Ontological Models To Support Planning Operations

Pete Bonasso, Mark Boddy*, Dave Kortenkamp, Scott Bell
TRACLabs, Inc., *Adventium Enterprises
bonasso@traclabs.com, mark.boddy@adventiumlabs.com, korten@traclabs.com, scott@traclabs.com

Abstract

A fundamental requirement for success with technology such as automated planning is that it needs to operate on valid models or ontologies of the application domain. Making these models is difficult because the data involved are voluminous, dynamic and come from a variety of sources and formats, so manual entry and maintenance is prohibitive. Using an ontological framework such as OWL can greatly alleviate this effort, but domain experts reason in domain terms, not the formal logic of ontologies. This paper describes an editing system that allows NASA domain experts to construct and maintain ontological information, and yet produce a standard form that can be manipulated by automated planners and other AI applications.

Motivation

Automation and system autonomy are key elements in realizing the vision for space exploration. As crosscutting technology areas, they are applicable to broad areas of technology emphasis, including heavy lift launch vehicle technologies, robotic precursor platforms, efficient use of the International Space Station (ISS), and enabling long duration space missions. The NASA exploration technology program has been developing a set of core autonomy capabilities that can adjust the level of human interaction from fully manual to fully autonomous. One such capability is a procedure representation language (PRL) (Kortenkamp, Bonasso et al. 2008), developed to capture the form of traditional procedures, but allowing for automatic translation into code that can be executed by NASA-developed autonomous executives. Another is interactive aids to generate activity plans. A third concerns tools for authoring and integrating planning information into PRL (Izygon, Kortenkamp et al. 2008).

However, the planning information for procedures that produce planning actions is a relatively small part of the information needed for planning. Beyond resources, conditions and timing constraints, a given planning action (and the procedure it encompasses) requires ontological information, such as the number and types of objects in the domain, their possible states and configurations and the relationships that can hold among them.

The reaction of our subject matter experts (SMEs) to our plan generation system prompted the current line of research. They saw value in the planning and execution tools we were developing, but rightly believed that the lion's share of the work would be developing the domain models. Indeed, for one planning domain effort we estimated that authoring planning actions covered only about 20% of the total modeling effort needed to support planning.

But the problem with making ontological information available to automated systems is three-fold. First, domain experts reason in terms of their domain rather than in terms of the formal logic used by ontology-constructing tools such as Protégé (Research 2011). Second, the states and configurations of the specific objects in the domain are both voluminous and dynamic, making manual entry and maintenance prohibitive. And third, the data required, especially state updates, need to be extracted or imported from other disparate systems.

The modeling framework described herein provides 1) an ontological representation of domain information in a standard ontological representation – the Web Ontology Language (OWL) (OWG 2004) – that can be used by NASA's developing automation software, 2) an interactive editing environment to allow SMEs to construct and maintain the ontological information, and 3) a general, systematic, and maintainable semantic mapping from external data sets into the user-constructed ontology. This paper details that framework and how it will support space operations planning.

Development Environment

Figure 1 shows how we author and integrate ontological information into our interactive planning environment. The environment hosts an interactive procedure editor, PRIDE (Izygon, Kortenkamp et al. 2008), for authoring

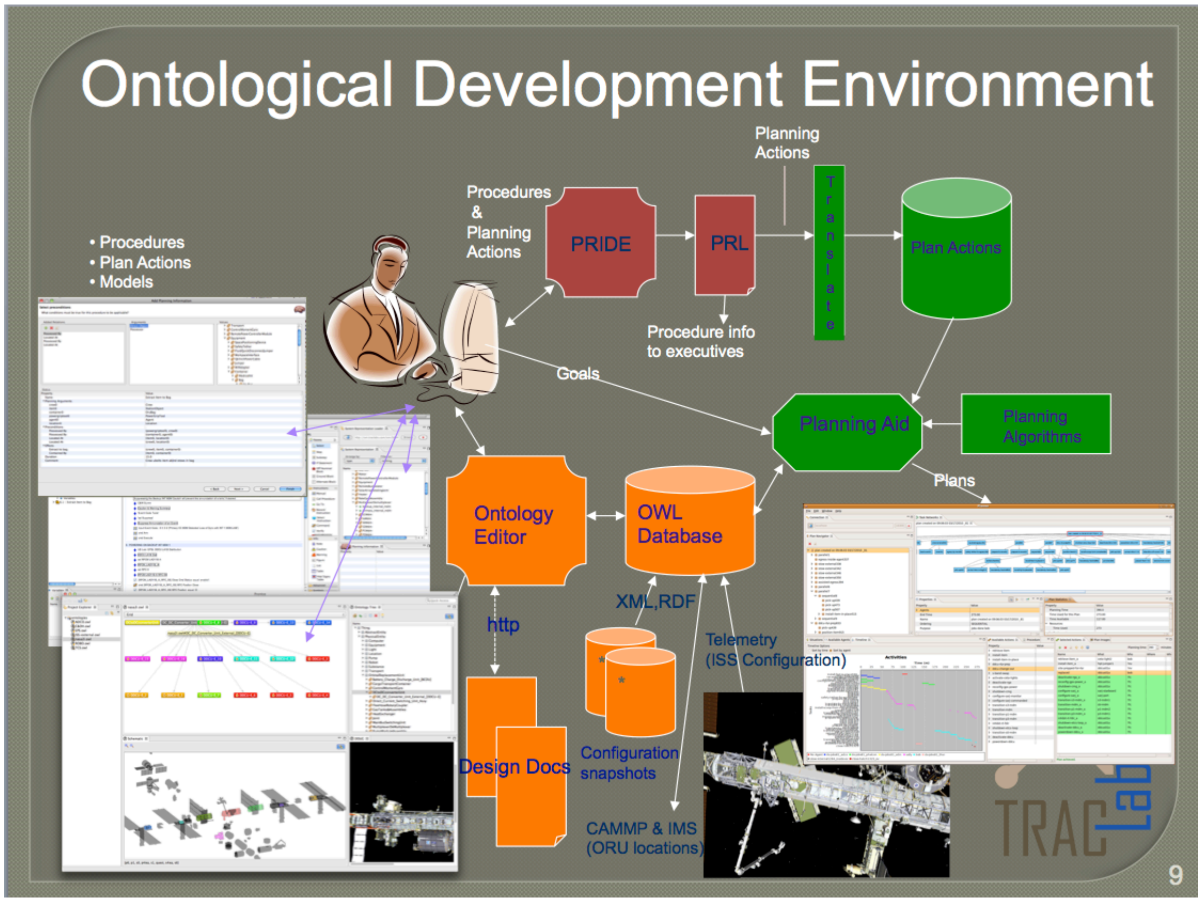
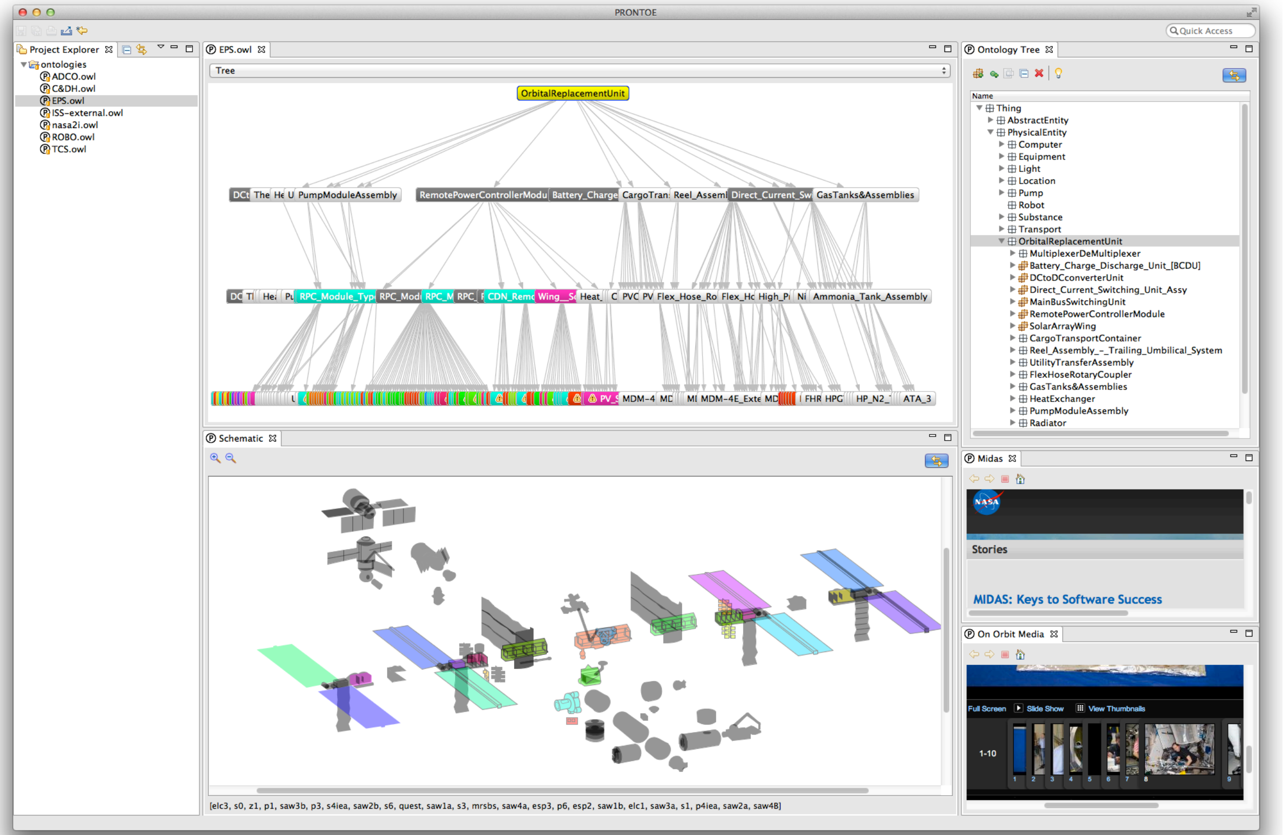


Figure 1 The environment in which to author and validate our ontology. The PRIDE authoring system, includes a procedure editor and the PRIDE Planning Wizard (PPW), represented by the top two screenshots on the far left and the maroon shapes. These and the interactive planning environment (IPE) – the green shapes and the plan screen shot on the right – were developed in previous efforts. The ontology editor, its user interface (bottom left screenshot), databases (orange shapes) and integration with PRIDE and the IPE are the subject of this paper.

procedures. PRIDE also contains the PRIDE Planning Wizard (PPW) for adding planning information to procedures (Bonasso, Boddy et al. 2009). PRIDE can translate the procedure PRL files into languages used by automated executives and planning systems. Both PRIDE and our planner, the Adversarial Planner (AP) (Elsaesser and Stech 2007) a hierarchical task net (HTN) planner, use ontological information stored in OWL files. PRIDE can read OWL files for use in authoring planning and procedural information. PRIDE outputs PRL files containing procedure and planning information; the latter is translated into the planning domain description language (PDDL) syntax (Fox and Long 2003) for AP to ingest[*]. AP can also read OWL triple-store data as the domain model for planning.

[*] AP actually uses an extension to PDDL we call PDDL-e. It is essentially PDDL 2.1 extended to include axioms and hierarchical actions.

Our editor, PRONTOE (the PRide ONTOlogical Editor), provides a capability to author class hierarchies, instances and their relations and provides graphical aids to help the user visualize the various interconnections among those objects. PRONTOE not only interfaces with the planner and procedure executives (not shown in Figure 1), but also with several external data sources. The inventory and location of equipment and tools dealt with by the crew in and around the station is maintained in the Inventory Management System (IMS) for internal stowage and in the Configuration Analysis Modeling and Mass Properties (CAMMP) databases for external stowage, in particular, in the External Configuration Analysis and Tracking Tool (ExCATT) database. Finally, in work planned for the latter part of the project, the OWL models will be updated using data streamed from the International Space Station (ISS).



[elc3, s0, z1, p1, saw3b, p3, s4iea, saw2b, s6, quest, saw1a, s3, mrsbs, saw4a, esp3, p6, esp2, saw1b, elc1, saw3a, s1, p4iea, saw2a, saw4B]

Figure 2 Screenshot of the PRONTOE user interface. The user selects an ontology file on the left and PRONTOE produces several related views: a class/subclass hierarchy (top right), an ontology graph (top center) and a color-coded depiction of the area location of all the instances below the hierarchy level selected. Additionally, one can access a variety of Web-based data such as the Mission Integration Database Applications System (MIDAS), to obtain weight and size data, and on-orbit photos and videos of the equipment in the ontology.

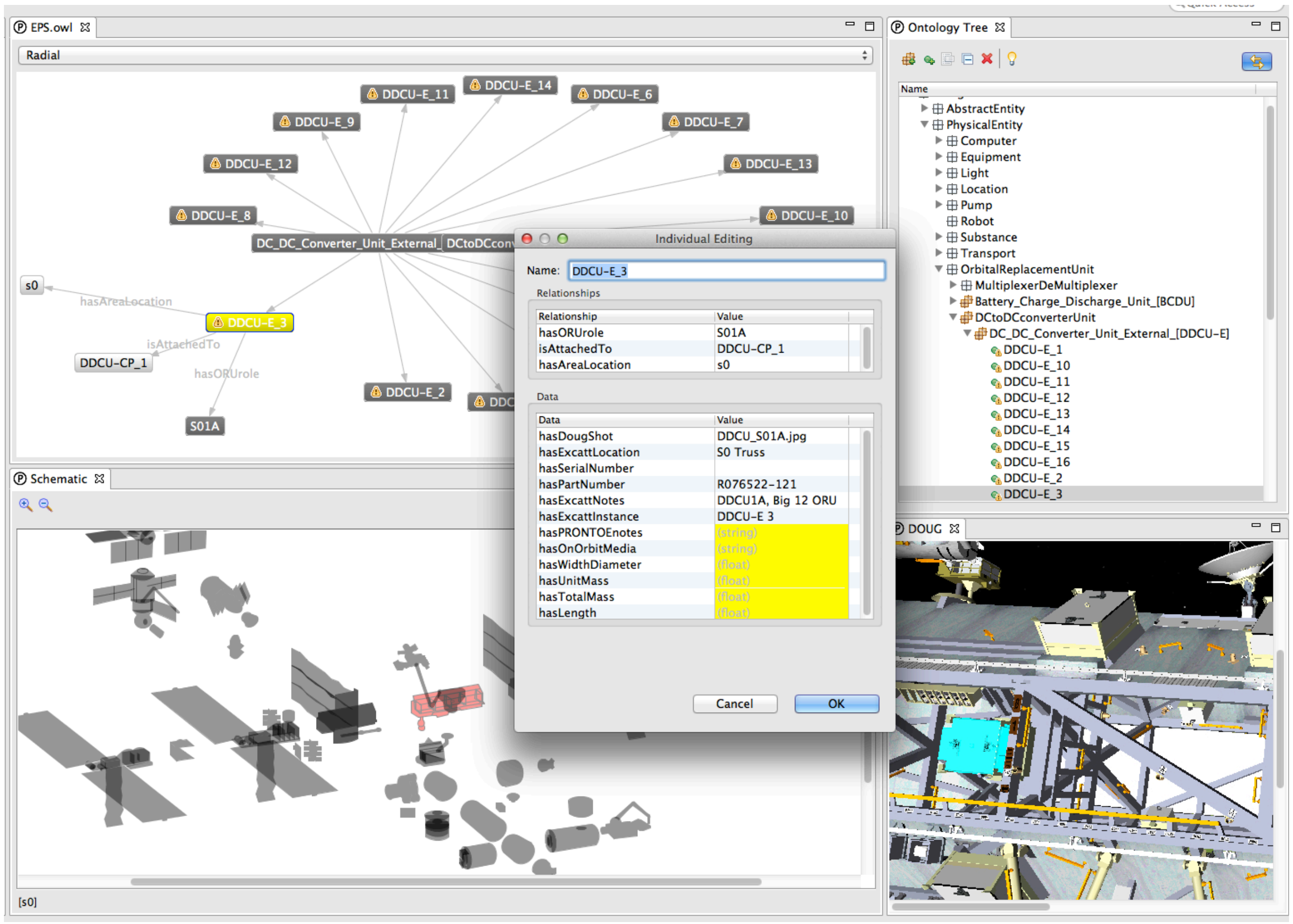
Types of Ontological Information

As mentioned above, our ontology has class/subclass relations among the objects of interest as shown in Figure 2. Additionally, we have data properties and object properties for the classes and instances in the hierarchy. These are formal relations with a domain and range. The restrictions on the domain and ranges help minimize errors during user input (see Figure 3).

Additionally, the data in the objects and their relationships with other objects constitutes a specific *configuration* that can be saved as a data set. Then other apps – planners and executives – can be directed to use a specific data set during operations. Using data sets that represent future configurations allows users to generate plans for anticipated future operations.

We have divided our ontology into a *base* and *kernel-extensions*. A base ontology is the set of classes and their relations that reflect epistemic choices upon which the rest of the models depend. It also provides a default level of axioms (discussed below). Thus, by definition, the base ontology will not be edited by the user (so it is not shown in the list of ontologies in Figure 2). Kernel extensions import from the base ontology and contain ontological information related to a sub-discipline, such as, for the ISS, the electrical power system (EPS), the thermal control system (TCS) or extravehicular activities (EVA or ISS-external in the list of ontologies in Figure 2). In order to make editing and extending the ontology easy for SMEs, we include in each kernel representative classes, subclasses and instances so that the user can copy and edit to create new classes and instances as the need arises.

Axioms and constraints constitute another important type of information included in our ontology is that we col-



Individual Editing

Name: DDCU-E_3

Relationships

Relationship	Value
hasORUrole	S01A
isAttachedTo	DDCU-CP_1
hasArealLocation	s0

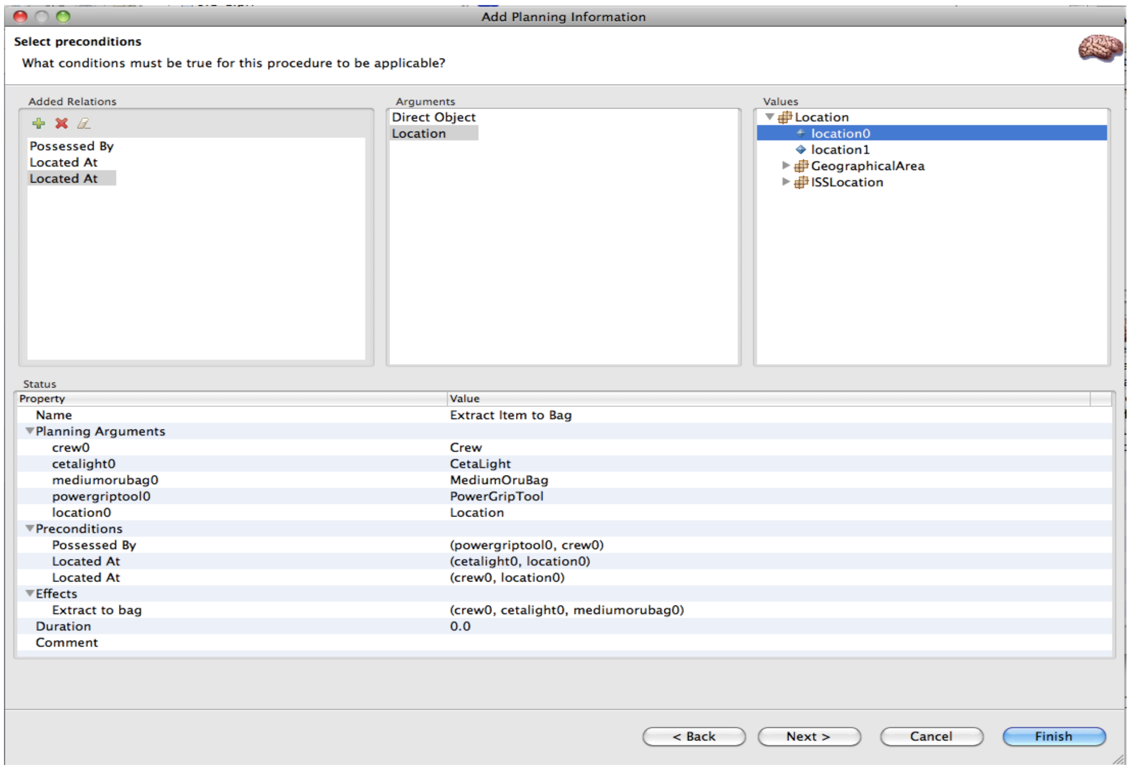
Data

Data	Value
hasDougShot	DDCU_S01A.jpg
hasExcattLocation	S0 Truss
hasSerialNumber	
hasPartNumber	R076522-121
hasExcattNotes	DDCU1A, Big 12 ORU
hasExcattInstance	DDCU-E 3
hasPRONTOEnotes	(string)
hasOnOrbitMedia	(string)
hasWidthDiameter	(float)
hasUnitMass	(float)
hasTotalMass	(float)
hasLength	(float)

Figure 3 Details of an EPS DC to DC converter unit (DDCU) in the ontology. The data properties are shown in the lower part of the dialog, while the object properties are shown at the top. Some data properties are visual, such as the specific location of the DDCU in the 3D depiction in the bottom right. The object properties are also displayed graphically, showing connections to other objects in the ontology. A warning icon on any item indicates missing information, which is highlighted in yellow in the corresponding dialog.

lectively term *directives*. Axioms are rules that manage bookkeeping during planning, e.g., moving a container changes the location of all the items in the container, and for domain physics, such the loss of fluid flow when a pump loses power. PRONTOE supports the authoring of axioms as they pertain to the class/relations information, which is the basis of preconditions and effects authored with the PPW. But constraints model operational choices, not physical laws, like the constraint that at least two gyros must be operational for certain navigation modes of station flight, or the flight rule that restricts certain modes of controlling a robot arm when it's too close to structures. These are mostly related to the planning part of the architecture, not the domain model, because constraints can be overridden by human order.

Axioms are used in planning to infer additional changes to the world state caused *indirectly* by action effects. In PRONTOE, these inferences must also be made in order to keep a given configuration (which is just a description of a world state) consistent, when any new information is added *directly*, either by the user or through interaction with an automated data source. When data on the conditional side of an axiom is updated, some mechanism must fire to infer the appropriate logical changes in the configuration. One approach is to use SWRL (OWG 2004), which can be used to form rules whose left hand sides (LHSs) and right hand sides (RHSs) are OWL relations, and which can be fired by an ontological reasoner such as Hermit (Library 2011). In PRONTOE, the user can invoke such a reasoner at any time during the editing process by clicking the light bulb icon above the ontology tree (see Figure 3 for example).



Select preconditions

What conditions must be true for this procedure to be applicable?

Added Relations

Possessed By
Located At
Located At

Arguments

Direct Object
Location

Values

Location
location0
location1
GeographicalArea
ISSLocation

Status

Property	Value
Name	Extract Item to Bag
Planning Arguments	
crew0	Crew
cetalight0	CetaLight
mediumorubag0	MediumOruBag
powergriptool0	PowerGripTool
location0	Location
Preconditions	
Possessed By	(powergriptool0, crew0)
Located At	(cetalight0, location0)
Located At	(crew0, location0)
Effects	
Extract to bag	(crew0, cetalight0, mediumorubag0)
Duration	0.0
Comment	

Figure 4 A screenshot of the PRIDE Planning Wizard (PPW) interface for action authoring. The relations used for preconditions and effects are imported from our ISS ontology.

Implications for the Planning Wizard

The preconditions and effects used by the PPW during the authoring of plan actions are derived from the object and data properties in the ontology (see Figure 4). But there is a complex interplay between PRONTOE and the PRIDE Planning Wizard (PPW) as highlighted in the use of directives, because both tools must interact with the domain model. PPW creates and modifies actions, sometimes necessitating changes to the ontology and instances in the domain; PRONTOE makes changes as needed to the ontology and instances, with possible corresponding changes to actions. For example, consider the addition of a new (entirely mythical) class of tool, the laser_emulsifier and its three instances, LE1, LE2 and LE3. Because it's a type of tool, the domain model will endow it with a location property as well as the constraint that only one person can use it at a time. When the user wishes to author an emulsify action, either the author or some intelligent machinery in the PPW must assert that the crew is co-located with the instance of laser_emulsifier and that during the action, the tool is unavailable to anyone else. We can (and in some languages will be required to) model this kind of resource usage without axioms by encoding it in the action's preconditions and effects, but that only strengthens the interaction between PRONTOE-mediated changes to the ontology and PPW-authored action definitions.

In our current design, we intend to extend the capability of the plan action translator shown in Figure 1 such that it will encode the axioms in the action's preconditions and effects, or take advantage of axiomatic inferences if the target planner can use them.

Implications for Planning

Of course the effect of a change in the ontology – data or directives – can have a significant affect on generated plans. Take the case of the internal thermal control system (ITCS) of the US Lab on the ISS. In the top picture of Figure 5, multiplexer-demultiplexers S01 and S11 (multiplexer-demultiplexers or MDMs are computers on the ISS) are the primary and backup controllers for the Loop A coolant pump that is part of the External Thermal Control System (ETCS). This coolant loop draws the heat from the low temperature (LT) loop of the ITCS out to the radiators. Loop B does the same for the medium temperature (MT) loop. A higher-level MDM, EXT-1, controls the heat exchangers in the lab ITCS and MDM PMCU-1 controls DDCUS01A. DDCUS01A powers the Loop A pump as well as MDMs S01, S11 and EXT-1. The LAB ITCS is running in "dual loop" mode, that is, both the medium temperature and low temperature loops are operating.

The bottom picture shows the ETCS and ITCS situation anticipated when DDCUS01A is powered down. Control and power will both be lost for Loop A's pump, and power will be lost to MDMS EXT-1, S01 and S11. These MDMs must be transitioned to an off state. But since the heat exchangers in the lab are controlled by EXT-1, MDM EXT-2, the backup for EXT-1 must take over control of the heat

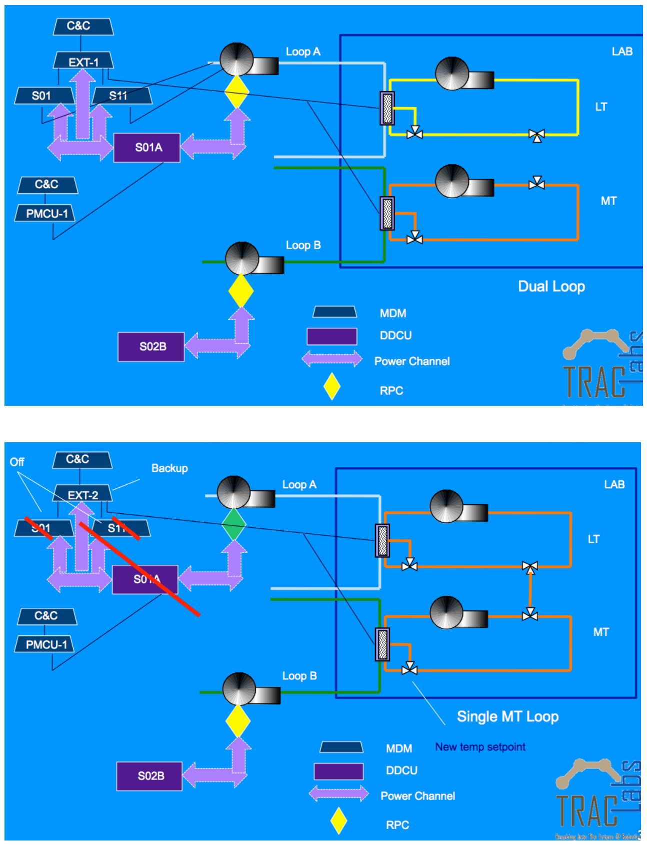


Figure 5 A portion of the US Laboratory Internal Thermal Control System (ITCS) in dual-loop mode (top) and single loop mode (bottom).

exchangers. Also, the fault detection software in PMCU-1 needs to be suppressed before the DDCU is deactivated. Finally, with the loss of Loop A, the low temperature loop in the lab won't function properly, so the lab ITCS needs to be reconfigured to single loop mode. In this mode, the three-way valves are set so that all the water passes through the MT heat exchanger that is serviced by Loop B.

In our ontology we model relationships of the various power and computer units to the thermal control objects, some of which are shown Figure 6. The AP planner reads the initial situation from a specific OWL configuration file and generates plans accordingly. When AP generates a plan to remove and replace DDCUS01A, if the lab ITCS is in single loop mode, no lab ITCS action must be taken (see top of Figure 7 on the page following Figure 6). But when the ITCS is in dual loop, the necessary additional computer and thermal control actions are added to the plan (bottom of Figure 7).

Besides reading in the starting situation for planning, the planner will also be a contributor to the current ontological configuration. AP has an execution monitoring (EM) mode that passes action goals to the agents of execution. This requires an execution system that can interface with AP at the level of its effects, such as 3T (Bonasso, Firby et al. 1997). The lower level execution system will assert state information for all the affected devices, but AP must also assert the *action-level* PPW-authored effects of successful actions into the external ontological configuration from which it generated a plan in the first place. Hence AP both reads from and updates the ontology during planning.

AP also serves as a sort of rough verification engine. We have a set of over 100 planning problems in a given ISS scenario that serve as our test suite. As the ontology is edited, we run AP on the test suite to insure that AP can still obtain valid plans for those problems, or else edit the definitions of the problems to match the changed ontology.

Related Work

The bulk of the efforts in knowledge engineering (KE) for planning involve AI programmers eliciting planning information from domain experts, and then using KE aids to model and validate this information. Examples are (Fernandez, Borrajo et al. 2007) and (Simpson 2007), and the work of Biundo and Stephan (Biundo and Stephan 1997) on a domain modeling tool that supports incremental, modular model development by extending and refining existing models.

Work on meta-theories (e.g., (Herzig and Varzincak 2007)) may be considered related in that it attempts to view an ontology from a perspective of common concepts and elements. Myers' work on planning domain meta-theories (Myers 2000) falls in this vein, where she discusses such things as characterizing air/land/water as "transport media", and that movement concepts involve a source and a destination. Our work on a base ontology as distinct from kernel ontologies is similar and our interactive approach will use abstraction levels to make the authoring of models easier for the user.

Ontological engineering (OE) has been a regular activity in the AI community for many years. In 1999 it was considered in its infancy for lack of use of widely accepted methodologies (Lopez), but as late as 2007, the majority of OE researchers still did not use any methodology (Cardoso 2007). Yet, most OE research accepts as fundamental the need for an efficient, consistent paradigm for knowledge engineering ontologies (Soares and Foncesca 2009).

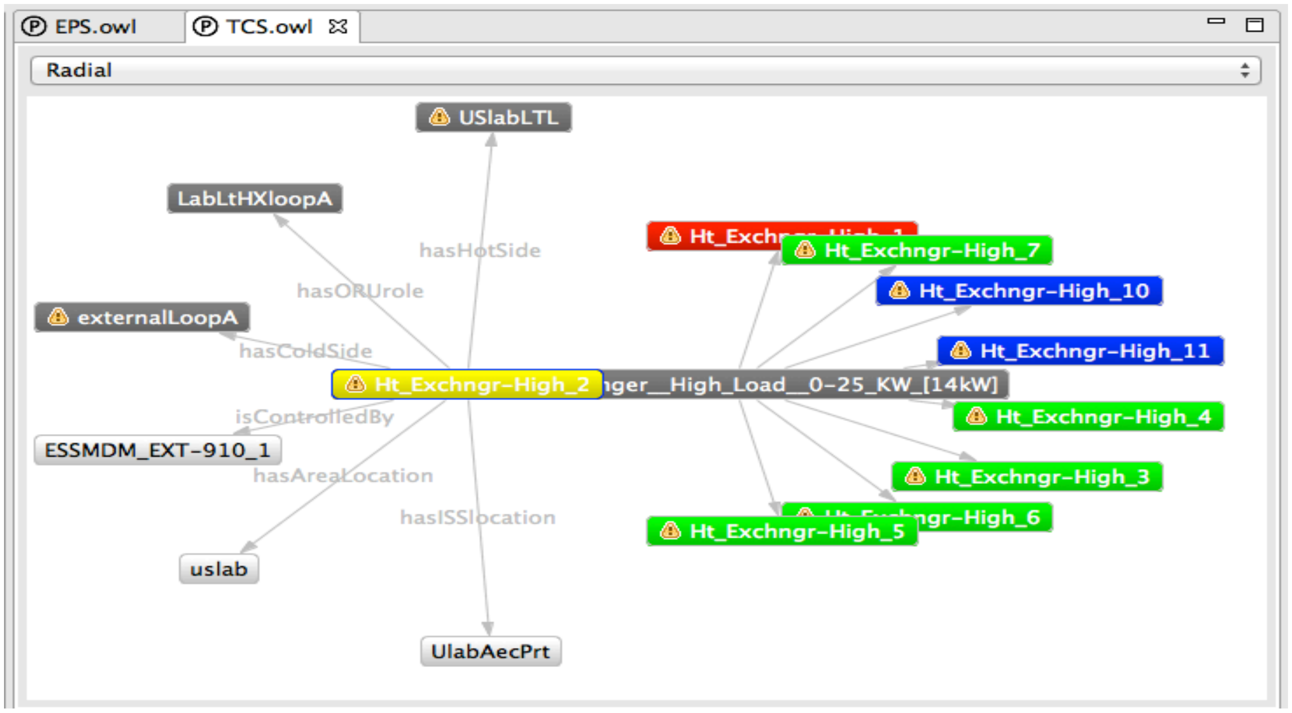


Figure 6 A graphical view of the relations of the heat exchanger for the US Lab low temperature loop. This constitutes a small portion of the relationships among the various disciplines in the ontology. The radial view helps more clearly identify the relations rather than the tree view. In the radial view, the classes are in the center, with the instances radiating around them.

The work on developing flight rules by Barreiro et al (Barreiro, Chachere et al. 2010) is directly complementary to our work on constraints, though the developed flight rules were not cast into an ontological framework but into a specialize representation to be used by planners and schedulers.

Summary & Future Work

In this paper we have described a framework that provides 1) an ontological representation of domain information in a standard format that can be used by NASA's developing automation software, 2) an interactive editing environment to allow SMEs to construct and maintain the ontological information, and 3) a general, systematic, and maintainable semantic mapping from external data sets into the user-constructed ontology.

During our Phase 1 effort, we designed a set of capabilities for PRONTOE that were validated by our SMEs. As of this writing we are iterating our PRONTOE prototype through a series of SME demonstrations. Initial reactions to the implemented prototype are favorable and discussions tend to center on the use of the prototype in workflows. EVA and robotics flight controllers have an understandably different workflow than the other console operators. The former are more concerned with the external location of objects, tools and equipment than are the flight controllers who monitor the core systems. As a result we plan to add a set of capabilities geared toward rapidly visualizing and updating the external location data in the ontology.

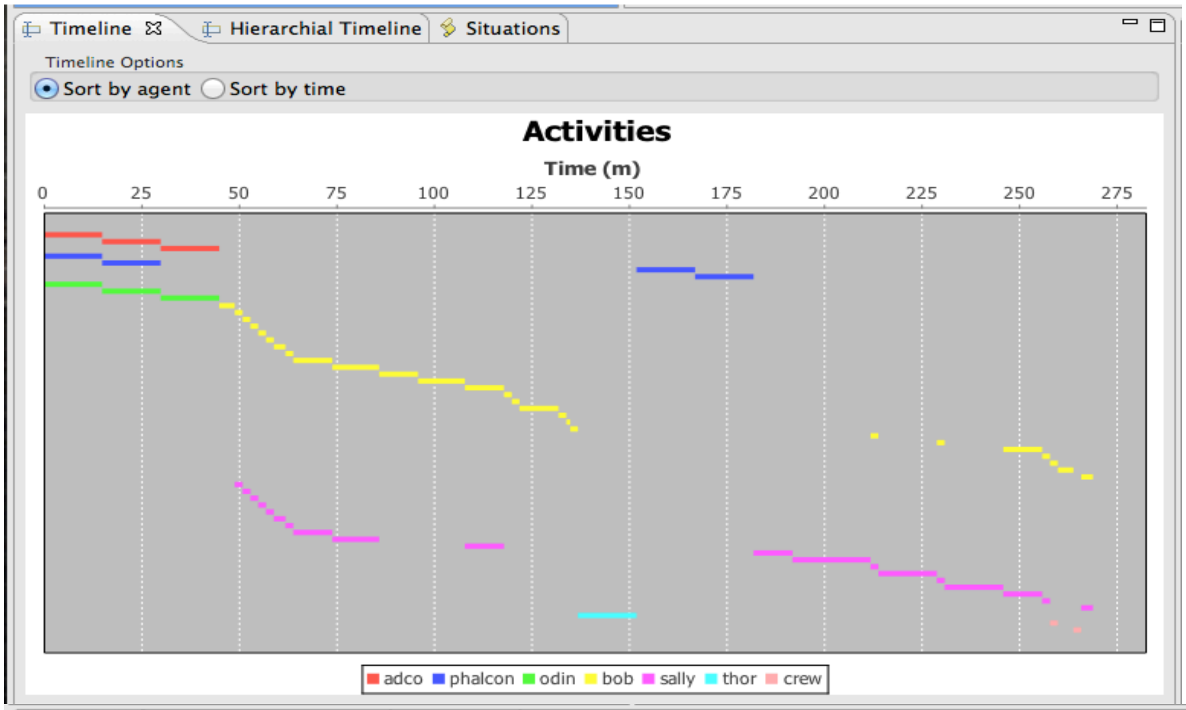
References

Barreiro, J., J. Chachere, et al. (2010). Constraint and Flight Rule Management for Space Mission Operations. . International Symposium on Artificial Intelligence, Robotics, and Automation in Space. Sapporo, Japan

Biundo, S. and W. Stephan (1997). System Assistance in Structured Domian Model Development. IJCAI-97. M. Kaufmann. NAgoya, Japan.

Bonasso, R. P., M. Boddy, et al. (2009). Enhancing NASA's Procedure Representation Language to Support Planning Operations. IWPSS 09. Pasadena, CA.

Bonasso, R. P., J. R. Firby, et al. (1997). "Experiences with an Architecture for Intelligent, Reactive Agents." Journal of Experimental and Theoretical Artificial Intelligence **9**: 237-256.



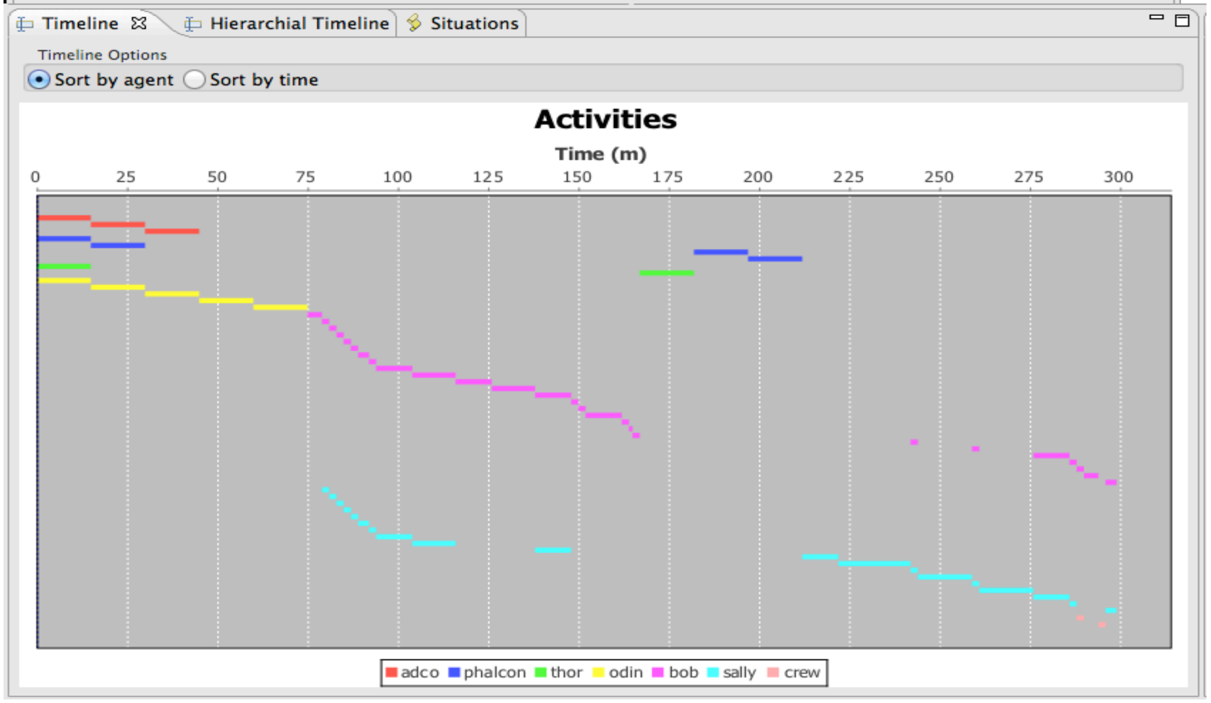


Figure 7 Two plans to remove and replace DDCUS01A. Activities are color-coded by agent (shown in the legends) with the EVA tasks in yellow and pink in the top figure and pink and blue in the bottom figure. The top plan is generated when the US Lab ITCS is in single-loop mode. In this plan, ODIN has three tasks and THOR has one. The bottom plan to remove and replace DDCUS01A is generated when the US Lab ITCS is in dual loop mode. The activity color-coding is different, with ODIN being yellow and THOR being green. With the ITCS in dual loop mode (and DDCUS01A controlling Loop A), THOR has the extra task to reconfigure the ITCS to single loop mode, while ODIN has two added tasks to transition MDMs S01 and S11 to off. The resulting plan is about 30 minutes longer.

Cardoso, J. (2007). "The Semantic Web Vision: Where Are We?" IEEE Intelligent Systems: 22-26.

Elsaesser, C. and F. J. Stech (2007). Detecting Deception. Adversarial Reasoning: Computational Approaches to Reading the Opponent's Mind, . A. K. a. W. M. McEneaney. Boca Raton, Chapman & Hall/CRC: 101-124.

Fernandez, S., D. Borrajo, et al. (2007). "PLTOOL: A Knowledge Engineering Tool for Planning and Learning. ." The Knowledge Engineering Review **22**: 1-24.

Fox, M. and D. Long (2003). "PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains." Journal of Artificial Intelligence Research **20**: 61-124.

Herzig, A. and I. Varzincak (2007). "Metatheory of Action: Beyond Consistency." Artificial Intelligence **171**: 951-984.

Izygon, M., D. Kortenkamp, et al. (2008). A procedure integrated development environment for future spacecraft and habitats. Space Technology and Applications International Forum (STAIF). Albuquerque, NM, American Instutute of Physics. **969**.

Kortenkamp, D., R. P. Bonasso, et al. (2008). A Procedure Representation Language for Human Spaceflight Operations. The 9th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS-08). Los Angeles, CA.

Library, O. C. (2011). "Hermit Reasoner." from http://www.hermit-reasoner.com/.

Lopez, F. M. (1999). Overview of Methodologies for Building Ontologies. IJCAI-99 Workshop on Ontologies & Problem-Solving Methods. Stockholm, Sweden.

Myers, K. L. (2000). Domain Metatheories: Enabling User-Centric Planning. Workshop on Representational Issues for Real-World Planning Systems (AAAI-2000), AAAI Press.

OWG (2004). "Semantic Web Rule Language." from http://www.w3.org/Submission/SWRL/.

OWG (2004). "the Web Ontology Language (OWL)." from http://www.w3.org/TR/owl-features/.

Research, S. C. f. B. I. (2011). "protégé home page." from http://protege.stanford.edu/.

Simpson, R. M. (2007). "Structural Domain Definition using GIPO IV." The Knowledge Engineering Review **22**: 117-134.

Soares, A. and F. Foncesca (2009). Building Ontologies for Information Systems: What we have, what we need. iConference '09. Chapel Hill, NC.