Testing Plan

Project Title: Rules Based Decision Aid Framework

Distribution:

ASRC Federal Mission Solutions, Kimberly Davis

ASRC Federal Mission Solutions, Kevin Wainwright

ASRC Federal Mission Solutions, Christopher Barone

Rowan University, Professor Jack Myers

Revision and Signoff Sheet

Document History

Version	Date	Author	Description of Changes
1	04 December 2016	Michael Crinite	Draft

Approvers List

Name	Role	Approver/Reviewer	Approval/Review Date

Reference

Version	Date	Name
0.1	04 December 2016	Rules Based Decision Aid Framework

Table of Contents

Introduction

Document Purpose

This document outlines the Test Plan for ASRC Federal Mission Solution's Rules Based Decision Aid Framework. The plan's goal is to supplement the attached Test Script Document with relevant background information about the software and testing environment.

The Test Plan states how the Wolves development team will verify that the product is sufficiently functional according to the standards previously agreed upon. This document is also used as an accompaniment to the Validation Plan which outlines which forms of testing are being used, what sections of the product were tested, the extent of which they were tested and the environment being tested in.

Document Organization

The remainder of this document is organized as follows:

- Test Strategy
    - Testing objectives
    - Assumptions
    - Scope
    - Levels of testing
- Test Environment
    - System requirements
    - Hardware configuration
    - Training requirements of the testers

Project Overview

This document contains the Test Protocol for the Wolf Team Rules Based Decision Aid Framework. The version of the framework being released is 0.1. The software gives users the means to create and execute properly-formatted Drools Rules Engine files using disparate data to drive resultant actions. For more information, see the documentation included in the software package.

Test Strategy

Objectives

This protocol is written to accomplish the following testing objectives to ensure

- The product works as intended, according to the Design and Requirements Documents;
- Included features are fully-functional;
- Intended functionality is simple and user-friendly;
- Considerations have been made for possible variations in the user's desktop environment.

Assumptions

The user:

- Is not an expert with the JBoss Drools software utilized by the RBDAF;
- May have access to a JBoss Drools software expert;
- Has a basic understanding of the intended functionality and purpose of the RBDAF;
- Is trained with the RBDAF and can follow the instructions in the test scripts.

Data Approach

Sample data is provided to the user.

Scope and Levels of Testing

Two levels of testing should be completed: Developer, Sample User

- Developer -
    - Unit testing is to be completed in order to check for errors in the source code.
    - Integration testing will test for interaction between components.
    - System testing will test the full system, from an Drools/RBDAF expert point of view.
- Sample User -
    - The test script should be completed by the consumer, who should fit the assumptions in the section above.

Test Case/Requirements Traceability

See traceability matrix.

Test Environment

System Requirements

See “Requirements” in the attached Test Script Document.

Hardware Configuration

The testing will occur using

- (At least) one Windows machine and
- (At least) one Mac OS X machine

meeting the requirements specified above.

Training Needs

Testers of the software should be trained in use of the RBDAF application. At least one tester should *not* be an expert with the Drools rules engine.

Roles and Responsibilities

Name	Role	Responsibilities
Klaydon Balicanta	Scrum Master	Enacting the scrum principles and rules; Handles any problems occurring within the group.
Michael Moscariello	Product Owner	The contact between the Scrum team and stakeholders.
Trae Lewis	Development Team	Delivering potentially releasable increments of “done” at the end of each sprint.
Ian Markind	Development Team	Delivering potentially releasable increments of “done” at the end of each sprint.
Shiv Patel	Development Team	Delivering potentially releasable increments of “done” at the end of each sprint.
Michael Crinite	Development Team	Delivering potentially releasable increments of “done” at the end of each sprint.
Professor Jack Myers	Project Manager	Works closely with the scrum master to offer the team guidance.

Test Script Document

Project Title: Rules Based Decision Aid Framework

Distribution:

ASRC Federal Mission Solutions, Kimberly Davis

ASRC Federal Mission Solutions, Kevin Wainwright

ASRC Federal Mission Solutions, Christopher Barone

Rowan University, Professor Jack Myers

Script Identifier	Rules Based Decision Engine Test Script: Rule Creation and Firing			Script Version	
Environment	☐ Non Production ☐Production	Script Purpose	☐ Install ☐ Verify	Run #	

APPROVERS are not required

SCRIPT DETAILS

Objective

This test script will test all of the functions of the Rules Based Decision Engine, Wolf Team's front-end for the Rules Based Decision Aid Framework. Each test is separated by a number in the left-most column of the table. The tests were created to test both valid and invalid use.

- Test 1: Import Data - Tests input of a valid file (invalid file types are not displayed by the file chooser).
- Test 2: Add New Action - Tests input of a valid Action object, and invalid field options (empty fields).
- Test 3: Add New Rule
  - 3A: Tests invalid creation of rule (the save button should not be allowed in this step).
  - 3B: Tests invalid creation of rule before Action is added. Tests valid creation of rule.
- Test 4: Add New Condition - Tests valid and invalid values for fields.
- Test 5: View Rules - Tests view rule functionality.
- Test 6: Fire Rules - Tests firing of valid rules (Rules without proper syntax cannot be fired by the Drools engine).
- Test 7: Exit the Application - Tests successful system exit function.
- Test 8: Test Rule Chaining - Ensures Rule Chaining functionality

Script Identifier	Rules Based Decision Engine Test Script: Rule Creation and Firing			Script Version	
Environment	☐ Non Production ☐Production	Script Purpose	☐ Install ☐ Verify	Run #	

Requirements

- The user *must* have a valid installation of the software. The software can be downloaded using Git from https://github.com/TestPlan/rules-based-decision-engine.git.
- The user *must* be using IntelliJ to run the software at this time. *Because the graphical user interface is built using IntelliJ's editor, the software is currently required in order to build the GUI. In the future, the application will be released as a .jar file in which case the IntelliJ software will no longer be a requirement.*
- The user *must* have the following libraries installed:
  - Java JDK Update 8
  - JBoss Drools 6.1
  - json simple 1.1
  - Apache Commons CSV
- The user *may* be an expert with Drools Rules Engine, but the tests should also be completed by a user who is decidedly not an expert with the Drools Rules engine.
- The user *may* make sure the beacon.json file is included in the resources folder. The user *must* have a .json file available in the proper format.

Prerequisites and Setup

Before the tests are run, the user must have a version of the software that compiles. The user can open the software by running the main method in `/Rules Based Decision Engine/src/views/MainView.java`. If the software does not run, please make sure the requirements above are satisfied before contacting a Wolves development team member.

Script Identifier	Rules Based Decision Engine Test Script: Rule Creation and Firing			Script Version	
Environment	☐ Non Production ☐Production	Script Purpose	☐ Install ☐ Verify	Run #	

Tests:

Test	Description	Procedure	Test Data (Parameters)	Expected Results	Actual Results/Comments	Pass/Fail
1	Import Data	Click the “Import Data” button on the application’s main window.	none	A small window should pop up allowing the user to choose a File to import		
		Click the “Cancel Button”	none	The window should exit		
		Return to the dialog by clicking “Import Data” button again. Select the “Choose File” button.	none	A Windows File Explorer window should pop up allowing the user to select a file		
		Navigate to a the file in Rules Based Decision Engine/res/data called beacon.json. Click “Open”	beacon.json	The file path should appear in the text box		
		Click the “Save” button.	none	The dialog pop up should disappear.		
2	Add New Action	Click the “+ New Action” button in the application’s main window.	none	A small window should appear allowing the user to create a new Action		
		Click the “OK” button	none	A dialog should appear warning that the user must enter text.		
		Click the “OK” button	none	The warning should disappear and the Action popup should remain visible		
		Type to fill in the Name and Action fields	user-defined text	The fields should be populated.		

		NOTE: "System.out.println("Success") is recommended for the "Action" field	The "Action" field must contain a valid Java command, such as a print statement.			
		Click the "OK" button	none	The Action pop up should disappear.		
3A	Add New Rule	Click the "+ New Rule" button in the application's main window	none	The Rule dialog should pop up. In the "Action" section, the Action(s) created earlier should all be present and listed by name in the drop down menu.		
		Enter a title for your rule	user-defined	"Title" field should be populated		
		Click the "Save" button	none	A warning should appear that some parts of the Rule were not yet created.		
		Click the "OK" button	none	The pop up should disappear and return the user to the Rule window		
		Select a priority from the drop down menu.	user-defined	Selected "priority" should appear in the box.		
4	Add New Condition	Click the "+ Add Condition" button.	none	A pop up should appear allowing the user to select an Entity, Field, and operator, and enter a Value.		
		Click the "OK" button	none	An error message should appear warning that the user did not select or import an Entity.		

		Select an Entity from the Entity drop down menu	user-defined For this section, we reccomend selecting BEACON1	The drop down menu should contain "BEACON1" and "BEACON2." Upon selecting an Entity, the Fields drop down menu should populate with various Fields and autoselect the first available Field.		
		Select a Field from the Field drop down menu	user-defined For this section, we reccomend selecting DISTANCE	The drop down menu should contain the name of the selected Field.		
		Select an operator NOTE: The operator drop down menu is not labeled. It is the third drop down, and by default contains "="	user-defined For this section, we recommend selecting "<"	The drop down menu should contain the selected operator		
		Click the "OK" button	none	An error message should appear warning the user that the user did not input a value to compare to.		
		Enter a value in the Value field	user-defined For this section, we recommend entering 1000	The field should contain the entered value.		
		Click the "OK" button	none	The Condition dialog pop up should disappear. The screen should return to the Rule screen.		
3B	(Add New Rule)	Click the "Save" button	none	A error should appear warning that the user did not select and add an Action		

Script Identifier	Rules Based Decision Engine Test Script: Rule Creation and Firing			Script Version	
Environment	☐ Non Production ☐Production	Script Purpose	☐ Install ☐ Verify	Run #	

		Click the “OK” button	none	The pop up should disappear and return the user to the Rule window		
		Select an Action from the drop down menu and click the “+” button.	user-defined	The Actions text should include the Action the user selected.		
		Click the “Save” button	none	The Rule the user created should now be listed in the table on the application’s main screen. Its name should be the Title entered before.		
5	View Rule	Click the Rule’s name in the Rule table, on the application’s main window	none	The Rule’s row in the table should become highlighted.		
		Double-click the Rule’s name in the Rule table.	user-defined	A dialog window should pop up displaying the Rule text in its entirety. If the recommended values were input for the fields in steps 1-4, this should be a valid Drools rule.		
		Click the “OK” button	none	The pop up should disappear, returning the user to the main application screen.		
6	Fire Rules	Click the “Fire Rules” button	none	A Windows File Explorer window should pop up, allowing the user to navigate to a rule file.		
		Navigate to Rules Based Decision Engine/src/rules	none	All user-created rules and a few samples should appear in the window.		

		Select the Rule created in steps 1-4 and press "Open"	none	NOTE: The results of this step depend entirely on whether the user entered valid fields in their Data, Action, Rule, and Condition windows. If the user followed the recommended instructions, a valid Drools rule will have been created. If the user did not, a valid Drools rule *may not* have been created. If the user created a Rule following the recommendations, the console should output "Success." If the user created a valid Drools rule not following the recommendations, the output depends on the created Action. If the user created an invalid Drools rule, the console should output an Exception.		
7	Exit the Application	Press the "X" in the menu bar.	none	The application should successfully exit.		
8.	Test Rule Chaining	Reopen the program so that we can be sure that the Drools memory is empty.	none	The application window should pop up with no Rules in the table		
		Import Search_And_Rescue.json	Search_And_Rescue.json	The Entities from the file should be visible in the New Rule>New Condition>Select Entity Dropdown		
		Create an Action named "Rescue" with description `System.out.println("Dispatch to target")`	none	The application should return to the main menu		

Script Identifier	Rules Based Decision Engine Test Script: Rule Creation and Firing			Script Version	
Environment	☐ Non Production ☐Production	Script Purpose	☐ Install ☐ Verify	Run #	

		Create an Action titled "Abort" with the description `System.out.println("Abort Rescue")`	none	The application should return you to the main menu		
		Create an Action titled "Retrieve weather." Instead of entering a description, check the "Create Entity for Chaining" box	none	The application should open the Entity creation form		
		Name the Entity "Weather", click the "Add Field" button to add a field	none	A column with two rows should appear in the table (on Mac it may not be visible until clicked)		
		Double click the first column of the row to edit. Type "Storm" in the box. Click the row.	none	The first column of the row should contain "Storm"		
		Double click the second column of the row to edit. Type "true" in the box (case sensitive). Click the row	none	The second column of the row should contain "true"		
		Click the "Save" button	none	The Action's "Description" field should autopopulate and should now become un-editable.		
		Click the "Save" button	none	The application should return you to the main menu.		

Script Identifier	Rules Based Decision Engine Test Script: Rule Creation and Firing			Script Version	
Environment	☐ Non Production ☐Production	Script Purpose	☐ Install ☐ Verify	Run #	

		Click the New Rule button.	none	The New Rule Dialog should pop up		
		Create 3 rules: The first Rule should be titled “Rescue.” Its salience should be 2. Its condition uses these values: HELICOPTER, FUEL, >, 50 Add the Rescue Action.	as specified	A Rule should be created in the table with the fields specified		
		The second Rule should be titled “Abort.” Its salience should be -1. Its condition uses these values: WEATHER, STORM, =, true Add the Abort Action	as specified	A Rule should be created in the table with the fields specified		
		The third Rule should be titled “Get Weather.” Its salience should be 0. Its condition uses these values: WEATHERSTATION, TRANSMITTING, =, true Add Retrieve Weather Action	as specified	A Rule should be created in the table with the fields specified		

Script Identifier	Rules Based Decision Engine Test Script: Rule Creation and Firing			Script Version	
Environment	☐ Non Production ☐Production	Script Purpose	☐ Install ☐ Verify	Run #	

		Click the “Fire Rules” button	none	A Windows file explorer window should pop up		
		Navigate to your rules folder and select your newly created “Rescue.drl”	Rescue.drl	The new .drl should exist in your rules directory		

		Click the “OK” button	none	The console should print the results of the rule firing. Your screen should print:
				`Dispatch To Target`
				`Abort Rescue`		

RUN COMMENTS: (Enter tester notes and brief summaries of any Deviations and Exceptions documented)

Script Identifier	**Rules Based Decision Engine Test Script: Rule Creation and Firing**			**Script Version**	
Environment	☐ **Non Production** ☐**Production**	**Script Purpose**	☐ **Install** ☐ **Verify**	**Run #**	

ATTESTATIONS

Tester: By Signing below, the Tester(s) attest to completing the steps identified below

Printed Name and Signature		**Steps Completed:**	**Date:**

Witness: By Signing below, the Witness(es) attest to witnessing performance of the steps identified below**:**

Printed Name and Signature		**Steps Completed:**	**Date:**

Revision History

Version	Date	Description of Changes
1.0	12/07/2016	Initial Release
1.0	12/19/16	Added Tests for Chaining