MCMASTER UNIVERSITY

CAS 4ZP6

TEAM 9

CAPSTONE PROJECT 2013/2014

PORTER SIMULATION

Test Plan Revision 0

Authors:
Vitaliy Kondratiev
Nathan Johrendt
Tyler Lyn
Mark Gammie

Supervisor:
Dr. Douglas Down

October 30, 2013

CONTENTS

1 Revision History

Revision #	Author	Date	Comment
1	Vitaliy Kondratiev, Nathan Johrendt, Tyler Lyn, Mark Gammie	October 28	Adding Test Plan Revision 0
2	Vitaliy Kondratiev, Nathan Johrendt, Tyler Lyn, Mark Gammie	October 29	Test Plan Updates
3	Vitaliy Kondratiev, Nathan Johrendt, Tyler Lyn, Mark Gammie	October 29	Test Plan Update
4	Vitaliy Kondratiev, Nathan Johrendt, Tyler Lyn, Mark Gammie	October 30	Final Update for Revision 0 Test Plan

2 Executive Summary

2.1 Introduction

This document is designed to outline testing methods and techniques that are to be used during and after development of the Porter Simulation. Below listed in detail are the main test factors and the rationale for choosing them. Following the main test factors is a comprehensive list of specific system tests including information about the Test's factor, life cycle phase, type, whether it is static or dynamic, manual or automated, and the specific techniques used to conduct the test.

2.2 Automation Tools

1. **Mock:** Mock is a Python library used for easy isolation of Python functions for testing and assertions
2. **Unittest:** Unittest is a Python framework for constructing automated tests
3. **Excel:** Excel will be used for analysing statistical information and verifying its correctness
4. **Virtual Box:** Virtual Box will be used when testing the portability of our software between different operating systems

3 Test Factors and Rationales

3.1 Reliability

Rationale: Since the simulation software is to be used by a non-technical staff, consistent execution and termination of the program is required. Individual simulations could run for long periods of time without requiring user interaction, and are expected to terminate and store results without user supervision.

3.2 Ease of Use

Rationale: End users are non-technical so any interaction with the program, whether input or output, should contain a minimal amount of technical information.

3.3 Portability

Rationale: Control of the systems that the Simulation will run on is left to the end users, so the implementation will be designed to function on a wide variety of industry-popular operating systems.

3.4 Correctness

Rationale: Since the Simulation will be reporting statistical data, the Simulation must be correct in managing and manipulating that data.

4 Specific System Tests

4.1 Event List Correctness 1

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Simulation event list is generated by initializing the simulation

(ii) **Input:** Event List of 50 Events

(iii) **Description:** As the simulation is put through one step iteration on event list pops off the event list

(iv) **Expected Output:** Observe that the system Event List contains 49 Events

4.2 Event List Correctness 2

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Static

Manual/Automated: Manual

Technique:

(i) **Initial State:** Assume Event List is generated

(ii) **Input:** Assume event List contains 50 Events

(iii) **Description:** Examine the code by working through the process of popping off one event

(iv) **Expected Output:** Conclude the event list contains 49 Events

4.3 Event List Correctness 3

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Automated

Technique:

(i) **Initial State:** Simulation event list of length 100 is generated by initializing the simulation

(ii) **Input:** Event list

(iii) **Description:** As the system is tested during the course of development a report is always generated after execution

(iv) **Expected Output:** System returns an output for every event that has come off the list

4.4 State Change Correctness 1

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Simulation event list is generated by initializing the simulation, Porters states generated by initializing simulation and set to "Available"

(ii) **Input:** Single Event to be assigned to a Porter, Porter with state "Available"

(iii) **Description:** After single iteration of the simulation, execution of Event causes a single Porter to change state

(iv) **Expected Output:** Porter with state "Dispatched"

4.5 State Change Correctness 2

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Static

Manual/Automated: Manual

Technique:

(i) **Initial State:** Assume Event list is initialized, Assume all Porters are initialized to "Available"

(ii) **Input:** Assume a set of "Pending" events and a set of "Available"

(iii) **Description:** Examine the code by working through the process of single event being assigned to an "Available" Porter

(iv) **Expected Output:** Conclude the Porter's state is changed to "Dispatched"

4.6 State Change Correctness 3

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Automatic

Technique:

(i) **Initial State:** Simulation event list of length 100 is generated by initializing the simulation, 100 Porter states generated by initializing simulation and set to "Available"

(ii) **Input:** List Event to be assigned to a Porter, List of Porters with state "Available"

(iii) **Description:** As the system is tested during the course of development a report is always generated after execution detailing the state of Porters

(iv) **Expected Output:** System returns an output for every Porter state change

4.7 Porter/Event Linkage Correctness 1

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Simulation event list is generated by initializing the simulation, Porters states generated by initializing simulation

(ii) **Input:** A single "Pending" event, Single "Available" Porter

(iii) **Description:** After a single iteration the event is linked to the Porter

(iv) **Expected Output:** Porter and Event a linked together uniquely by comparing to all other Porter/Event pairs

4.8 PORTER/EVENT LINKAGE CORRECTNESS 2

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Static

Manual/Automated: Manual

Technique:

(i) **Initial State:** Assume simulation event list is generated by initializing the simulation, assume porters states generated by initializing simulation

(ii) **Input:** A single "Pending" event, Single "Available" Porter

(iii) **Description:** Examine the code and work through the process of linking an "Pending" event to a "Available" Porter

(iv) **Expected Output:** Conclude that Porter and Event a linked together uniquely by comparing to all other Porter/Event pairs

4.9 PORTER/EVENT LINKAGE CORRECTNESS 3

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Automated

Technique:

(i) **Initial State:** Simulation event list of length 100 is generated by initializing the simulation, 100 Porter states generated by initializing simulation

(ii) **Input:** List of "Pending" Events, Set of "Available" Porters

(iii) **Description:** System records every linkage created during simulation and compares them to each other.

(iv) **Expected Output:** A report is generated detailing every link between Porter/Event. System compares all linkages and concludes they are unique.

4.10 TASK POOL CORRECTNESS 1

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Simulation event list is generated by initializing the simulation, porter states generated by initializing simulation all with states ≠ "Available"

(ii) **Input:** Single Event, Set of Porters with state ≠ "Available", Empty Task Pool

(iii) **Description:** Events that cannot link to a "Available" Porter go to the Task Pool

(iv) **Expected Output:** Task Pool contains one additional Event

4.11 TASK POOL CORRECTNESS 2

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Static

Manual/Automated: Manual

Technique:

(i) **Initial State:** Assume simulation event list is initialized, assume porter states are initialized with all states ≠ "Available"

(ii) **Input:** Single Event, Set of Porters with state ≠ "Available", Empty Task Pool

(iii) **Description:** Examine code and work through the process of events that cannot link to a "Available" Porter, and checking that those events go to the Task Pool

(iv) **Expected Output:** Conclude that the Task Pool contains one additional event

4.12 TASK POOL CORRECTNESS 3

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Automatic

Technique:

(i) **Initial State:** Simulation event list of length 100 is generated by initializing the simulation, 100 Porter states generated by initializing simulation

(ii) **Input:** List of events, set of porters

(iii) **Description:** Each time an event is added to the task pool the system generates a report

(iv) **Expected Output:** Report of all events that were added to the Task Pool

4.13 TERMINATION CORRECTNESS 1

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Task pool is empty of events and all porters are in state "Available"

(ii) **Input:** Empty Task Pool

(iii) **Description:** System recognizes initial state as a termination condition

(iv) **Expected Output:** Simulation termination

4.14 TERMINATION CORRECTNESS 2

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Static

Manual/Automated: Manual

Technique:

(i) **Initial State:** Assume Task pool is empty of events and assume all porters are in state "Available"

(ii) **Input:** Assume empty Task Pool

(iii) **Description:** Examine the code to conclude that a termination condition has been met

(iv) **Expected Output:** Conclude simulation termination

4.15 TERMINATION CORRECTNESS 3

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Automated

Technique:

(i) **Initial State:** 100 simulations are initialized

(ii) **Input:** 100 simulations with termination conditions "Empty Task Pool, All Porters in state 'Available'"

(iii) **Description:** Testing system starts up 100 simulations and records which have terminated once reaching the termination condition

(iv) **Expected Output:** A report is generated by a testing system detailing what systems have terminated

4.16 INPUT/INITIALIZATION CORRECTNESS 1

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Uninitialized Simulation

(ii) **Input:** Simulation Input File (.csv extension), containing information; 10 Porters, 10 Events

(iii) **Description:** Once input file is loaded by the simulation, it has become initialized with modified values

(iv) **Expected Output:** Simulation initializes 10 Porters and 10 Events

4.17 INPUT/INITIALIZATION CORRECTNESS 2

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Automatic

Technique:

(i) **Initial State:** 100 Uninitialized Simulations

(ii) **Input:** 100 Simulation Input Files (.csv extension) containing information; 1 to 10 Porters, 1 to 10 Events, 100 unique combinations

(iii) **Description:** Testing program takes in 100 Input files and Initializes 100 Simulation

(iv) **Expected Output:** Testing Program generates a report on all 100 simulation that were initialized

4.18 THE GOLDEN TEST

Test Factor: Reliability & Correctness

Life Cycle Phase: Throughout Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Uninitialized Simulation in a deterministic mode

(ii) **Input:** 'Gold Copy' set of events. 5 Events; Porter moving patient from Emergency to ICU using stretcher, Porter moving oxygen tank from storage to a room in Ward A, Porter moving specimen samples from laboratory A to laboratory B, Porter moving patient from X-Ray to Ultrasound using wheelchair, Porter takes a break

(iii) **Description:** Using a 'Gold copy' of a set of events, new program builds are retested using the 'gold copy' to ensure consistency of execution by examining inconsistencies in the output file

(iv) **Expected Output:** A difference file comparing 'gold copy' statistics against the newly tested code outlining the inconsistencies between them

4.19 USABILITY TEST 1

Test Factor: Ease of Use

Life Cycle Phase: Final Stages of Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Simulation prior to execution

(ii) **Input:** End user with an example simulation set; Create Input file, Load Input File into Simulation, Initialize Simulation

(iii) **Description:** End user is provided with a set of instructions to Initialize the Simulation

(iv) **Expected Output:** The end user successfully Initializes the Simulation

4.20 COMPATIBILITY TEST

Test Factor: Portability

Life Cycle Phase: Final Stages of Development

Type: Functional

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** Unknown Operating System with the simulation accessible on local storage

(ii) **Input:** 'Gold Copy' simulation parameters (Details outline previously under 'The Golden Test')

(iii) **Description:** Simulation is run and compared to results computed on other operating systems to ensure it is functioning normally in the new environment

(iv) **Expected Output:** Simulated results are consistent with previously generated values

4.21 DISTRIBUTION CORRECTNESS 1

Test Factor: Correctness

Life Cycle Phase: Throughout Development

Type: Structural

Dynamic/Static: Dynamic

Manual/Automated: Manual

Technique:

(i) **Initial State:** N/A

(ii) **Input:** Known distributions in our simulation: Poisson

(iii) **Description:** Run the distribution when creating the event times and examine the results through a proven program that compares the regression (Excel)

(iv) **Expected Output:** After examination the results follow the Poisson distribution

5 PROOF OF CONCEPT TEST

Proof of Concept Test will include these tests

(a) Event List Correctness 1

(b) State Change Correctness 1

(c) Task Pool Correctness 1

(d) Termination Correctness 1

(e) Input/Initialization Correctness 1

6 SCHEDULE

Phase #	Phase Name	Start Date	Description
1	Development Phase 1	November 1^{st}	N/A
2	1^{st} Proof of Concept	November 12^{th}	20 Minute demonstration delivered for the rest of the capstone class
3	Development Phase 2	November 13^{th}	N/A
4	2^{nd} Proof of Concept	December 2^{nd}	Demonstration delivered to the stakeholders
5	Development Phase 3	December 3^{rd}	N/A
6	Test Phase 1	March 1^{st}	N/A
7	Development Phase 4	March 15^{th}	N/A
8	Testing Phase 2	March 28^{th}	N/A
9	Final Demonstration	March 29^{th}	N/A