Proceedings of the 7th International Protege Conference, Bethesda, MD, July 6 - 8, 2004.

Experiences in Developing an Intelligent Ground Vehicle (IGV) Ontology in Protégé[1]

Craig Schlenoff, Tony Barbera (NIST){craig.schlenoff, tony.barbera} @nist.gov
Randy Washington, (DCS Corporation) rwashington@dcscorp.com

1.0 Introduction

The level of automation in ground combat vehicles being developed for the Army's objective force is greatly increased over the Army's legacy force. This automation is taking many forms in emerging ground vehicles; varying from operator decision aides to fully autonomous unmanned systems. The development of these intelligent ground vehicles requires a thorough understanding of all of the intelligent behavior that needs to be exhibited by the system so that designers can allocate functionality to humans and or machines. Tradition system specification techniques focused heavily on the functional description of the major systems of a vehicle and implicitly assumed that a well-trained crew would operate these systems in a manner to accomplish the tactical mission assigned to the vehicle. In order to allocate some or all of these intelligent behaviors to machines in future ground vehicles it is necessary to be able to identify and describe these intelligent behaviors in detail.

For several years, DCS Corporation has been supporting the U.S. Army Tank Automotive Research, Development and Engineering Center (TARDEC) in the exploration of approaches to model the ground vehicle domain with explicit representation of intelligent behavior. This exploration included the analysis of modeling languages (i.e., UML, DAML, OWL) as well as reference architectures. During this same period the National Institute of Standards and Technology (NIST) has been applying its Real-time Control System (RCS) reference architecture to unmanned vehicle systems and has also been investigating various modeling techniques to describe RCS designs. Recognizing the utility of RCS methodology for identifying and describing intelligent behavior and a common interest in evaluating ontologies as an approach to modeling systems in this domain, TARDEC initiated a joint effort with DCS and NIST to prototype an ontology for the intelligent ground vehicle domain.

In this paper, we describe the joint effort currently being performed by DCS Corporation and NIST to develop an intelligent ground vehicle (IGV) ontology using Protégé. The goal of this effort is to develop a common, implementation-independent, extendable knowledge source for researchers and developers in the intelligent vehicle community that will:

- Provide a standard set of domain concepts along with their attributes and inter-relations
- Allow for knowledge capture and reuse
- Facilitate systems specification, design, and integration , and
- Accelerate research in the field.

This paper describes the methodology we have used to capture knowledge in this domain and approaches we have tried to store and visualize that knowledge using Protégé 2000[2].

2.0 The 4D/RCS Methodology

4D/RCS was developed by NIST for the control of intelligent systems, and has recently been used to control intelligent vehicles within military environment. [1]. The RCS methodology concentrates on the task decomposition as the primary means of understanding the knowledge required for intelligent control. This approach begins with the knowledge "mining" activities to retrieve knowledge from subject matter experts (SMEs). The gathering and formatting of this knowledge can be summarized in six steps, each of which follows from the knowledge uncovered by the previous steps:

[1] This work is funded by U.S. Army Tank Automotive Research Development and Engineering Center (TARDEC).
[2] Certain software tools are identified in this paper in order to explain our research. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the software tools identified are necessarily the best available for the purpose.

1) The first step involves an intensive analysis of domain knowledge from manuals and SMEs, especially using scenarios of particular subtask operations. The output of the effort is a structuring of this knowledge into a task decision tree form of simpler and simpler commands (actions/verbs) at simpler and simpler levels of task description.
2) This step defines the hierarchical organization of agent control modules that will execute these layers of commands in such a manner as to reasonably accomplish the tasks. This is the same as coming up with a business or military organizational structure of agent control modules (people, soldiers) to accomplish the desired tasks. This step forces a more formal structuring of all of the subtask activities as well as defining the execution structure.
3) This step clarifies the processing of each agent's input command through the use of rules to identify all of the task branching conditions with their corresponding output commands. Each of these command decompositions at each agent control module will be represented in the form of a state-table of ordered production rules (which is an implementation of an extended finite state machine (FSM)). The sequence of simpler output commands required to accomplish the input command and the named situations (branching conditions) that transition the state-table to the next output command are the primary knowledge represented in this step.
4) In this step, the above named situations that are the task branching conditions are defined in great detail in terms of their dependencies on world and task states. This step attempts to define the detailed precursor states of the world that cause a particular situation to be true.
5) In this step, we identify and name all of the objects and entities together with their particular features and attributes that are relevant to defining the above world states and situations.
6) The last step is to use the context of the particular task activities to establish the distances and, therefore, the resolutions at which the above objects and entities must be measured and recognized by the sensory processing component. This step establishes a set of requirements and/or specifications for the sensor system at the level of each separate subtask activity.

More details about this methodology can be found in [2].

3.0 Representing Tactical Behavior Knowledge in Protégé

For our initial work, we performed a task decomposition on a Scout Troop performing a tactical road march to an assembly area. To evaluate the use of ontologies in representing this knowledge we have decided to focus on the representation of the situation command pairs captured in step 4 above. Although the 4D/RCS approach uses state tables to represent the information, the representation within the ontology does not necessarily need to be captured within state tables as long as no information is lost. Table 1 shows a small piece of a state table representation that was developed from the 4D/RCS methodology for the "Conduct Tactical Road March to Assembly Area" scenario. The left column shows the condition that must be true for an action to occur. The notation S# notates a state value. For example, in the first line, the system must be in state 4 and the condition "Scout Platoon Ready to Conduct Route Recon" must be true for the action "Scout Platoon Conduct Route Reconnaissance" to occur. When the action is executing, the system will change over to state 5.

Conduct Tactical Road March To Assembly Area	
Condition	**Action**
…	…
S4 Scout Platoon Ready to Conduct Route Recon	S5 Scout Platoon Conduct Route Reconnaissance
S5 Quartering Party Ready to Organize Assembly Area	S6 Quartering Party Follow Recon Platoon to Assembly Area
S6 Quartering Party Clear of Start Point	S7 Main Body and Trail Party Prepare for Road March
S7 Main Body and Trail Party Recon to Start Point Done	S8 Troop Prepare Detailed Movement Plans
S8 Scout Platoon Route Recon Done	S9 Scout Platoon Establish Assembly Area Security
S9 Quartering Party at Release Point	S10 Quartering Party Conduct Area Recon of Assembly Area
S10 Quartering Party Area Recon of Assembly Area Done	S11 Quartering Party Organize Assembly Area
S11 Quartering Party Status Assembly Area Suitable	S12 First Main Body Unit Move Into Road March Formation

S12 Main Body Unit At Start Point	S12 Main Body Unit Execute Tactical Road March Next Main Body Unit Move Into Road March Formation
S12 Last Main Body Unit at Start Point	S13 Main Body Unit Execute Tactical Road March Trailing Party Move Into Road March Formation
…	…

Table 1: Excerpt from a 4D/RCS State Table

The two most important requirements in representing the information is that 1) no information gets lost and 2) information is captured in a way so that a knowledge engineer who is modifying or extending the information can logically find the information they are looking for. Throughout this work, we have recognized that there are many ways of representing information in a format such that no information gets lost. As such, we describe a few of these approaches below and with an emphasis on the advantages of the knowledge organization considering the structure that the knowledge engineering may wish to view it.

Initially, we have focused on representing the actions (the right column). A few of the approaches we have explored are described below. Note that in all cases, we used the OWL plug-in within Protégé to serve as a framework for representing the information.

- **Organizing actions by who is executing them:** In this approach, we structure the ontology around the agent who is executing the action. There is a class in the ontology called "Quartering Party Actions" which include the actions "Follow Recon Platoon to Assembly Area", "Conduct Area Recon of Assembly Area", "Organize Assembly Area", etc. In this approach, we organize the ontology to emphasize the agent who is performing the activity as opposed to the type of activity that is being performed.
- **Organizing actions by an abstract look as to why the action is being performed:** In this approach, we structure the ontology based upon the abstract goal that the action is attempting to accomplish. For example, we may have a class called *MovementCommands* which contain commands such as "Follow Recon Platoon to Assembly Area" and "Move Into Road March Formation" in its sub-structure.
- **Represent actions as services:** In this approach, we use the Web Ontology Language – Services (OWL-S) as the underlying representation for actions. In essence, we represent the control structure as a set of agents who can provide services, and use the OWL-S data structures to describe the services that they can provide.

4.0 Conclusions and Future Work

This paper describes the rationale and some very preliminary results of work being carried out by DCS Corporation and NIST in support of the TARDEC IGV Taxonomy effort. Though this work is just getting started, a number of issues have arisen that would play a strong impact on how we move forward. Some of these issues include:

- How do we expect this ontology will be used? Do we expect the ontology to be incorporated within the intelligent vehicle and extended as necessary, or do we expect that this would serve as a "reference knowledge structure" that would simply provide an example of how such a system could be built?
- Within this example, when should we use classes and when should we use instances? Should instances only be used during implementation time?
- Should we tie in with an upper ontology? We are exploring utilizing SUMO [3], but have not yet identified that value that it brings to this domain. What value does an upper ontology bring to an IGV ontology and does it justify the extra effort?
- How conformant should we be with standards (such as OWL-S)?
- How will the data structures that we explored scale when different types of information are added (e.g., conditions, world model objects, etc?

These questions, along with many others, will be explored and addresses as the project progresses.

References

1. Albus, J., "4-D/RCS: A Reference Model Architecture for Demo III," *NISTIR 5994*, Gaithersburg, MD, 1997.

2. Barbera, T., Messina, E., Huang, H., Schlenoff, C., and Balakirsky, S., "Software Engineering for Intelligent Control Systems," *To appear in the Kunstliche Intelligenz Journal: Special Issue on Software Engineering for Knowledge-based Systems*, 2004.

3. Niles, I. and Pease, A., "Towards a Standard Upper Ontology," *Proceedings of the 2nd Internal Conference On Formal Ontology in Information Systems (FOIS-2001)*, 2001.