Test Plan

Mastergoal Machine Learning Environment

Version 0.1

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

Alejandro Alliana

CIS895 – MSE Project

Change Log:

Date	Author	Revision	Comments
11/05/2007	Alejandro Alliana	0.1	Initial version

Index

1. Test plan identifier

CIS 748 Test plan Electrical distribution design system V1.0

2. Introduction

This document provides a test plan for the validation of the Mastergoal Machine Learning Environment. This document presents a set of test suites that the software must pass to be considered complete. Test stopping and resumption criteria as well as testing tasks are also specified.

3. References

[1] Alliana, Alejandro Vision Document 1.4 available at:
http://people.cis.ksu.edu/~aalliana/895/
[2] Alliana, Alejandro Project Plan 1.4 available at:
http://www.cis.ksu.edu/people/~aalliana/895

4. Test item

The project follows a modern software process. It is expected that testing will be carried out on every iteration. On the construction phase, the source code is expected to be tested at the unit level using automatic testing environments.

5. Features to be tested

This section specifies the features to be tested.

5.1. Use Case "Create New Experiment"

5.1.1. SR 1.1: Create Experiment (Critical)

The user must be able to create new experiments.

5.1.2. SR 1.1 Select Learning Technique (Critical)

The user must be able to select from different learning techniques for the experiment.

5.1.3. SR 1.2 Select Terms for experiment (Critical)

The user must be able to select which terms must be used for the current experiment

5.1.4. SR 1.3 Setup Search Algorithm (Critical)

The user must be able to setup a search algorithm, maximum depth for the search algorithm and maximum time to be spent in searching.

5.2. Use Case "Load experiment"

5.2.1. SR 2.1: Load experiment (critical)

The system shall allow the user to load previously saved experiment. The properties of the loaded experiment should match those specified by the data in the saved experiment.

5.2.2. SR 2.2 Save Experiment (Critical)

The system shall allow the user to save an experiment. The data saved must match the properties of the saved experiment.

5.2.3. SR 2.3 Edit Experiment (Critical)

The system shall allow the user to modify the information of an experiment.

5.2.4. SR 2.4 Experiment Format (Critical)

The system shall define a format to be used when saving and restoring the experiments. This format should allow extensibility to allow saving experiments with different types of learning techniques.

5.3. Use Case "Train Strategy"

5.3.1. SR 3.1: Train Strategy (Critical)

The program shall provide a command to train a strategy from an existing experiment. Different learning components will train their strategy differently.

5.3.2. SR 3.5: Rules of play (Critical)

The system shall enforce the rules at every game played during training.

5.3.3. SR 3.6: Correct search (Critical)

The systems search algorithms should return the best move according to the evaluation criteria and the search algorithm behavior.

5.4. Use Case "Export strategy "

5.4.1. SR 4.1: Export strategy (non critical)

The system shall provide a mechanism to export a learned strategy to a file in external format.

5.5. Use Case "Play Game"

5.5.1. SR 5.1: Play a game (Critical)

The system shall provide a GUI to play games. This user interface can be used to try the learned strategies. The games can be played between Human Players and or Computer program players.

5.5.2. SR 5.2: Rules (Critical)

During a game, the system shall enforce the application of the rules at all times.

5.6. Use Case "Explore Game"

5.6.1. SR 6.1: Explore a game (Critical)

The system shall provide a GUI to explore the games played during training.

6. Features not to be tested

The performance of the system is an item that is not going to be tested for the first version.

7. Approach

The system is going to be tested following the traditional approach of unit, component and integration testing. Automated testing suites are going to be used where possible.

7.1. Unit testing

The system is going to be tested at the unit level using the *CppTest* tool which allows automatic execution of test cases to prove for regression bugs.

7.2. Component testing and system testing.

After the software has been tested for bugs at the unit level, integration testing is going to be performed at the component and system levels

7.3. Documentation Testing

At the completion of the project, the user documentation will be assessed against the program and the requirements. These tests will ensure that no features are missing, all the documents are synchronized and the contents can be understood easily and unambiguously.

8. Item pass / fail criteria

Each test suite will be considered successful when the results are as specified in the vision document and failure otherwise.

9. Suspension criteria and resumption requirements

In case of failure of a test suite, the failed test cases should be recorded in the test log along with the description for failure. Testing should be suspended and errors fixed unless it requires a design change, in which case the fix can be delayed until the next iteration.

When a new version of the system is available after a bug fix, all previous tests will be rerun to ensure program changes have not inadvertently created regression bugs.

10. Test deliverables

The project will maintain a test log including the pass/fail result of each test, as well as reasons for failure, date, time and a recommended solution.

The project must also maintain a list of the open bugs with an expected date of fix.

11. Test tasks

11.1. Test suite 1: Use Case "Create New Experiment"

11.1.1. Test case 1.1:

Test Case	TC 1.1 SR 1.1 Create Experiment
Actors	Experimenter
Goal	Check create a new experiment
Preconditions	The system must be started and loaded.
Actions	• The experimenter selects the Train Strategy option. • The experimenter inputs the experiment data, o Experiment name (string), date (valid date), and obs. Data (string) • Searching data o Search algorithm (select from combo box), search depth [1..15], max time [0..60]sec. • Learning parameters:

	○ Terms to be used for the experiment. ○ Learning technique to be used: GA, RL, etc. ○ Learning technique specific parameters: i.e. for GAs: Population, crossover rate, mutation rate, fitness function.
Post conditions	• The system creates a new experiment.
Exceptions	• If the application had an active experiment opened with unsaved changes, the user must receive information about this.
Open issues	

11.1.2. Test case 1.2:

Test Case	**TC 1.2 SR 1.2 Select learning technique.**
Actors	Experimenter
Goal	Check the use different learning techniques with the environment.
Preconditions	The system must be started and loaded. The experimenter must create an experiment.
Actions	• The experimenter might select from a list of learning techniques. For the initial version only GAs will be provided • The user selects the data button to add the specific data to a learning technique.
Post conditions	• The system creates a new experiment specific to the learning technique.
Exceptions	• If the application had an active experiment opened with unsaved changes, the user must receive information about this.
Open issues	

11.1.3. Test case 1.3:

Test Case	**TC 1.3 SR 1.3 Select terms for experiment**
Actors	Experimenter
Goal	Check the use a subset of the terms available for an experiment.
Preconditions	The system must be started and loaded. The experimenter must create an experiment.
Actions	• The experimenter must select a set of terms to be used for the experiment from the checkboxes of terms in the experiment UI.
Post conditions	• The system creates a new experiment that uses that set of terms as the terms.
Exceptions	• If the application had an active experiment opened with unsaved changes, the user must receive information about this.
Open issues	

11.1.4. Test case 1.4:

Test Case	TC 1.4 SR 1.4 Setup search algorithm
Actors	Experimenter
Goal	Check the use a specific setting for the search algorithms
Preconditions	The system must be started and loaded. The experimenter must create an experiment.
Actions	• The experimenter must select a set of values to parameterize the search to be used in the experiment o Search algorithm (select from combo box) o Search depth [1..15] o Max time [0..60]sec.
Post conditions	• The system creates a new experiment that uses the search data just selected.
Exceptions	• If the application had an active experiment opened with unsaved changes, the user must receive information about this.
Open issues	

11.2. Test Suite 2: Use Case Load experiment.

11.2.1. Test case 2.1:

Test Case	TC 2.1 SR 2.1 Load Experiment SR 2.4 Experiment Format
Actors	Experimenter
Goal	Check loading an experiment from a file.
Preconditions	The system must be started and loaded. The experiment must exist in a file.
Actions	• The experimenter selects the train strategy option. • The experimenter selects the *open experiment* option. • The experimenter selects the experiment to load.
Post conditions	• The experiment is loaded into the GUI and the status and results are displayed if any.
Exceptions	• If the application had an active experiment opened with unsaved changes, the user must receive information about this.

	• There might be a file system error. • The file conforms to the experiment file specification
Open issues	

11.2.2. Test case 2.2:

Test Case	**TC 2.2 SR 2.2 Save Experiment SR 2.4 Experiment Format**
Actors	Experimenter
Goal	Check saving an experiment to a file.
Preconditions	The system must be started and loaded. The experiment must be created and loaded
Actions	• The experimenter selects the save option.
Post conditions	• The experiment is saved into a file.
Exceptions	• If a required field is not set, the system should display an error. • There might be a file system error. • The file conforms to the experiment file specification
Open issues	

11.2.3. Test case 2.3:

Test Case	**TC 2.3 SR 2.3 Edit Experiment SR 2.4 Experiment Format**
Actors	Experimenter
Goal	Check editing a previously saved experiment.
Preconditions	The system must be started and loaded. The experiment must be loaded
Actions	• The experimenter modifies some of the fields in the experiment. • The experimenter saves the experiment again.
Post conditions	• The experiment is saved into a file. • The attributes of the experiment in the file match the modified values. • The file conforms to the experiment file specification
Exceptions	• If a required field is not set, the system should display an error. • There might be a file system error.
Open issues	

11.3. Test Suite 3: Use Case "Train strategy"

11.3.1. Test case 3.1:

Test Case	TC 3.1 SR 3.1 Train strategy
Actors	Experimenter
Goal	Check obtaining a new strategy (set of new strategies) according to the training options and the learning procedure specified by the learning components.
Preconditions	• The new experiment must be created or loaded and the training options must be set.
Actions	• The experimenter selects the *start training option* • The program starts training with the specified arguments. • The program creates the directory structure to hold the strategies that are created.
Post conditions	• The learned strategy or set of strategies should be identified by the Experimenter using the GUI.
Exceptions	
Open issues	• The experimenter might want to stop training.

11.3.2. Test case 3.2:

Test Case	TC 3.2 SR 3.5 Rules of play.
Actors	Experimenter
Goal	Check that the games are played using correct rules.
Preconditions	• A game is created and the board set up.
Actions	• This test case should be tested creating different board situations and checking that the algorithm creates only valid moves and that it creates all the valid moves.
Post conditions	• Invalid moves should be identified. • All valid moves must be created. • Only valid moves should be created
Exceptions	
Open issues	

11.3.3. Test case 3.3:

Test Case	TC 3.3 SR 3.6 Correct Search.
Actors	Experimenter
Goal	Check that the search algorithm behaves correctly.
Preconditions	• A game is created and the board set up. • Search algorithm has all parameters set.
Actions	• This test case should be tested creating different starting board situations and analyzing subsets of the tree to check if the algorithm returns valid results.
Post	• The search algorithm should return a correct value according to

conditions	the search algorithm policy.
Exceptions	
Open issues	• The search is time and ply constrained, this should be accounted for when testing the search algorithm

11.4. Test Suite 4: Use Case "Export strategy"

11.4.1. Test case 4.1:

Test Case	**TC 4.1 SR 4.1 Export Strategy.**
Actors	Experimenter
Goal	Check the exportation of the strategy in a format that can be used by other programs.
Preconditions	• A trained strategy must exist saved in the file system.
Scenario	• The experimenter selects the *export strategy option*. • The experimenter selects the strategy to be exported. • The experimenter selects the location where to export the strategy. • The experimenter selects an exporter algorithm.
Post conditions	• The strategy is saved into a file using the exporter algorithm file format.
Exceptions	• Leaving some of the fields blank should be reported to the user as an error.
Open issues	

11.5. Test Suite 5: Use Case "Play Game"

11.5.1. Test case 5.1:

Test Case	**TC 5.1 Play Game**
Actors	Experimenter
Goal	Test the ability of a program to play games using a UI.
Preconditions	• At least one trained strategy exists in the framework
Scenario	• The experimenter selects the *Play/Explore game* option. • The experimenter sets up the game information. o Level [1,2, 3] o Team that starts [1,2] o Max time per play [1..60] sec o Team Name (String)

	○ Type of agent (Human, Computer) • The experimenter sets up the search tree information. ○ Search algorithm {loaded at runtime} ○ Max depth [1..15] • The experimenter selects the strategy. ○ Strategy {select from saved strategies} ○ Same data for other team. • The experimenter selects the start game option.
Post conditions	• Playing games reports no errors to the user.
Exceptions	
Open issues	

11.5.2. Test case 5.2:

See TC 3.2 Rules of play.

11.6. Test Suite 6: Use Case "Explore Game"

11.6.1. Test case 6.1:

Test Case	TC 6.1 SR 6.1 Explore Game
Actors	Experimenter
Goal	Allow the experimenter to scroll through moves of a game or to review games played during training.
Preconditions	• At least one trained strategy exists in the framework • The game to be explored must be opened in the program.
Scenario	• The experimenter scrolls through the moves of the game.
Post conditions	• The board gets updated when selecting the previous or next move.
Exceptions	
Open issues	

12. Schedule

Testing should be done throughout the development cycle. A schedule for the testing activities is given in the Project Plan [2]