Experiences in Developing An Intelligent Ground Vehicle (IGV) Ontology In Protégé

Craig Schlenoff, NIST
Randy Washington, DCS Corporation
Tony Barbera, NIST

July 8, 2004
Agenda

• Background:
  – What is an Intelligent Ground Vehicle (IGV)?
  – NIST 4D/RCS Methodology and Architecture

• Ontology Development:
  – 4D/RCS to Ontology Mapping
  – Interchange Formats and Upper Ontologies
  – IGV Military Equipment
  – IGV Behaviors
  – IGV Conditions

• Current Status

• Issues and Lessons Learned
What is an Intelligent Ground Vehicle?
Interchange Formats and Upper Ontologies

• **OWL**
  - Neutral (W3C) interchange format
  - XML base enables use XSLT transforms
  - Provides access to emerging semantic web technologies

• **OWL-S**
  - Rich semantics for describing complex processes (without being too complicated)
  - Well suited to agent architectures

• **Pieces of SUMO (Suggested Upper Merged Ontology)**
  - Class structure and properties provide a good starting point for developing domain specific ontology
  - Native KIF format too complex for target community and not necessary for requirements capture

• **Namespaces**
  - Used quite a bit to make ontology more manageable
IGV Conceptual Model

External Service Request by a process

Troop Commander Agent → Platoon Leader Agent → Section Lead Agent → Vehicle Commander Agent → Lethality System Agent → Surveillance System Agent → Mobility System Agent → Survivability System Agent → Support System Agent

Vehicle Commander Agent → Propulsion Subsystem Agent → Transmission Component → Track-Drive Component

Automotive Subsystem Agent → Auxiliary Subsystem Agent

Brake Component → Engine Component
Representing an IGV (cont.)
### StartUpAndOperate

<table>
<thead>
<tr>
<th>New StartupAndOperateCommand</th>
<th>S1 proc_StartEngine</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1 EngineStarted</td>
<td>S2</td>
</tr>
<tr>
<td>S2 GearChangeRequired</td>
<td>S3 proc_ChangeGear</td>
</tr>
<tr>
<td>S3 GearChanged</td>
<td>S2</td>
</tr>
<tr>
<td>S2 NewCommandedVelocity</td>
<td>S4 proc_AdjustEngineThrottle</td>
</tr>
<tr>
<td>S4 EngineThrottleAdjusted</td>
<td>S2</td>
</tr>
<tr>
<td>S2 ShutDownRequested</td>
<td>S5 proc_SetGearToPark</td>
</tr>
<tr>
<td>S5 GearInPark</td>
<td>S6 ShutDownEngine</td>
</tr>
<tr>
<td>S6 EngineShutDown</td>
<td>S0 Done</td>
</tr>
</tbody>
</table>

#### Input Conditions
- StartUpAndOperate
- New StartupAndOperateCommand
- S1 EngineStarted
- S2 GearChangeRequired
- S3 GearChanged
- S4 EngineThrottleAdjusted
- S2 ShutDownRequested
- S5 GearInPark
- S6 EngineShutDown

#### Output Commands
- S1 proc_StartEngine
- S2
- S3 proc_ChangeGear
- S2
- S4 proc_AdjustEngineThrottle
- S2
- S5 proc_SetGearToPark
- S6 ShutDownEngine
- S0 Done
Representing a Propulsion Service
Propulsion Service Graph
More Visualization Features
Conditions

- Commanded Velocity Positive (AND) Gears In Reverse
- Commanded Velocity Negative (AND) Gears In Forward
- Commanded Velocity Positive But Gear Is In Reverse (or)
- Commanded Velocity Negative But Gear Is In Forward
- Gear Change Required

Table:

<table>
<thead>
<tr>
<th>Input Conditions</th>
<th>Output Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0 EngineStarted</td>
<td>S2</td>
</tr>
<tr>
<td>S2 GearChangeRequired</td>
<td>S3 proc_ChangeGear</td>
</tr>
<tr>
<td>S3 GearChanged</td>
<td>S2</td>
</tr>
<tr>
<td>S2 NewCommandedVelocity</td>
<td>S4 proc_AdjustEngineThrottle</td>
</tr>
<tr>
<td>S4 EngineThrottleAdjusted</td>
<td>S2</td>
</tr>
<tr>
<td>S2 ShutDownRequested</td>
<td>S5 proc_SetGearToPark</td>
</tr>
<tr>
<td>S5 GearInPark</td>
<td>S6 ShutDownEngine</td>
</tr>
<tr>
<td>S6 EngineShutDown</td>
<td>S0 Done</td>
</tr>
</tbody>
</table>
IGV Condition Example
Model Development Status

• OWL entities defined
  – Classes 175
  – Properties 130
  – Instances 700
Issues and Lessons Learned

• Developing an ontology is a slow iterative process
  – It difficult to evaluate a model construct without inputting detail.
  – It is very difficult to change the model once you have entered any level of detail.
• Difficult to develop consistent rules for when to use a Classes vs. an Instance in a large domain
  – Is knowledge in class restrictions or instances?
• Difficult to present large models to domain experts
• Experiences with OWL-S shows that it has applications outside of the semantic web.
  – Would like to get involved in its development