Chapter 1

TEST PLAN (TP) TEMPLATE

Version 1.0, April 2014

FOREWORD

TEST PLAN (TP)

FOR

Swords and Sorcery

Version 0.5
May 8, 2014

Prepared for:
Dr. Jeffery

Prepared by:
Keith Drew
Ian Westrope
Jonathan Flake
Colin Clifford
David Klingenberg
Sean Shepherd
University of Idaho
Moscow, ID 83844-1010
CS383 TPD

uislogan.png

RECORD OF CHANGES (Change History)

Change Number	Date completed	Location of change (e.g., page or figure #)	A M D	Brief description of change	Approved by (initials)	Date approved

A - ADDED M - MODIFIED D – DELETED

TABLE OF CONTENTS

Section Page

1 TEST PLAN IDENTIFIER

TestPlan 0.1

2 REFERENCES

Use cases and UML diagrams are available on the class website, as well as game rules.
http://www2.cs.uidaho.edu/ jeffery/courses/383/

3 INTRODUCTION

Our plan is to integrate the working hud/game code and test that it works in tangent, as opposed to only working in discrete locations of the project. We will be using junit tests to evaluate discrete methods and manual tests to test the GUI and game logic.

4 TEST ITEMS

- Code Coverage (esp. Model)
- GUI Tests
  - Hexes/Units tile ok, look ok
  - Mouse clicks work
  - Menu navigation
  - Game starting (networked, scenario)
  - General gameplay (to the extent it's implemented)
  - Unit movement
  - Other
- Other manual tests (game compiles and runs on different platforms)
- Auto tests - Junit
- Ensure all files under resources can be loaded (if feasible)

5 SOFTWARE RISK ISSUES

Software to be tested includes the following:

1. GUI - Doesn't load working map, can't support unit movement
2. JSON Library - Loads scenario incorrectly, if not at all
3. Incorporating Networking with GUI and game logic may be difficult
4. Undetected Logic Errors (ULEs)
5. Misinterpretation of Rules

6 FEATURES TO BE TESTED

- Movement
- Teleporting
- Chat
- Loading
- Solar Display
- Diplomacy Display

7 FEATURES NOT TO BE TESTED

- Game networking
- Full game
- Spell casting
- Everything in backlog

8 APPROACH

The plan is to use Junit tests and manual tests to ensure that our code follows the rules of the game and works itself. Junit tests are done according to the individuals who have developed the methods being tested, and those are not listed here. However, they should be able to be run together and work. The manual tests will work as though a mock player (tester) is running the game. They should be able to select a unit, move a unit, teleport a unit, advance and view the solar display, send chat messages, and view the diplomacy display. Functionalities of each display should also be tested. For example, a unit with a move allowance should only be viewed moving at or under their limit and a chat message should be sent from one user and be seen by all users.

9 ITEM PASS/FAIL CRITERIA

All tests should meet the specifications of the Swords and Sorcery board game rules manual, as well as follow logical design patterns and language rules/conventions. The manual tests of movement should fall within 20% of total movement possibilities the movement rules indicate.

10 SUSPENSION CRITERIA AND RESUMPTION REQUIREMENTS

- If the github project is broken or otherwise compromised, productive development and testing would be suspended for a time. To resume, fix whatever has been broken on github.
- If any component test fails, the larger dependent pieces are unable to be tested until the component is fixed. In such a case, fix the component and continue testing.

11 TEST DELIVERABLES

- Test plan document
- Test cases
- Relevant error logs or problem reports
- Possible solutions

12 REMAINING TEST TASKS

There are many remaining test tasks, as we have not achieved a working game yet. Remaining tests include full combat, spell casting, scenario loading, full gameplay, etc.

13 ENVIRONMENTAL NEEDS

We need two computers connected over the internet, using Gabe's RaspberryPi. Otherwise, we cannot test full networking capabilities and full gameplay as intended.

14 RESPONSIBILITIES

The hierarchy of "inchargeness" is as follows:

1. Dr. J
2. John Goettsche
3. Everyone else

15 SCHEDULE

Junit tests should be implemented and run as functionality is added to classes and packages. Manual testing will be done as the GUI is developed and functionality added, as well as when more aspects of other group's code are added.

16 PLANNING RISKS AND CONTINGENCIES

As the end of the semester is a fixed date, there are no contingencies for failure. Public beatings will be carried out as needed.

17 APPROVALS

The only person capable of fully approving any fixes/modifications is Dr. J.

18 GLOSSARY

Junit test is a discrete code test designed to be ran as part of an automated test sequence that tests all Junit tests.