back to ground.

The task specifies the objectives of the mission. This is a series of experiments whose data must be collected and transmitted to a ground buffer via one of two antennas. O-Plan is able to generate a plan for such a mission and give output in a form that could be accepted by the normal diary based dispatch execution system on board a simple spacecraft.

The following table describes the files and task schemas which are available in this domain.

Filename	Task Schemas	Comments
eusat	task_mission_objectives_1	Capture the data from a series of experiments and transmit it to a ground buffer. The order of the data capture is specified by the user and is sequential
	task_mission_objectives_2	Similar to the task above except that the order of data capture is unspecified.

The O-Plan planning agent has been demonstrated generating a plan for such a task and passing it to an O-Plan architecture based execution system for simple dispatch and monitoring to take place.

Other related work at Edinburgh has led to the two planning systems for the European Space Agency. The first was the Plan-ERS [7] system which could generate mission plans for the European Space Agency's ERS-1 spacecraft. This prototype was built in the KEE [8] knowledge representation system and uses a simple plan representation. A second system, OPTIMUM-AIV [1], is able to generate and support the execution of plans for spacecraft assembly, integration and verification. This second planner uses a Goal Structure based plan representation working alongside links to a traditional project management support system (ARTEMIS [9]).

8 Space Platform Construction

This application shows the development of a plan for the construction of one of a number of different Space Platforms. Platforms are constructed from a series of joints, trusses, pressurised modules, solar panels, radiators and antennas. A diagram of a small and a large space platform can be found in Figure 3.

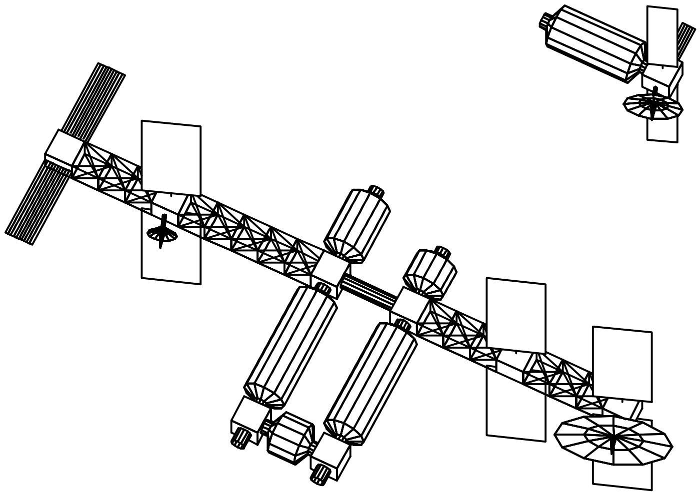

Figure 3: Diagram of two Space Platforms

This example has been included to demonstrate the AutoCAD user interface which has been constructed for O-Plan. The interface is setup from two scripts `oplan-acad-plan-view` and `oplan-acad-world-view` which are defined in Section 4 of the User Guide. The following table describes the files and task schemas which are available in this domain.

Filename	Task Schemas	Comments
space-platform	task_space_platform	Builds a medium sized space platform with 1 antenna, 4 solar panels, 2 radiators, 2 modules, 3 trusses and 3 joints
	task_small_space_platform	Builds a small sized space platform with 1 antenna, 2 solar panels, 1 radiator, 1 module and 1 joint
	task_large_space_platform	Builds a large sized space platform with 2 antenna, 6 solar panels, 2 radiators, 5 modules, 4 trusses, 3 tubes and 8 joints

9 Spanning Problem

This applications shows the development of a solution to a problem involving condition satisfaction. The problem show that certain solutions will be missed by the planner if it chooses to achieve conditions from the start of the expansion which introduced them rather than from any point in the plan state as was the case in earlier planners, such as NOAH [11], Nonlin [12] and SIPE [15]. O-Plan currently allows the widest possible range for achieving a condition (i.e. back to the start of the plan). A future release will allow a more restricted temporal scope. For example, the same scope as for Nonlin, NOAH and SIPE can be specified by the default `after_node_end = begin_of self`.

The problem domain involves two carrying out two setups in a `clean_room` which must remain "clean" in order to later on carrying out some assembly work. One of the main constraints in the problem is that once the `clean_room` is dirty, there is no action to allow it to be cleaned. The `clean_room` is asserted to be clean at the start of the problem.

Filename	Task Schemas	Comments
spanner.tf	spanner_1	there is one solution for task `spanner_1` with the TF options provided . If schema `setup_1_a` is added, then there are two solutions.
	spanner_2	with the TF options provided, there is one solution. There would be no solutions if the achieve had to remain within the temporal scope of its parent. with the TF options provided, if `schema setup_1_a` is added, then there are two solutions. There would then be one solution if the achieve had to remain within the temporal scope of its parent.

10 Creating and Running your own Applications

The TF language has been specifically designed to allow the user to develop their own applications. It is recommended that the user read the TF Manual before attempting to create their own applications. Some of the demonstration applications could be used as skeletal outlines to form the basis of a new application.

This section gives an overview of the structure of a TF application file. The basic format of the TF description file is as follows:

10.1 Task Schemas

The task schemas introduce the tasks the planner is asked to carry out. The first entry of the task schema name line must begin with `task` to indicate to the planner that it is a task schema. The advertised effects of the schema (the `only_use_for_effects` field) is to note that the task is to be achieved and a dummy plan is provided i.e. a start node and a finish node which are always numbered nodes 1 and 2 respectively. The `conditions` fields indicate to the planner any conditions it must achieve. In the example below there are two conditions to achieve by the finish of the plan which together represent a 3 block tower `a b c`.

```
task build_stack_ABC;
  nodes 1 start,
        2 finish;
  orderings 1 ---> 2;
  conditions achieve {on a b} = true at 2,
             achieve {on b c} = true at 2;
end_task;
```

The task schema can also be used to provide a high level action which the planning system must expand into progressively lower level actions until an executable plan is available. In the example below the action to be expanded is to build a house.

```
task build_house;
  nodes 1 start,
        2 finish,
        3 action {build house};
  orderings 1 ---> 3, 3 ---> 2;
end_task;
```

10.2 Global Data and Object Types

These data items specify the objects in the domain and the classes to which they belong. They also declare specific statements which cannot be refuted by the effects of plan actions actions.

These are referred to as **always** statements. For example in the block stacking domain the following global data could be specified.

```
always {cleartop table} = true;      ;;; the table is always a clear object

types objects = (a b c table),       ;;; all objects in the domain
      movable_objects = (a b c);     ;;; stop the planner trying to
                                     ;;; move the table
```

10.3 Application Schemas

The application schemas provide the "actions" of the plan together with information concerning variables, resources, time windows and domain information to help in search control. The schema below defines an action which puts one object on top of another, i.e. {`puton ?x ?y`}. Information concerning the variables `?x` and `?y` is given in two different places:

1. **vars**
   This statement introduces the variables of the schema together with their types if known and other relationships. The type is an enumerated set which allows the planner in the worst case to calculate all possible values the variable could take.
2. **var_relations**
   This statement introduces the equality and inequality relationship between variables in the schema. In the example below the relationship stops the planner binding the object being moved and the destination of the move to the same value.

```
schema puton;
  vars ?x = ?{type movable_objects},
       ?y = ?{type objects},
       ?z = ?{type objects};
  var_relations ?x /= ?y, ?x /= ?z, ?y /= ?z;
  expands {puton ?x ?y};
  only_use_for_effects
          {on ?x ?y}    = true,
          {cleartop ?y} = false,
          {on ?x ?z}    = false,
          {cleartop ?z} = true;
  conditions only_use_if {cleartop ?x},
             only_use_if {cleartop ?y},
             only_use_for_query {on ?x ?z};
end_schema;
```

The **conditions** field informs the planner as to how a particular condition should be satisfied through *condition typing*. This also provides the planner with information as to the strength of commitment to maintaining the condition in a particular way. A complete list of condition types is described in the TF Manual.

The **only_use_for_effects** and **effects** fields inform the planner of the effects which the schema asserts. The **only_use_for_effects** field indicates the effects which the schema can be used to achieve, i.e. its primary effects. The **effects** field indicates the side effects of the action. The schema should not be chosen to specifically achieve them.

The sections above give a brief overview of the structure of a TF application file. Facilities such as resources and time windows have not been described but are fully documented in the TF Manual.

See Section 2.2 of the Installation Guide for instructions on how to invoke O-Plan. Section 3 of this guide describes the steps to be taken when specifying and running a particular application.

References

[1] Arentoft, M.M., Parrod, Y., Stader, J., Stokes, I. & Vadon, H. *OPTIMUM-AIV: A Planning and Scheduling System for Spacecraft AIV* Telematics and Informatics Vol. 8, No. 4, pp. 239-252, Pergamon Press.

[2] Day, D. and McAlpin, S. Supplementary material describing USTRANSCOM transportation planning, Department of the Air Force, 1989.

[3] Department of the Air Force, HQ, USAF, Washington, DC 20330-5000, (May 1987), *Airlift Planning Factors (Military Airlift)*, Air Force Pamphlet 76-2.

[4] Drabble, B. *Applying O-Plan to the NEO Scenarios*, O-Plan Technical Report ARPA-RL/O-Plan/TR/23 version 1, dated 13th June 1995.

[5] Drabble, B. *Mission Scheduling for Spacecraft: The Diaries of T-SCHED*, In the Proceedings of First International Conference on Expert Planning Systems, Metropole Hotel, Brighton, June 1990, Institute of Electrical Engineers, Savoy Place, London.

[6] Drummond, M.E., Currie, K.W. and Tate, A. *O-Plan meets T-SAT: First results from the application of an AI Planner to spacecraft mission sequencing*, 1988, AIAI-PR-27, AIAI, University of Edinburgh.

[7] Fuchs, J.J., Gasquet, B., Olalainty, B., & Currie, K.W. (1990) *Plan-ERS1: An Expert System for generating Spacecraft Mission Plans*, Proceedings of the First International Conference on Expert Planning Systems, Brighton, UK. Available from IEE, London.

[8] Intellicorp, KEE - Knowledge Engineering Environment Manuals.

[9] Lucas Management Systems Ltd, ARTEMIS Project Management System.

[10] Reece, G.A. *Reactive execution in a command, planning, and control environment*, Technical Report 121, Department of Artificial Intelligence, University of Edinburgh, Scotland, 1992.

[11] Sacerdoti, E.D., *The Non-linear Nature of Plans*, In: Proceedings of the Fourth International Joint Conference on Artificial Intelligence, pp206-214, William Kaufman Inc, Los Altos California 1975.

[12] Tate, A.,*Generating Project Networks*, In: Proceedings of the Fifth International Joint Conference on Artificial Intelligence, pp888-893, William Kaufmann Inc, Los Altos California, 1977.

[13] Tate, A., Drabble, B. and Kirby, R. *Spacecraft Command and Control using AI planning Techniques – the O-Plan2 project – final report*, Technical report, USAF Rome Laboratory RL-TR-92-217, Also AIAI-TR-109, University of Edinburgh, 1992.

[14] Tate, A., Drabble, B. and Kirby, R. *O-Plan2: an open architecture for command, planning and control.* In Fox, M. and Zweben, M., (eds.), *Intelligent Scheduling*, Morgan Kaufmann, 1993.

[15] Wilkins, D.E., *Practical Planning: Extending the Classical AI Planning Paradigm*, Morgan Kaufmann Inc, Los Altos California, 1984.