



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



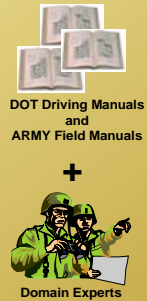
What is an Intelligent Ground Vehicle?





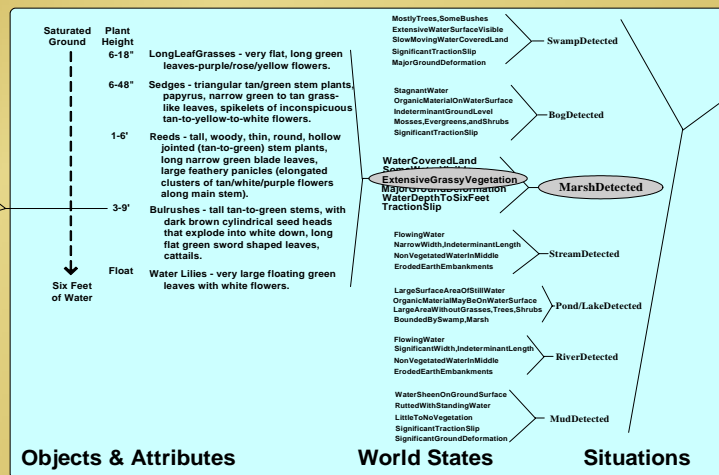
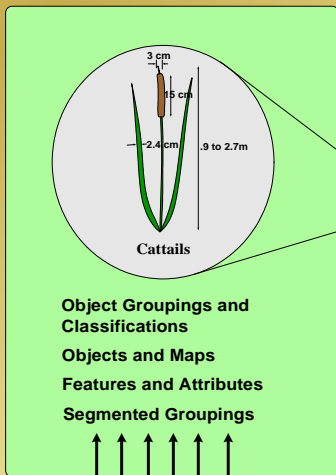
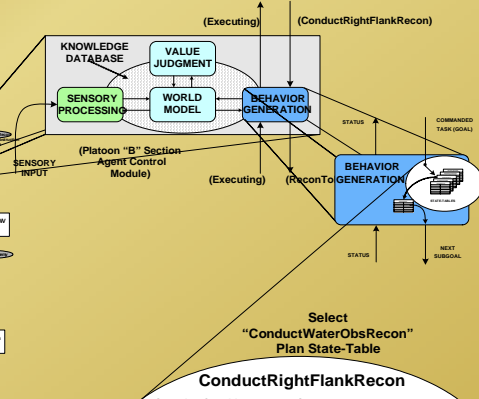
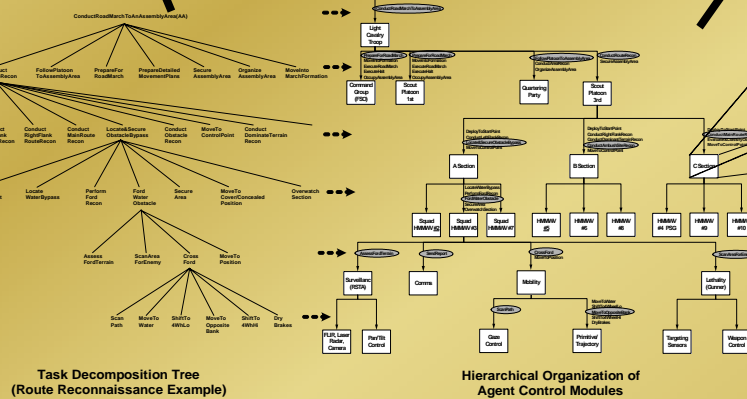
4D/RCS Mapping to IGV

Ontology



Service Instances

Commander & Vehicle Agent Instances



Select "ConductWaterObsRecon" Plan State-Table

ConductRightFlankRecon

Selection Conditions	Selected Plan
NormalRouteReconSituation	ConductRightFlankRecon
WaterObstacleDetected	ConductWaterObstacleRecon
WaterObstacleDetected	AssessMinifiedRecon
WaterObstacleDetected	ConductWaterObstacleRecon
WaterObstacleDetected	ConductWaterObstacleRecon

PLAN SELECTION TABLE

ConductWaterObstacleRecon

NewPlan	S1	S2	S3	S4	S5	S6	S7
SetupReport&NewControlMeasures	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank
ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank
ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank
ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank
ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank
ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank
ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank	ReconToRouteFlank

Input Conditions Output Commands

PLAN STATE-TABLE

Process Model Instances

Environment Instances

WORLD MODEL KNOWLEDGE

Condition Instances

BEHAVIOR GENERATION

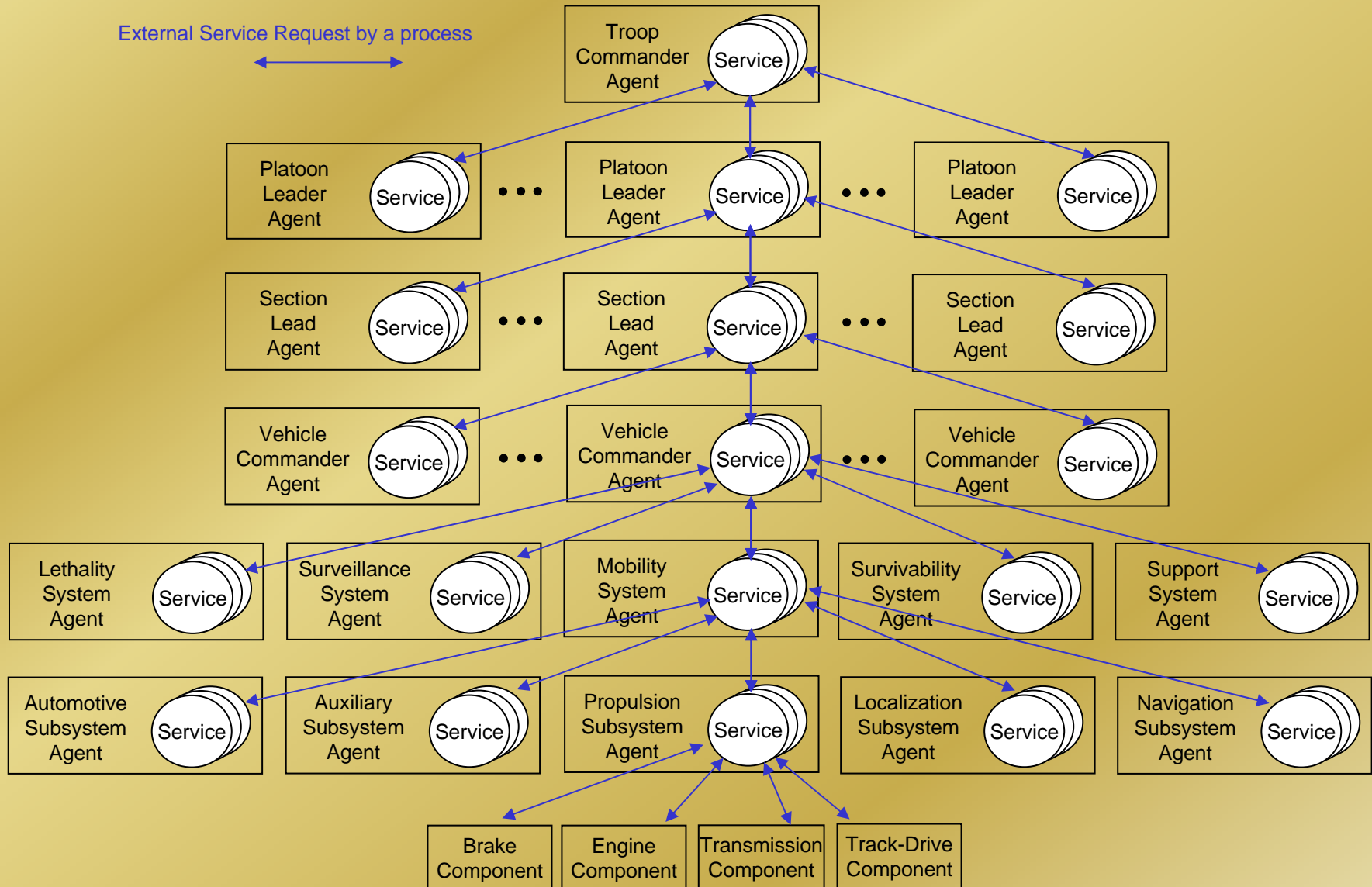


Interchange Formats and Upper Ontologies



- 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



igvmilitaryequipment Protégé 2.1 (file:\C:\Owl\igvmilitaryequipment.pprj, OWL Files)

Project Edit Window OWL Wizards Help

OWLClasses Forms Individuals Metadata Properties OWLS-Tab

Subclass Relationship

Asserted Hierarchy

- owl:Thing
 - rdf:List
 - Equipment
 - CommunicationsDevice
 - EngineeringComponent
 - EngineeringConnection
 - IGVEngineeringComponent
 - IGVComponent
 - IGVSubsystem
 - CommunicationsSubsystem
 - LethalitySubsystem
 - MobilitySubsystem
 - AutomotiveSubsystem
 - AuxiliarySubsystem
 - LocalizationSubsystem
 - NavigationSubsystem
 - PropulsionSubsystem
 - SupportSubsystem
 - SurveillanceSubsystem
 - SurvivabilitySubsystem
 - IGVSystem
 - CommunicationsSystem
 - LethalitySystem
 - MobilitySystem**
 - SupportSystem
 - SurveillanceSystem
 - SurvivabilitySystem
 - IntelligentGroundVehicle

- Weapon
- service:ServiceResource
 - IGVComponent
 - igvbehavior:IGVAgent
 - IGVAgentComponent
 - AutomotiveAgent
 - AuxiliaryAgent
 - CommunicationsAgent
 - LethalityAgent
 - LocalizationAgent
 - MobilityAgent
 - NavigationAgent
 - PropulsionAgent
 - SupportAgent
 - SurveillanceAgent
 - SurvivabilityAgent
 - VehicleAgent
- service:Service
- service:ServiceModel

MobilitySystem (type=owl:Class)

Name: MobilitySystem

rdfs:comment

Annotations

Property	Value	Lang
owl:versionInfo	IGV 1.0 RTW	

Asserted Inferred

Asserted Conditions

NECESSARY & SUFFICIENT

- IGVSystem
- \exists engineeringSubcomponent MobilityAgent
- \exists engineeringSubcomponent NavigationSubsystem
- \exists engineeringSubcomponent AutomotiveSubsystem
- \exists engineeringSubcomponent AuxiliarySubsystem
- \exists engineeringSubcomponent LocalizationSubsystem
- \exists engineeringSubcomponent PropulsionSubsystem
- engineeringSubcomponent = 6
- \exists maxCapabilityMeasure igvbasic:PowerMeasure

NECESSARY

Properties at Class

- engineeringSubcomponent
- maxCapabilityMeasure
- averageCapabilityMeasure

Disjoints



Tactical Behaviors

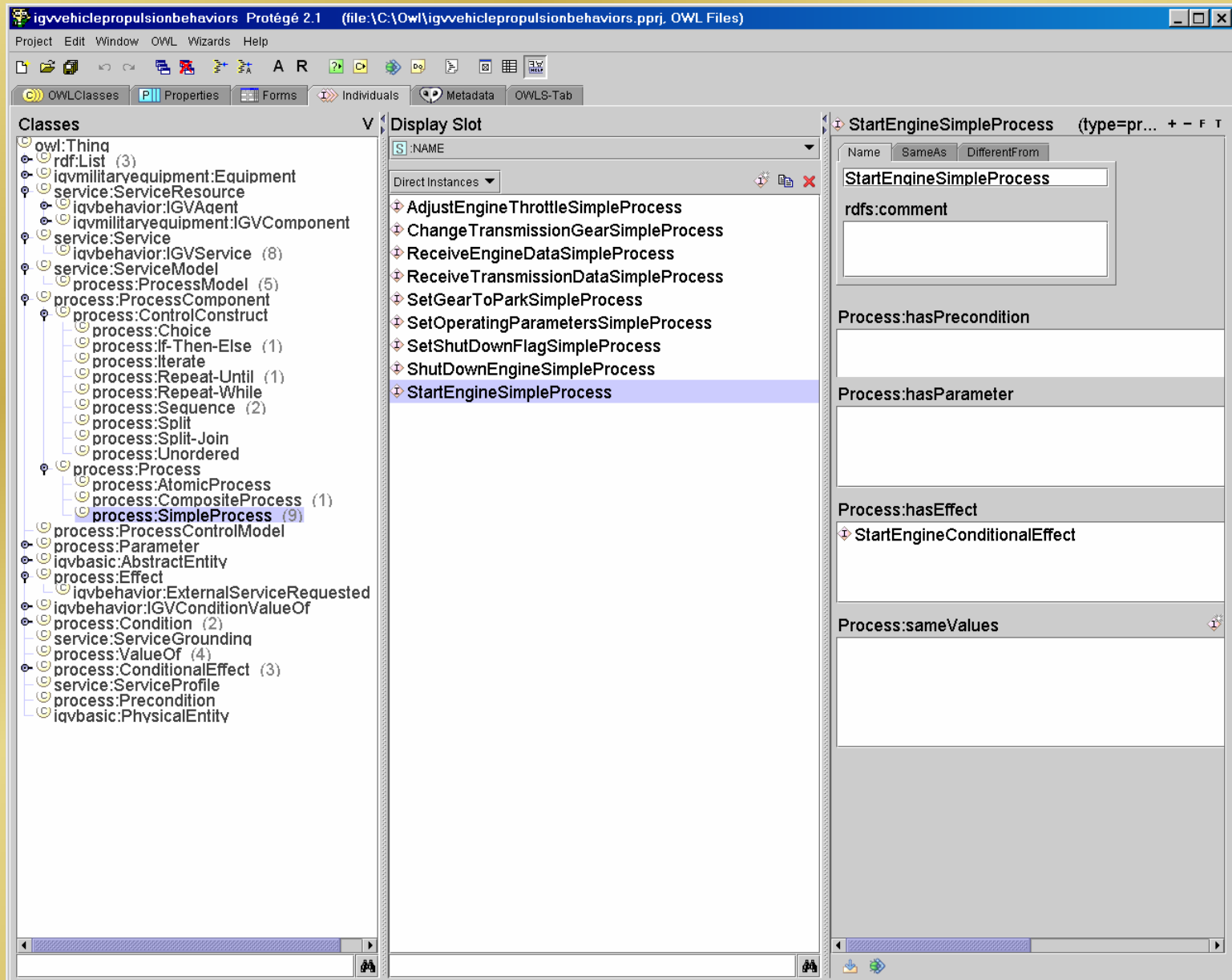
Plan State-Table Selection



StartUpAndOperate	
New StartupAndOperateCommand	S1 proc_StartEngine
S1 EngineStarted	S2
S2 GearChangeRequired	S3 proc_ChangeGear
S3 GearChanged	S2
S2 NewCommandedVelocity	S4 proc_AdjustEngineThrottle
S4 EngineThrottleAdjusted	S2
S2 ShutDownRequested	S5 proc_SetGearToPark
S5 GearInPark	S6 ShutDownEngine
S6 EngineShutDown	S0 Done

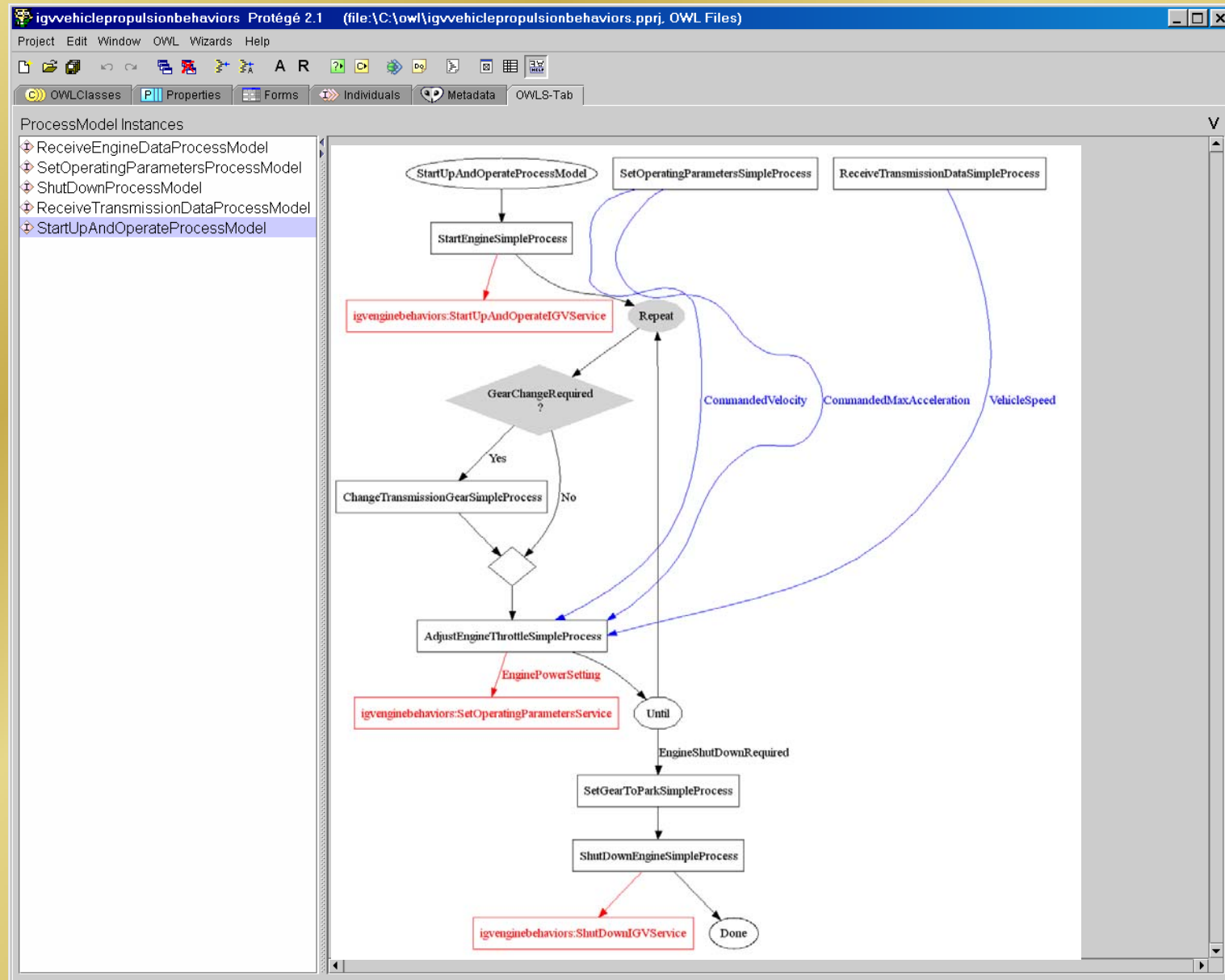
Input Conditions

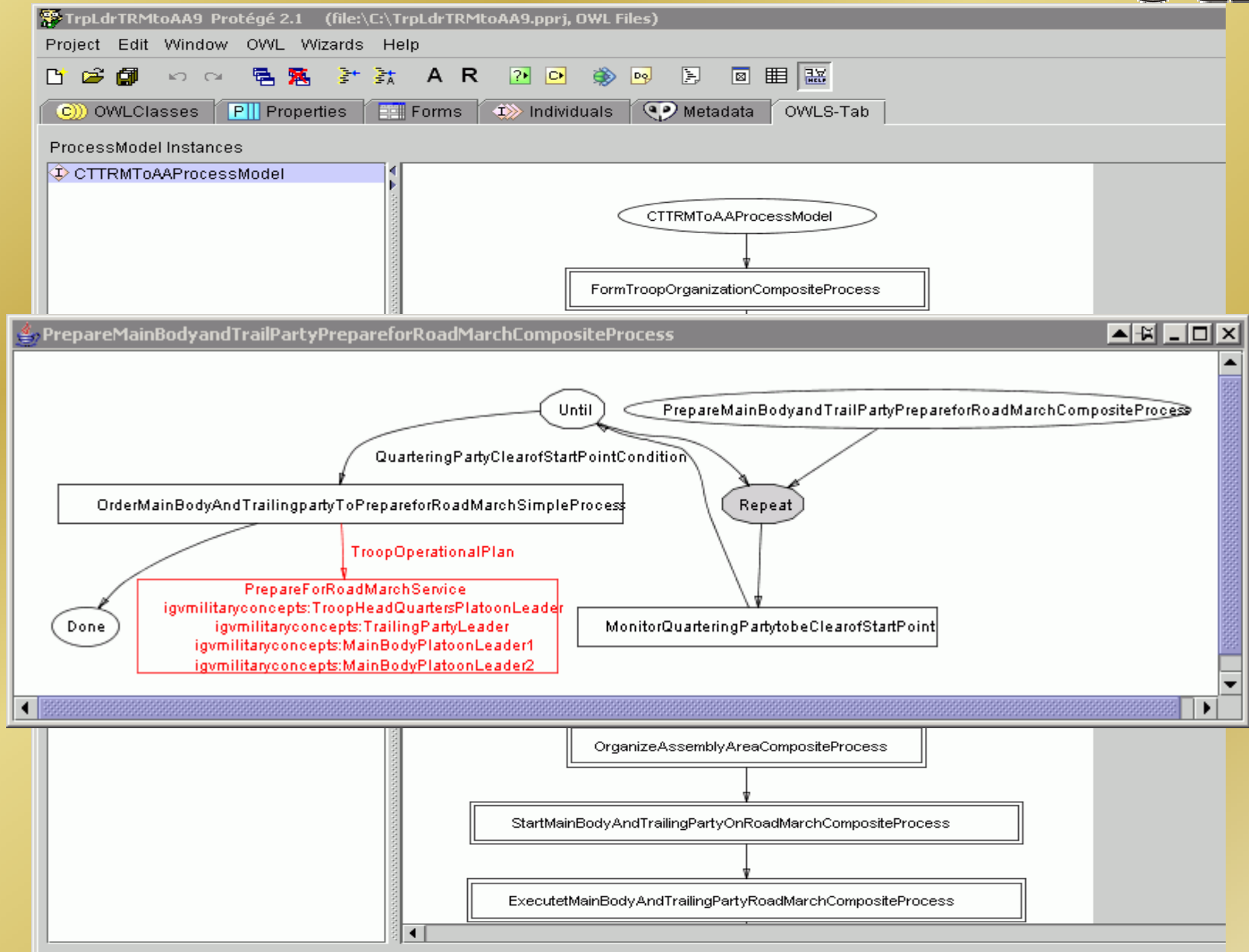
Output Commands



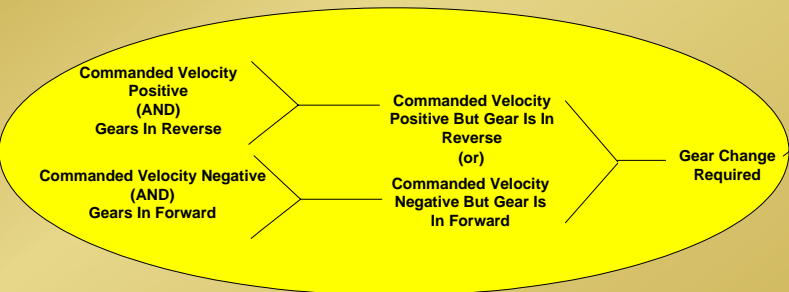
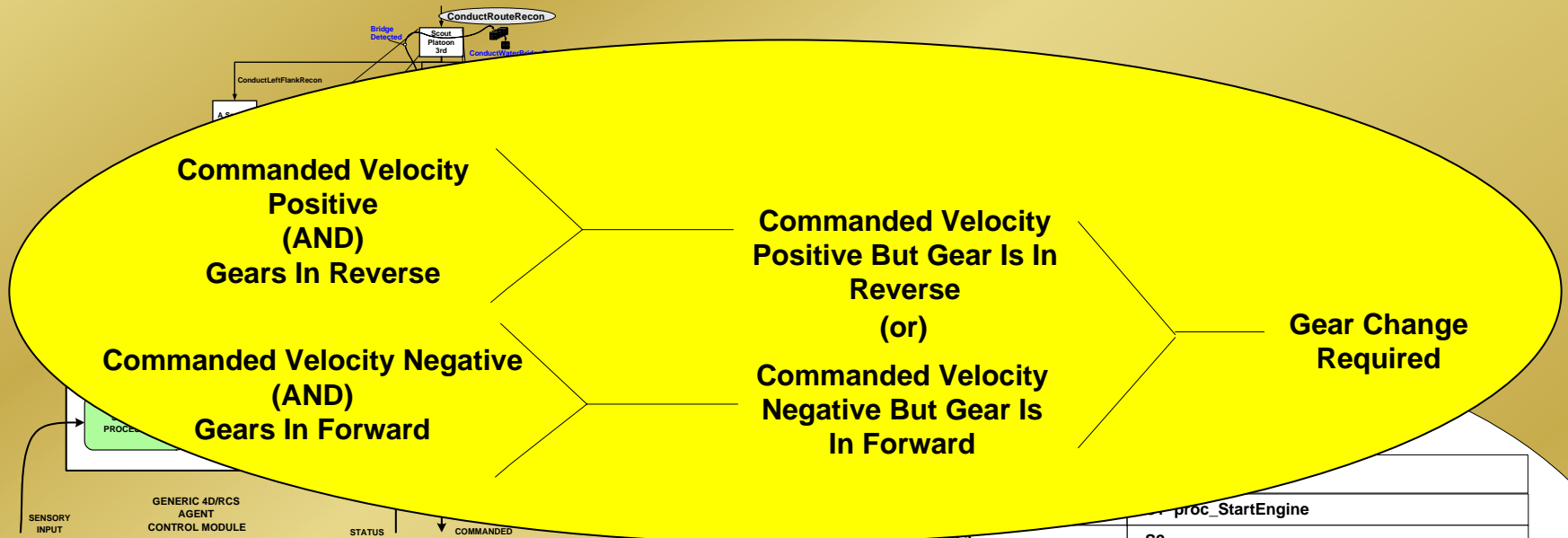
The screenshot shows the Protégé 2.1 interface with the following components:

- Title Bar:** igvvehiclepropulsionbehaviors Protégé 2.1 (file:\C:\Owl\igvvehiclepropulsionbehaviors.pprj, OWL Files)
- Menu Bar:** Project Edit Window OWL Wizards Help
- Toolbar:** Includes icons for file operations, navigation, and editing.
- OWLClasses Tab:**
 - Classes Panel (Left):** A hierarchical tree of classes. The selected class is `process:SimpleProcess` (9), which is a subclass of `process:ProcessComponent`. Other visible classes include `owl:Thing`, `rdf:List` (3), `iqvmilitaryequipment:Equipment`, `service:ServiceResource`, `iqvbehavior:IGVAgent`, `iqvmilitaryequipment:IGVComponent`, `service:Service`, `iqvbehavior:IGVService` (8), `process:ProcessModel` (5), `process:ProcessComponent`, `process:ControlConstruct`, `process:Choice`, `process:If-Then-Else` (1), `process:Iterate`, `process:Repeat-Until` (1), `process:Repeat-While`, `process:Sequence` (2), `process:Split`, `process:Split-Join`, `process:Unordered`, `process:Process`, `process:AtomicProcess`, `process:CompositeProcess` (1), `process:SimpleProcess` (9), `process:ProcessControlModel`, `process:Parameter`, `iqvbasic:AbstractEntity`, `process:Effect`, `iqvbehavior:ExternalServiceRequested`, `iqvbehavior:IGVConditionValueOf`, `process:Condition` (2), `service:ServiceGrounding`, `process:ValueOf` (4), `process:ConditionalEffect` (3), `service:ServiceProfile`, `process:Precondition`, and `iqvbasic:PhysicalEntity`.
 - Display Slot:** A list of instances of the selected class. The instances are: `AdjustEngineThrottleSimpleProcess`, `ChangeTransmissionGearSimpleProcess`, `ReceiveEngineDataSimpleProcess`, `ReceiveTransmissionDataSimpleProcess`, `SetGearToParkSimpleProcess`, `SetOperatingParametersSimpleProcess`, `SetShutdownFlagSimpleProcess`, `ShutdownEngineSimpleProcess`, and `StartEngineSimpleProcess` (highlighted).
- StartEngineSimpleProcess Class Details (Right):**
 - Name:** StartEngineSimpleProcess
 - rdfs:comment:** (Empty text area)
 - Process:hasPrecondition:** (Empty text area)
 - Process:hasParameter:** (Empty text area)
 - Process:hasEffect:** StartEngineConditionalEffect
 - Process:sameValues:** (Empty text area)





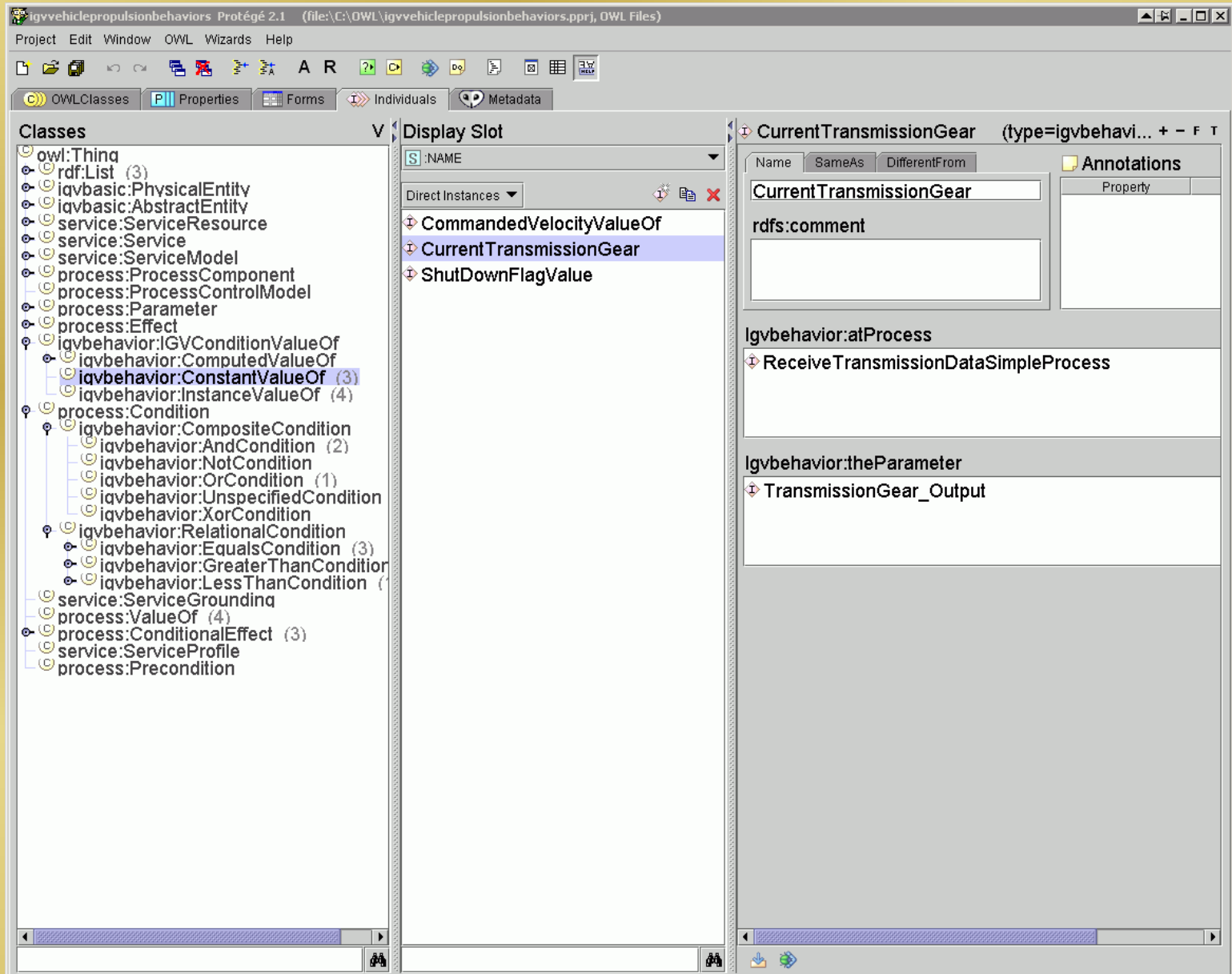
Conditions



	S1 proc_StartEngine
S1 EngineStarted	S2
S2 GearChangeRequired	S3 proc_ChangeGear
S3 GearChanged	S2
S2 NewCommandedVelocity	S4 proc_AdjustEngineThrottle
S4 EngineThrottleAdjusted	S2
S2 ShutDownRequested	S5 proc_SetGearToPark
S5 GearInPark	S6 ShutDownEngine
S6 EngineShutDown	S0 Done

Input Conditions

Output Commands



The screenshot shows the Protégé 2.1 OWL editor interface. The title bar indicates the file is "igvvehiclepropulsionbehaviors.pprj" located in "C:\OWL\igvvehiclepropulsionbehaviors.pprj, OWL Files".

Classes Pane: A hierarchical tree of classes is displayed. The selected class is `igvbehavior:ConstantValueOf (3)`. Other visible classes include `owl:Thing`, `igvbasic:PhysicalEntity`, `service:ServiceResource`, `process:ProcessComponent`, `igvbehavior:IGVConditionValueOf`, `igvbehavior:ComputedValueOf`, `igvbehavior:InstanceValueOf (4)`, `process:Condition`, `igvbehavior:CompositeCondition`, `igvbehavior:AndCondition (2)`, `igvbehavior:NotCondition`, `igvbehavior:OrCondition (1)`, `igvbehavior:UnspecifiedCondition`, `igvbehavior:XorCondition`, `igvbehavior:RelationalCondition`, `igvbehavior:EqualsCondition (3)`, `igvbehavior:GreaterThanCondition`, `igvbehavior:LessThanCondition (1)`, `service:ServiceGrounding`, `process:ValueOf (4)`, `process:ConditionalEffect (3)`, `service:ServiceProfile`, and `process:Precondition`.

Display Slot Pane: This pane shows the "Direct Instances" of the selected class. The instances listed are `CommandedVelocityValueOf`, `CurrentTransmissionGear` (which is highlighted), and `ShutDownFlagValue`.

CurrentTransmissionGear Pane: This pane displays the details for the selected instance, `CurrentTransmissionGear`. It includes a "Name" tab, a "SameAs" tab, a "DifferentFrom" tab, and an "Annotations" tab. The "Name" tab shows the name `CurrentTransmissionGear` and the comment `rdfs:comment`. The "Annotations" tab shows the property `Property`. Below the tabs, the instance is identified as `igvbehavior:atProcess` and `igvbehavior:theParameter`. The instance has a value of `ReceiveTransmissionDataSimpleProcess` and a parameter of `TransmissionGear_Output`.



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