Vision Document

For a GMoDS-based Runtime Agent Role Interpreter

Version 1.0

Submitted in partial fulfillment of the requirements of the degree of MSE

Kyle Hill
CIS 895 – MSE Project
Kansas State University
Table of Contents

1 Introduction .................................................................................................................................................. 3
   1.1 Motivation ........................................................................................................................................... 3
   1.2 Terms and Definitions ......................................................................................................................... 3
   1.3 References ........................................................................................................................................... 4
2 Project Overview ....................................................................................................................................... 5
   2.1 Project Goal ......................................................................................................................................... 5
   2.2 System Context ................................................................................................................................... 6
3 Requirements Specification ...................................................................................................................... 7
   3.1 Critical Use Cases ................................................................................................................................. 7
      3.1.1 Use Case 1: Receive Assignment ............................................................................................... 7
      3.1.2 Use Case 2: Interpret Role Level Goal Model ............................................................................. 8
      3.1.3 Use Case 3: Interpret Goal Capability Map ................................................................................. 8
      3.1.4 Use Case 4: Validate Capabilities .............................................................................................. 9
      3.1.5 Use Case 5: Execute Role Level Goal Model ............................................................................. 9
      3.1.6 Use Case 6: Perform Actions ....................................................................................................... 10
      3.1.7 Use Case 7: Report Status ........................................................................................................... 10
   3.2 Assumptions ........................................................................................................................................... 11
   3.3 Constraints ........................................................................................................................................... 11
   3.4 Environment ....................................................................................................................................... 11
1 Introduction
This document provides an overview of a Goal Model for Dynamic Systems [1] (GMoDS)-based Runtime Agent Role Interpreter. This interpreter serves as the basis for an agent architecture in an Organizational Model for Adaptive Computational Systems [2] (OMACS) multiagent system. This document will describe the critical use cases and requirements for this role interpreter, an agent architecture based on it, as well as a simple Wumpi World simulation to demonstrate this architecture.

1.1 Motivation
Designing agents using a generic role interpreter gives system designers great freedom and makes multiagent systems more flexible. By using a role interpreter, agents do not need to have roles defined at design time; rather, they can be assigned at runtime as the organization sees fit. This allows the organization to make optimizations at runtime to blend redundant roles together, or to decompose complex roles into multiple simple ones.

Adding role interpreters to agents increases system modularity by cleanly separating role definitions from the agent architectures that execute them. Agents are simply given a formal role definition that they execute. In our system, roles are defined using GMoDS goal models, allowing us to make use of its established execution semantics. By defining roles using GMoDS, we can decompose role behavior into low-level functions that can be mapped directly to agent capabilities using a goal capability map.

1.2 Terms and Definitions
1.2.1 OMACS
Organization Model for Adaptive Complex Systems is a multiagent systems architecture that defines a system in terms of an organization consisting of goals, roles, agents, capabilities, and the relationships between these entities.

1.2.2 GMoDS
The Goal Model for Dynamic Systems provides a formal definition and decomposition of system goals and the relationships between them. GMoDS provides a framework for executing a goal model within an organization.

1.2.3 Goal
A Goal is an OMACS entity that defines what the system should do. For our purposes, goals will be formally defined using GMoDS.

1.2.4 Role
A Role is an OMACS entity that defines algorithms and behaviors to achieve some goal. Roles are assigned to achieve a specific system goal at runtime.
1.2.5 Agent
An Agent is an OMACS entity that actually interacts with the system environment while playing its role to achieve some goal. Our Role Interpreter will run inside an agent instance, providing most of the agent architecture.

1.2.6 Capability
A Capability is an OMACS entity that denotes some function or ability that an agent can possess and that a role can require. Capabilities, along with other organizational policies, can be used to restrict the assignment of agents to roles.

1.2.7 RLGM
A Role Level Goal Model [3] (RLGM) is a GMoDS goal model that defines role behavior. Role Level Goal Models decompose role tasks into low-level goals that can be mapped directly to agent capabilities at runtime.

1.2.8 GCM
A Goal Capability Map [3] (GCM) provides a mapping of role level goals to agent capabilities. Our system will use a Goal Capability Map to associate leaf goals with specific agent capabilities.

1.2.9 Wumpi World
Wumpi World is simple OMACS simulation software for testing and validating agent architectures.

1.3 References
2 Project Overview

![Figure 1. Role Interpreter Project Overview](image)

The GMoDS-based Runtime Agent Role Interpreter will be integrated into an agent’s architecture. The role interpreter will take a role and a goal as input. The input role will be defined by a GMoDS role level goal model (RLGM), and a goal capability map (GCM). The RLGM will decompose role behavior into a tree structure where the leaf goals can essentially be treated as function calls. These low-level goals are mapped to agent capabilities using the role's GCM. This map provides a simple one-to-one mapping of abstract, low-level role behaviors to concrete agent capabilities.

At runtime, OMACS will assign agents to play roles to achieve goals. At this time, an agent instance will be provided with a goal instance and a role instance. The agent will execute its role following GMoDS semantics: The interpreter will non-deterministically select a goal from the set of active goals. It will then directly call that goal’s associated capability, with the specified parameters, by looking it up in the goal-capability map.

2.1 Project Goal

This project will provide an implementation of a GMoDS-based Runtime Agent Role Interpreter. This interpreter will make up the bulk of an individual agent’s architecture. To demonstrate that the Role Interpreter works correctly, this project will also provide a full agent architecture based on it, as well as a simple Wumpi World simulation.
2.2 System Context

The GMoDS-based Runtime Agent Role Interpreter will make up a large part of an individual agent’s architecture. At runtime, OMACS will assign a role and goal to an agent. That agent will play the assigned role to achieve the assigned goal. The agent must be capable of playing the assigned role, and the assigned role must achieve the assigned goal.

Figure 2. OMACS System Context
3 Requirements Specification

3.1 Critical Use Cases

![Role Interpreter Diagram]

**Figure 3. Critical Use Cases**

3.1.1 Use Case 1: Receive Assignment

**Description:** This use case describes the assignment of roles and goals to agents.

**Includes:** Interpret Role Level Goal Model, Interpret Goal Capability Map, Validate Capabilities

**Pre-Conditions:** The role must be able to achieve the assigned goal.

**Details:** The agent hosting the role interpreter is assigned a role and a goal by OMACS. The agent will play the assigned role by executing its associated RLGM to achieve the assigned goal.

**Post Conditions:** The agent is ready to play the assigned role to achieve its goal.

**Specific Requirements:**
3.1.1.1 SR1 [Critical Requirement]
The system shall accept a GMoDS RLGM and GCM as input representing the role it will play.

3.1.1.2 SR2 [Critical Requirement]
The system shall accept a GMoDS goal instance specification as input representing the goal it will achieve. The goal instance specification shall include goal type information and values for each attribute.

3.1.1.3 SR3
The system shall accept the RLGM input according to the Goal Model XML schema used by the agentTool3 software.

3.1.1.4 SR4
The system shall accept the GCM input according to an XML schema that will be used by a future version of the agentTool3 software. The GCM schema definition does not exist yet and it will be provided as part of this project.

3.1.2 Use Case 2: Interpret Role Level Goal Model

Description: This use case describes the parsing of a role’s Role Level Goal Model

Pre-Conditions: The given Role Level Goal Model is associated with the role assigned to this agent.

Details: OMACS provides a Role Level Goal Model as part of role assignment. This RLGM is a GMoDS goal model that describes the runtime behavior of the associated role.

Post Conditions: The agent is ready to execute the RLGM.

Specific Requirements:

3.1.2.1 SR 4 [Critical Requirement]
The system shall parse the input RLGM into a goal tree structure using the GMoDS parser.

3.1.3 Use Case 3: Interpret Goal Capability Map

Description: This use case describes the parsing of a role’s Goal Capability Map

Pre-Conditions: The given Goal Capability Map is associated with the role assigned to this agent. The agent possesses the capabilities required by the assigned role.

Details: OMACS provides a Goal Capability Map as part of role assignment. The GCM provides a one-to-one mapping between RLGM leaf goals and a role’s abstract capabilities. The agent’s concrete capabilities map directly to the role’s abstract capabilities at assignment.

Post Conditions: The agent can map RLGM leaf goals to specific capabilities.

Specific Requirements:
3.1.3.1 SR5 [Critical Requirement]
The system shall parse the input GCM into a map structure with leaf-level goals serving as keys, and concrete agent capabilities serving as values.

3.1.4 Use Case 4: Validate Capabilities
Description: This use case describes the verification that agents perform when assigned a role.
Pre-Conditions: The agent has already been assigned a role. The agent’s assigned role can achieve the given goal.
Details: OMACS should only assign roles that this agent is able to play. However, we should perform validation of an agent’s capabilities at assignment time to make the system more robust in case the organizational does not have accurate knowledge of the agent’s current capabilities, or if a capability has failed.
Post Conditions: The agent is ready to execute the RLGM.
Specific Requirements:

3.1.4.1 SR6 [Critical Requirement]
The system shall read the set of required capabilities out of the assigned role. These capabilities will be the union of all capabilities discovered by looking up all leaf-level goals in the role’s RLGM.

3.1.4.2 SR7 [Critical Requirement]
The system shall iterate over all required capabilities from the role and check to make sure that each capability is in the set of capabilities possessed by the agent.

3.1.4.3 SR8
If the system does not possess a required capability, it shall inform the organization so that it can take appropriate action, such as assigning a new role to this agent.

3.1.5 Use Case 5: Execute Role Level Goal Model
Description: This use case describes how the role interpreter executes the RLGM to achieve the assigned goal.
Includes: Perform Actions
Pre-Conditions: The agent is playing a role in pursuit of a goal. The agent’s role has been defined using both an RLGM and a GCM.
Details: RLGM execution takes place according to GMoDS semantics: The interpreter will non-deterministically select a goal from the role’s set of active goals. It will then directly call that goal’s associated capability, with the specified parameters, by looking it up in the GCM associated with the role. The GCM provides a one-to-one mapping between RLGM leaf goals and a role’s abstract capabilities. The role’s abstract capabilities are mapped to the agent’s concrete capabilities
Post Conditions: Either the assigned goal is achieved, obviated, or the agent failed to achieve it.
Specific Requirements:

3.1.5.1 SR9 [Critical Requirement]
The system shall execute the RLGM according to GMoDS semantics. It will first select a goal from the RLGM’s set of active goals and pursue it.

3.1.5.2 SR10 [Critical Requirement]
If the current goal is a leaf goal, it shall be looked up in the GCM, and the action is performed.

3.1.5.3 SR11 [Critical Requirement]
If the current goal is not a leaf goal, then one of its children shall become the active goal.

3.1.6 Use Case 6: Perform Actions
Description: This use case describes the actions performed by the agent running the role interpreter while playing its role in pursuit of its goal.
Pre-Conditions: The agent is playing a role in pursuit of a goal. The agent’s role has been defined using both an RLGM and a GCM.
Details: As an agent operates, it performs actions to help it achieve the goal associated with the role it is playing. During execution, the role interpreter will call role level goals that are mapped to agent capabilities through the GCM. These capabilities represent either hardware or software functions that this agent possesses. These capabilities act as effectors on the agent’s environment, modifying it in some way that helps the agent achieve its goal.
Post Conditions: Either the assigned goal is achieved, obviated, or the agent failed to achieve it.
Specific Requirements:

3.1.6.1 SR12 [Critical Requirement]
The agent shall perform a requested action by making use of its capabilities. These effectors shall modify the environment to help the agent achieve its goal.

3.1.6.2 SR13
If an agent’s capability fails while performing an action, it shall report the failure to the organization so that appropriate actions can be taken.

3.1.7 Use Case 7: Report Status
Description: This use case describes the reporting the agent performs after attempting to achieve its goal.
Pre-Conditions: The agent attempted to achieve its goal by playing its assigned role.
Details: The agent can achieve its goal, the goal can become obsolete, or the agent can fail to achieve its goal due to capability failure, or some other external factor. This status needs to be reported back to OMACS so that new assignments can be given.
Post Conditions: The status of the agent’s goal is reported back to OMACS.

Specific Requirements:

3.1.7.1 SR14 [Critical Requirement]
The agent shall report the status of its goal back to the organization after attempting to complete its task

3.1.7.2 SR15
The agent’s status can be one of the following: Goal Achieved, New Goal Triggered, Goal Failure, or Goal Obviated.

3.2 Assumptions
• The user has JRE 1.6 available at runtime.
• The user has the Wumpi World simulator software installed. The version used for this project will be CIS-844-Fall-2010 from the projects.cis.ksu.edu CVS repository.

3.3 Constraints
• Agents in this system will not attempt to play Roles that are not defined by a Role Level Goal Model.
• All Agents will be provided a Goal Capability Map as part of Role assignment.
• Only a simple demonstration of this architecture will be provided using Wumpi World. A more complex simulation using the MACR Cooperative Robotic Organization Simulator (CROS) will not be provided.
• No role blending or decomposition will be attempted in this implementation.

3.4 Environment
• The agent role interpreter will be written in the Java language and it will be compiled using JDK 1.6.
• The agent role interpreter will be developed using the Eclipse IDE.
• The agent role interpreter will be tested using the Wumpi World simulator on the Windows platform using Windows 7.
• The version used for this project will be CIS-844-Fall-2010 from the projects.cis.ksu.edu CVS repository.