Test Plan

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 Test Plan Identifier 4
2 Introduction 4
3 Test Items 4
4 Features to Be Tested 4
4.1 SR1.1 4
4.2 SR1.2 5
4.3 SR1.3 5
4.4 SR1.4 5
4.5 SR2.1 5
4.6 SR3.1 5
4.7 SR4.1 5
4.8 SR4.2 5
4.9 SR5.1 5
4.10 SR5.2 5
4.11 SR6.1 5
4.12 SR7.1 5
5 Features Not to Be Tested 6
5.1 SR4.3 6
5.2 SR5.3 6
5.3 SR6.2 6
5.4 SR7.2 6
6 Approach 6
7 Item Pass/Fail Criteria 7
8 Suspension Criteria and Resumption Requirements 7
8.1 Suspension Criteria 7
8.2 Resumption Criteria 7
9 Test Deliverables 7
9.1 Test Log 7
10 Testing Tasks 7
10.1 Test Case 1 7
10.2 Test Case 2 7

10.3 Test Case 3 7
10.4 Test Case 4 8
10.5 Test Case 5 8
11 Environment Needs 8

1 Test Plan Identifier

GAA-TestPlan-1.0

2 Introduction

This document describes the testing that will be performed on the GMoDS-based Runtime Agent Role Interpreter. This project provides a generic Role Interpreter that allows GMoDS models to define role plans. These models are read in at runtime and then executed by mapping leaf-level goals to concrete capability methods. This system consists of three basic parts: The Role Interpreter, the Demonstration Agent Architecture, and the interface required to fit into the existing OMACS and GMoDS frameworks. This testing will be performed in accordance with the guidelines listed in the Project Plan 2.0 and the Software Quality Assurance Plan 1.0.

3 Test Items

The following items will be tested. The requirements specification for these items is provided in Vision Document 2.0, and the architectural design for each item is provided in System Architecture Design 1.0:

- Role Interpreter Framework
  - RoleInterpreter
  - GoalCapabilityMap
  - RoleLevelGoalModel
  - GaaRole
  - GaaCapability
  - GaaAgent
  - OmacsInterface

- Demonstration Agent Architecture
  - GoldDigger
  - HunterKiller
  - AdvancedBazooka
  - AdvancedGoldGrabber
  - AdvancedMovement
  - AdvancedSensors
  - AreaSearcherRole
  - GoldFetcherRole
  - GoldReturnerRole
  - HunterKillerRole

4 Features to Be Tested

The following items, outlined in Vision Document 2.0, will be tested.

4.1 SR1.1

The system shall accept a GMoDS RLGM and GCM as input representing the role it will play.

4.2 SR1.2

The system shall accept a GMoDS goal specification as input representing the goal it will achieve. The goal specification shall include goal type information and values for each attribute.

4.3 SR1.3

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

4.4 SR1.4

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.

4.5 SR2.1

The system shall parse the input RLGM into a goal tree structure using the GMoDS parser.

4.6 SR3.1

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.

4.7 SR4.1

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.

4.8 SR4.2

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.

4.9 SR5.1

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.

4.10 SR5.2

If the current goal is a leaf goal, it shall be looked up in the GCM, and the action is performed.

4.11 SR6.1

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.

4.12 SR7.1

The agent shall report the status of its goal back to the organization after attempting to complete its task

5 Features Not to Be Tested

5.1 SR4.3

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.

The demonstration architecture has the set of capabilities hardcoded in agents at compile time. There is also no organization to report the problem to. This has been done purposefully to simplify the demonstration architecture, as the primary purpose of this project is to implement the Role Interpreter.

5.2 SR5.3

If the current goal is not a leaf goal, then one of its children shall become the active goal.

This is an inherent property of the GMoDS model. There is nothing here to test.

5.3 SR6.2

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.

In the demonstration architecture, capabilities cannot fail. There is also no organization to report the problem to. This has been done purposefully to simplify the demonstration architecture, as the primary purpose of this project is to implement the Role Interpreter.

5.4 SR7.2

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

This is an inherent property of the GMoDS model. There is nothing here to test.

6 Approach

Only functional black box testing will be performed on this project. A WumpiWorld scenario has been prepared and it will be executed by the demonstration agent architecture. This architecture internally makes use of the Role Interpreter on four different roles defined using GMoDS models in the agentTool3 XML format. In addition, a GCM exists for each of five different capability classes.

Since successful execution of any one of these GMoDS models requires the successful execution of all system requirements, we will run the system through a series of increasingly more complex WumpiWorld scenarios and provide scoring and completion requirements for each scenario.

The WumpiWorld scenario is scored to reflect how well the agents in the environment accomplish their assigned roles. The fewer capabilities used and messages sent will result in a higher score. Additionally, if agents are able to avoid traps, kill Wumpi adversaries, and recover gold, their score

will be higher. In general, the higher the score, the better the agents were able to perform their roles.

7 Item Pass/Fail Criteria

Tests will be considered passed if they meet the specification requirement outlined in Vision Document 2.0, otherwise, they are considered to have failed.

8 Suspension Criteria and Resumption Requirements

8.1 Suspension Criteria

If a test case fails, then testing will be suspended for all features that depend upon that test case. The failed test case will be logged, along with a description of the failure. Other test cases that do not depend upon the failed feature will continue.

8.2 Resumption Criteria

For failed test cases, testing will resume once the defect that caused the original failure has been resolved.

9 Test Deliverables

9.1 Test Log

A test log will be produced, documenting all test cases and if the test case passed or failed. If a test case fails, then a description of the failure will also be logged and a corrective action will be taken, if appropriate.

10 Testing Tasks

10.1 Test Case 1

Scenario	Features Tested	Passing Criteria
HunterKiller.xml	All	At least three Wumpi are killed. No agent falls into a pit or is killed by Wumpi. Minimum score: 19000.

10.2 Test Case 2

Scenario	Features Tested	Passing Criteria
HunterKillerTeam.xml	All	At least five Wumpi are killed. No agent falls into a pit or is killed by Wumpi. Minimum score: 23000.

10.3 Test Case 3

Scenario	Features Tested	Passing Criteria
GoldDigger.xml	All	At least two pieces of gold are successfully recovered. No agent falls into a pit or is

		killed by Wumpi. Minimum score: -2000.

10.4 Test Case 4

Scenario	Features Tested	Passing Criteria
GoldDiggerHunterKillerTeam.xml	All	At least three pieces of gold are successfully recovered and three Wumpi are killed. No agent falls into a pit or is killed by Wumpi. Minimum score: 22000.

10.5 Test Case 5

Scenario	Features Tested	Passing Criteria
WumpiWorld.xml	All	At least four pieces of gold are successfully recovered and five Wumpi are killed. No agent falls into a pit or is killed by Wumpi. Minimum score: 25000.

11 Environment Needs

The Role Interpreter and demonstration agent architecture will be tested on a Windows 7 computer running the Java JDK 1.6. The demonstration architecture will be executed within the GMoDS and OMACS frameworks defined by the CIS-844-Fall-2010 WumpiWorld archive from the projects.cis.ksu.edu CVS repository.