Ontological Models To Support Space Operations

R. Peter Bonasso[*], Mark Boddy, David Kortenkamp, and Scott Bell

TRACLabs Inc., Houston TX USA (Bonasso, Kortenkamp, and Bell)

Adventium LLC, Minneapolis MN USA (Boddy)

July 1, 2013

Abstract

A fundamental requirement for success with technology that supports space operations, such as automated procedures and interactive plan generation, is that the technology must 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 procedure authoring and execution, automated planning and other AI applications.

*Corresponding author. E-mail: bonasso@traclabs.com

1 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) [8], 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 capability is interactive aids to generate activity plans. A third concerns tools for authoring and integrating planning information into PRL. Such a tool is the Procedure Integrated Development Environment (PRIDE) [7]. 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. But the problem with making ontological information available for end users in automated systems is threefold. 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. 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) – that can be used by NASA's developing automation software, 2) an interactive editing environment an extension to PRIDE to allow subject matter experts (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.

2 PRONTOE

Over the past year we worked with flight controllers and other SMEs to design an ontological authoring system that would allow an end user to author and edit an ontological model of the International Space Station (ISS) in support of procedure authoring and maintenance as well as the interactive generation of operations plans. The resulting design included a class hierarchy of the instances of items in and around the station, the data associated with those instances and a set of directives axioms and operational constraints that bear upon the state of the models. This year we made available to our SMEs an initial prototype of the editing system called PRONTOE (the Pride ONTological Editor) to obtain feedback on our modeling approach and its usefulness in supporting procedure and plan development. The description of PRONTOE in this report reflects that feedback. The ontology is organized into a base ontology of domain concepts that is extended through model kernels, representing the different flight disciplines involved in ISS operations. The information in the ontology does not cover the whole of the ISS, but rather, is complete enough in its conceptual framework to allow the users to incrementally extend the kernel to support on-going operations.

PRONTOE is an extension of PRIDE, which is used for authoring procedures and rendering them in an XML schema known as PRL. PRIDE includes the PRIDE Planning Wizard (PPW) that allows the user to add resource, timing and constraint information to procedures. The resulting PRL files can be read by procedure viewers and executives and can be translated into standard planning languages such as PDDL [5] and ANML [13], so that AI planners can generate operations plans with the authored procedures. Since PRIDE and the PPW rely on a domain ontology for their use, PRONTOE allows the user to edit the underlying ontological models themselves, thus completing the human authoring process.

PRONTOE not only interfaces with planners and procedure executives 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).

2.1 PRONTOE Interface

The PRONTOE application is an Eclipse-based interactive graphical editor that presents the model information in taxonomy trees, concept graphs and as a distribution of objects on a graphic depiction of the ISS (see Figure 1). PRONTOE will be used by flight controllers of the core subsystems of the ISS e.g., power, motion control, life support but also by operators concerned with managing non-instrumented equipment around the station, such as extra-vehicular activity (EVA) and robotics. In Figure 1 the kernel extensions for the various flight disciplines are listed in the Project Explorer window, including an integrated kernel, nasa2i, used by EVA and robotics personnel. The different disciplines wanted to configure the windows in PRONTOE to more easily support their work, so we have added the capability to save window configurations called perspectives for later reuse. PRONTOE comes with default Core and EVA perspectives, shown in the tabs at the upper right of Figure 1.

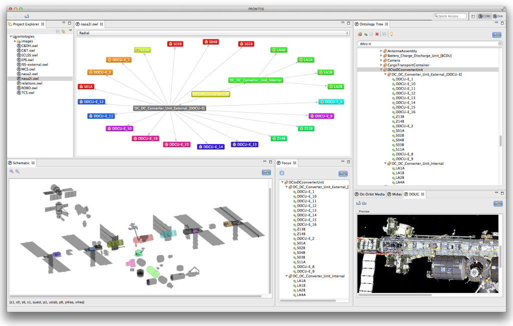
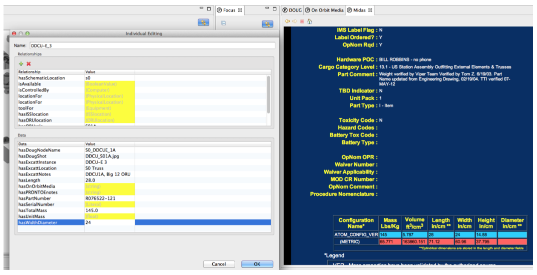


Figure 1: Screenshot of the PRONTOE user interface. The user selects an ontology file on the left and PRONTOE produces several related views: a class/subclass Ontology Tree (top right), an ontology graph (top center) and a color-coded depiction of the area location of all the instances below the hierarchy level selected (bottom right). An incremental search field above the ontology tree allows for quick access to named items. While the ontology tree shows the entire ontology the Focus view (bottom center) shows only those items selected. In the panes on the bottom right, one can display screenshots of the equipment, and 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.

PRONTOE allows the user to add, delete or modify class and instance information, to include the relations among the instances. Figure 2 shows an editing dialog used for adding or modifying the data and relations associated with a given piece of station equipment known as a DC-to-DC Converter Unit (DDCU). Some of the required data associated with an item can be obtained from external databases avail-



Name: DDCU-E_3

Relationship	Value
hasSchematicLocation	s0
isAvailable	(BooleanValue)
isControlledBy	(Computer)
locationFor	(PhysicalLocation)
locationFor	(PhysicalLocation)
toolFor	(Equipment)
hasISSlocation	(ISSlocation)
hasORUlocation	(ORUlocation)

Data	Value
hasDougNodeName	S0_DDCUE_1A
hasDougShot	DDCU_S01A.jpg
hasExcattInstance	DDCU-E 3
hasExcattLocation	S0 Truss
hasExcattNotes	DDCU1A, Big 12 ORU
hasLength	28.0
hasOnOrbitMedia	(string)
hasPRONTOEnotes	(string)
hasPartNumber	R076522-121
hasSerialNumber	(Literal)
hasTotalMass	145.0
hasUnitMass	(float)
hasWidthDiameter	24

IMS Label Flag : N
Label Ordered? : Y
OpNom Rqd : Y

Hardware POC : BILL ROBBINS - no phone
Cargo Category Level : 13.1 - US Station Assembly Outfitting External Elements & Trusses
Part Comment : Weight verified by Viper Team Verified by Tom Z. 6/19/03. Part Name updated from Engineering Drawing, 02/19/04. TTI verified 07-MAY-12
TBD Indicator : N
Unit Pack : 1
Part Type : I - Item

Toxicity Code : N
Hazard Codes :
Battery Tox Code :
Battery Type :

OpNom OPR :
Waiver Number :
Waiver Applicability :
MOD CR Number :
OpNom Comment :
Procedure Nomenclature :

Configuration Name*	Mass Lbs/Kg	Volume ft^3/cm^3	Length In/cm **	Width In/cm	Height In/cm	Diameter In/cm **
ATOM_CONFIG_VER	145	5.787	28	24	14.88	
(METRIC)	65.771	163860.151	71.12	60.96	37.795	

**Cylindrical dimensions are stored in the length and diameter fields

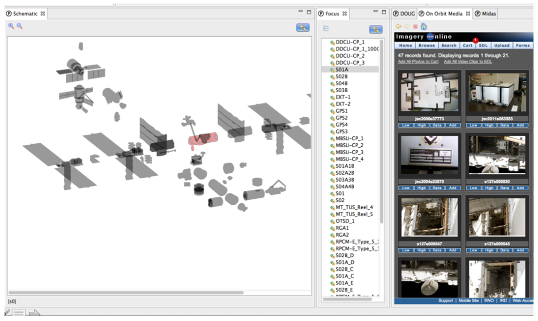
*Legend

Figure 2: A subset of the PRONTOE windows showing the data and relation editing dialog for a piece of power equipment on the left, and on the right, a view of that items mass properties as retrieved by searching MIDAS using the part number.

able via the Internet. For instance, Figure 2 shows the mass properties associated with the item designated as DDCU-E_3. These can be copied into the items dialog directly from the MIDAS search page.

3 Location of External Equipment

Of particular interest to EVA and robotics flight controllers are the number, type and location of external equipment. PRONTOE has a number of facilities to assist with quickly searching for location items. To begin with, we populated the ontology from the parts listed in the part catalog of and the instances from the ExCATT database. When an EVA or robotics activity relocates a piece of external equipment, the flight control team sends those changes to ExCATT (as well as to the IMS). PRONTOE exchanges information with ExCATT via JavaScript Object Notation (JSON) files.



DDCU-CP_1
DDCU-CP_1_100
DDCU-CP_2
DDCU-CP_3
S01A
S02B
S04B
S03B
EXT-1
EXT-2
GPS1
GPS2
GPS4
GPS3
MBSU-CP_1
MBSU-CP_2
MBSU-CP_3
MBSU-CP_4
S01A1B
S02A2B
S03A3B
S04A4B
S01
S02
MT_TUS_Reel_4
MT_TUS_Reel_5
OTSD_1
RGA1
RGA2
RPCM-E_Type_5_
RPCM-E_Type_5_
S02B_D
S01A_D
S02B_C
S01A_C
S01A_E
S02B_E

47 records found. Displaying records 1 through 21.
Add All Photos to Cart Add All Video Clips to EDL

jsc2006e37773
jsc2011e063583
jsc2004e23870
s127e009530
s127e009547
s127e009545

Figure 3: A subset of the PRONTOE windows showing the schematic view of the ISS on the left. The user has clicked on truss segment S0 and the Focus View lists the equipment located there. On the right is a set of assembly and closeout photos of the DDCU designated as S01A. This screenshot was taken with PRONTOE in the EVA perspective, which concentrates on the schematic and the group of tabs in the lower right.

Additionally, all of the items in PRONTOE have a schematic location data slot. If one clicks on a subarea of the ISS schematic (see Figure 3), a list of all the items located there is shown in the Focus view. Further, one can add screenshots of the item from NASA's Dynamic On-board Ubiquitous Graphics (DOUG) system (see Figure 4), a 3D rendering system that is kept up to date with location changes from the flight controllers.

ExCATT only gives general locations as shown in the schematic view. But in order to plan paths to and manipulate the equipment, EVA and robotics need a specific physical location. As part of the PRONTOE development, we have designed location specifications for each of the external areas of the ISS the truss segments, the external stowage places and the station modules, such as the

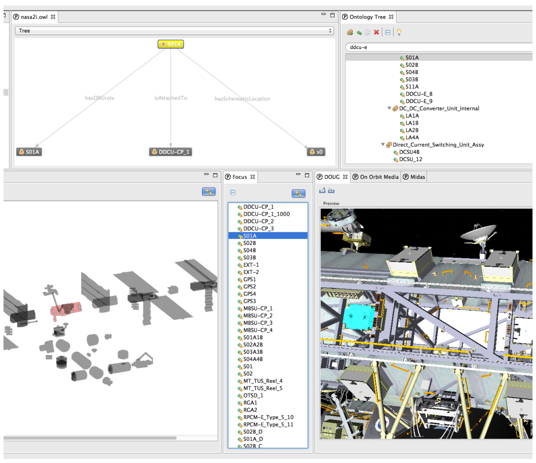


Figure 4: A subset of the PRONTOE windows showing the relationships associated with the DDCU known as S01A in the ontology tree, and the DOUG shot of the DDCU (highlighted in blue) in the lower right. The small slanted arrow icons in the DOUG window allow the user to interface directly with the DOUG application.

US Lab and nodes. A truss location is a ten-letter code specifying the truss segment, bay, face (some segments are six sided and some are four sided) and a zenith-middle-nadir and port-middle-starboard location on the face (see Figure 5).

The external stowage places are External Stowage Platforms (ESPs) and EXpedite the PRocessing of Experiments to Space Station (ExPRESS) logistics Carriers (ELCs). The location specification consists of a sixteen-letter code designating the stowage platform name, the position on the platform (up to 8, depending on the platform), and three three-letter designations for the orientations aft, forward, nadir, zenith, starboard and port with respect to the forward direction of ISS travel, with zenith towards the Earth (see Figure 6).

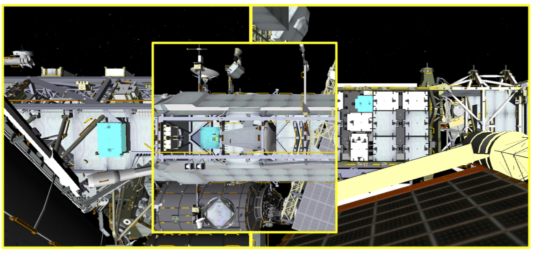


Figure 5: Sample truss location specifications. The center picture is that of a nitrogen tank assembly at the mid-port section of face 1 of bay 6 on the P1 truss, or P1B06F01MP. The left picture is a main bus switching unit on the aft face, face 4, of the S0 truss segment. Its specification is S0B02F04MP. The right picture is s DDCU on one of the four-sided trusses known as integrated equipment assemblies (IEAs). Its location is given as S4B23F04ZM.

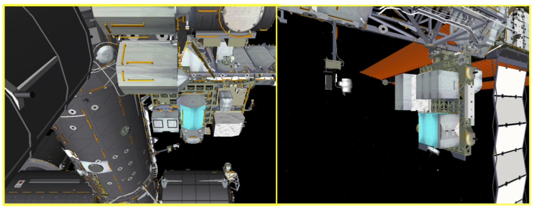


Figure 6: Sample external stowage location specifications. The left picture is a flex hose rotary coupler on ESP 2, position 7. Its orientation is aft, port, nadir, so the full spec is ESP0207aftPrtNad. The right picture is that of a pump module assembly on ELC01. Its location spec is ELC0107fwdStbNad.

Modules are cylinders that can be seen as being circumscribed by rectangles. Each side of the rectangle has two parts with orientations depending on the modules over all orientation. For instance if a modules long axis is fore/aft each side will have a forward section and an aft section. What remains is to place the item in the opposing parts of the side, such as nadir-zenith (see Figure 7). Items on the ends of the rectangle are on end cones designated by the orientation of the module. So if the module is oriented fore/aft it will have both forward and aft end cones.

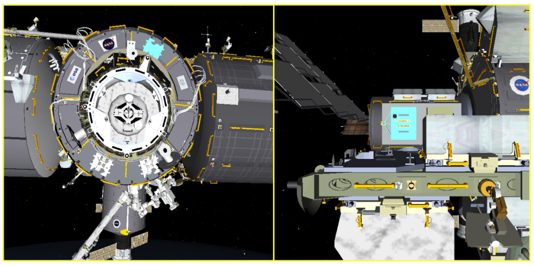


Figure 7: Sample module location specifications. The right picture is a toolbox on the Quest airlock. The airlock is oriented along the port-starboard axis and the toolbox is on the forward segment of the airlock. Thus, its location spec is ALCKfwdStb. The left picture is a heat exchanger on the forward end cone (fec) of Node 2. Its location spec is NOD2fecPrt.

An additional facility requested by the EVA and robotics teams to support equipment location was a PRONTOE to DOUG interface, the idea being that as one selects an item in PRONTOE, the DOUG camera view will orient on that item. Since we have a formal specification for object locations, it was a matter of geometry to transform the location spec into a DOUG camera view. But the Virtual Reality Laboratory at JSC that developed DOUG also suggested that when a user mouses over a DOUG item, the item should be displayed in PRONTOE. It happens that when the DOUG application loads, it runs a set of tool command language (TCL) scripts that can be used as the main interface machinery. One of those scripts sets up a command service that allows DOUG to receive a special set of commands from outside sources. So one simply executes an http-post with the appropriate command on the appropriate port of the machine running DOUG. The commands we use are for the camera positioning (see the upward pointing arrow in the DOUG window in Figure 4), and for fetching the name of the DOUG node under the mouse at any given time, (see the downward pointing arrow in the DOUG window in Figure 4). When a user selects an item in PRONTOE, if the DOUG application is available, PRONTOE will trigger DOUG to fly to the object and show it from a commensurate standoff position. This is useful when the EVA/ROBO user is planning approaches to the equipment for extraction and repairs. When a user mouses over an object in the DOUG interface, that item will be highlighted in the PRONTOE interface along with its position in the ISS taxonomy. This is useful when one desires to get detailed information about an item displayed in DOUG.

4 Using Ontological Reasoners

One of the advantages of having a formal ontology is that we can invoke a reasoner to keep the data consistent and to infer additional relationships among the objects, given the ones asserted by the user. We use a reasoner in PRONTOE to execute axioms and constraints, which constitute an important type of information included in our ontology that we collectively 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 as the loss of fluid flow when a pump loses power. While we currently author our axioms in the OWL API directly, PRONTOE will support user authoring of axioms as they pertain to the class/relations information, which is the basis of preconditions and effects authored with the PPW. We are using SWRL (OWG SWRL), 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 [9]). PRONTOE currently uses the Pellet reasoner (http://clarkparsia.com/pellet/).

But we can use the reasoner to make the editing of a set of objects less tedious. For example, various classes of remote power controller modules (RPCMs) are restricted in the number of Remote Power Controllers (RPCs) they can manage, a data property we call hasMaxRPCs. Executing the reasoner over such classes will automatically populate the hasMaxRPCs property of all the instances appropriately. So if because of a design change one were to change that value of the class from 4 to 6, running the reasoner would preclude the user from having to update all the instances of that class.

Another aspect of a formal ontology is being able to designate relations that are inverses of each other. For instance, if an Orbital Replacement Unit (ORU) isContainedBy an ORU bag then conversely that ORU bag contains that ORU. The two relations are inverses of each other. We need both relations. The Contains relation allows us to ascertain the contents of a given container with one query. But for planning applications we can immediately locate a tool with the isContainedBy relation rather than search through all the toolboxes for the one has that tool. As another example, suppose our ISS computers flight controller has changed the computer that controls two heat exchangers. She does this by editing the isControlledBy property of the heat exchangers. Without the concept of inverses she must then add two suppliesC&DHto properties to the computer item. But because suppliesC&DHto is an inverse of isControlledBy, she can run the reasoner to establish the new relations for the computer. Figure 8 shows the added relationships for a DDCU after running our Pellet reasoner. Compare with Figure 4. In particular, users of the other kernels, such as the thermal control system (TCS), assert that their various equipments have power channels, which in turn have electrical components in the electrical power system (EPS) kernel. The reasoner infers the inverse of those relations to the EPS items, e.g., hasDDCU(powerChannelS01A_A_02, DDCU-E_3) and inPowerChannel(DDCU-E_3, powerChannelS01A_A_02).

5 Knowledge Engineering

During our first year effort, we designed a set of capabilities for PRONTOE that were validated by our SMEs. After the initial version of the prototype was complete we had several detailed one-on-one sessions with specific flight controllers that generated a number of modifications described in this paper. Flight controllers of all disciplines desired some modifications, such as incremental search. Another modification concerned the optional display of roles. The items in the core systems are generally referred to by their roles. For instance,

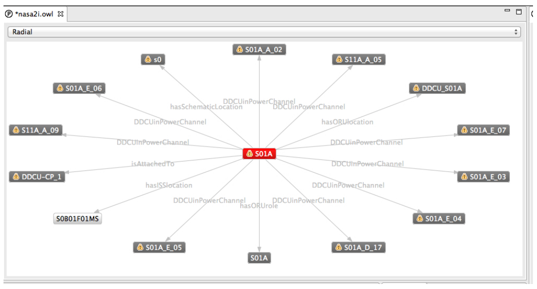
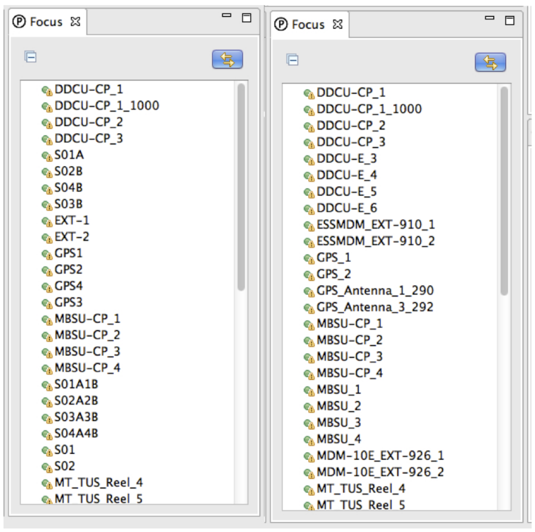


Figure 8: DDCU S01A after running the reasoner. In Figure 4 this DDCU had the relations hasORUrole, isAttachedTo and hasSchematicLocation. After running the reasoner we see it now has a physical location specification. This is because it was attached to a cold plate (DDCU-CP_1) that was in turn attached to a mount that had a physical location. SWRL rules mapped that location to both the cold plate and the DDCU. Many pieces of equipment have power channels that have DDCUs providing the voltage conversion. Here we see now all the channels associated with this DDCU. This kind of view makes it easier to see what parts of the ISS will be affected if a piece of electrical equipment fails.


Focus

DDCU-CP_1
DDCU-CP_1_1000
DDCU-CP_2
DDCU-CP_3
S01A
S02B
S04B
S03B
EXT-1
EXT-2
GPS1
GPS2
GPS4
GPS3
MBSU-CP_1
MBSU-CP_2
MBSU-CP_3
MBSU-CP_4
S01A1B
S02A2B
S03A3B
S04A4B
S01
S02
MT_TUS_Reel_4
MT_TUS_Reel_5

Focus

DDCU-CP_1
DDCU-CP_1_1000
DDCU-CP_2
DDCU-CP_3
DDCU-E_3
DDCU-E_4
DDCU-E_5
DDCU-E_6
ESSMDM_EXT-910_1
ESSMDM_EXT-910_2
GPS_1
GPS_2
GPS_Antenna_1_290
GPS_Antenna_3_292
MBSU-CP_1
MBSU-CP_2
MBSU-CP_3
MBSU-CP_4
MBSU_1
MBSU_2
MBSU_3
MBSU_4
MDM-10E_EXT-926_1
MDM-10E_EXT-926_2
MT_TUS_Reel_4
MT_TUS_Reel_5

Figure 9: Optional Display of Roles. The shot on the left shows the Focus view of all the items on the S0 truss displayed by role if present. But the roles make it difficult to see the different classes of equipment on the truss. The right picture shows the same display with the role display option turned off, and one can see more clearly the classes of equipment listed.

DDCU-E_3 has the role of S01A, that is, by virtue of where its plugged into the station EPS, it provides the DC-to-DC conversion for the 1A power bus on the S0 truss. Core systems, EVA & Robotics all refer to the item to be serviced by their roles. And yet, sometimes EVA needs to be able to count the number and types of equipment at a given location, and the role designation makes that difficult (see Figure 9). So we added a preference that will allow items to be displayed without their roles.

As mentioned earlier, EVA and robotics flight controllers have an understandably different workflow than the core system operators. As such most of the major modifications of the displays came from the former. The use of the ISS schematic, the Focus view, the DOUG screenshots and being able to build specific perspectives came from sessions with the EVA team. But for them, tying PRONTOE to DOUG a tool they use regularly for planning EVAs was the modification that made PRONTOE most useful to them; so much so that this summer we will be training EVA team members on the use of PRONTOE to support EVA planning.

6 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 [4] and [12], and the work of Biundo and Stephan [2] on a domain modeling tool that supports incremental, modular model development by extending and refining existing models. PRONTOE on the other hand removes the need for an AI scientist middleman. Work on meta-theories, e.g. [6], 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 [11] 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 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 [10], but as late as 2007, the majority of OE researchers still did not use any methodology [3]. Yet, most OE research accepts as fundamental the need for an efficient, consistent paradigm for knowledge engineering ontologies [14]. The work on developing flight rules by Barreiro et al [1] is directly complementary to our work on directives, though in that project the developed flight rules were not cast into an ontological framework but into a specialized representation to be used by planners and schedulers.

7 Summary and Future Work

PRONTOE has shown to be an authoring framework that provides 1) an ontological representation of domain information in a standard format that can be used by NASA's developing AI 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 databases into the user-constructed ontology. Our future efforts will support the EVA flight control teams desire to use PRONTOE as the single locus of the change data for ORUs. So we will be adding several structures to our ontology. First, we will design and develop a dialog for an EVA activity. An EVA activity will contain information on which items in the ontology will be affected in location or device settings by the activity. We will then group sets of these activities into an EVA Task List, which essentially predicts how the data in the ontology will change as a result of executing these tasks. Once the actual EVA is completed, flight controllers can check off those activities that were completed, and can PRONTOE automatically post those changes to the ExCATT and IMS databases.

References

[1] J. J. Barreriro and J. Chachere. Constraint and flight rule management for space mission operations. In *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, Sapporo Japan, 2010.

[2] S. Biundo and W. Stephan. System assistance in structured domain model developments. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, 1997.

[3] Jorge Cardoso. The semantic web vision: Where are we? *IEEE Intelligent Systems*, 22(5):84–88, 2007.

[4] S. Fernandez and D. Borrajo. PLTOOL: A knowledge engineering tool for planning and learning. *The Knowledge Engineering Review*, 22:1–24, 2007.

[5] Maria Fox and Derek Long. Pddl2.1: An extension to pddl for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20:2003, 2003.

[6] A. Herzig and I. Varzincak. Metatheory of action: Beyond consistency. *Artificial Intelligence*, 171:951–984, 2007.

[7] Michel Izygon, David Kortenkamp, and Arthur Molin. A procedure integrated development environment for future spacecraft and habitats. In *Proceedings of the Space Technology and Applications International Forum (STAIF)*, 2008. Available as American Institute of Physics Conference Proceedings Volume 969.

[8] David Kortenkamp, R. Peter Bonasso, and Debra Schreckenghost. A procedure representation language for human spaceflight operations. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2008.

[9] O.C. Library. *Hermit Reasoner*. PhD thesis, http://www.hermit-reasoner.com, 2011.

[10] F. M. Lopez. Overview of methodologies for building ontologies. In *Proceedings of the IJCAI Workshop on Ontologies and Problem-Solving Methods*, 1999.

[11] Karen Myers. Domain metatheories: Enabling user-centric planning. In *Proceedings of the AAAI Workshop on Representational Issues for Real-World Planning Systems (AAAI Technical Report WS-00-07)*, 2000.

[12] R.M. Simpson. Structural domain definition using GIPO IV. *The Knowledge Engineering Review*, 22:117–134, 2007.

[13] David Smith and W. Cushing. The anml language. In *In Proceedings of the 9th International Symposium on Artificial Intelligence, Robotics and Automation for Space (i-SAIRAS)*, 2008.

[14] Andrey Soares and Frederico Fonseca. Building ontologies for information systems: What we have, what we need. In *Proceedings of iConference*, 2009.