Software Test Plan (STP)

Software It Counts (SWIC)

CMSC 447

Updated on November 25, 2015

Table of Contents

1 Scope

1.1 Identification

For this project, we will be working with the web application, Parable of the Polygons. The software will simulate segregation levels between three different shapes, allowing them to move to different locations on the board in an attempt to be happy. The website will be programmed in HTML while the application will be programmed in JavaScript. The repository is located at https://github.com/Kirkas1/polygons, and branches off the source code at https://github.com/dncnmcdougall/polygons.

1.2 System overview

This project has tasked us with inserting a 3rd polygon, a red circle, into the game, allowing for relations between 3 different shapes to be shown. In addition, two new algorithms will be created. One such algorithm will be based around movement when based on the happiness levels of single polygons, henceforth known as the "happiness algorithm." The second algorithm will be based around the happiness of single polygons, as well as the happiness of those in the 8 squares adjacent to them, henceforth known as the "collective happiness algorithm."

The following document will address the test plan used to accomplish the requirements above, as given by the customer Russ Cain. The subsequent test plan for the project will thoroughly detail how the testing is to be carried out by the group (SWIC). This document will (like all documents) be uploaded to the project's repository at https://github.com/Kirkas1/polygons, which has the history of system development thoroughly summarized.

1.3 Document overview

This document will specifically layout a plan for testing the system and its requirements that the group needs to fulfil for the project (as previously discussed by Professor Cain). The test plan will address the testing environment in detail and the items necessary to properly carry out the testing. It will also discuss the general test conditions and the overall test progression. The test procedure and handling shall be carried out via the test schedule by only group members, and the results of such shall be published and assessed by our customer, Russ Cain.

2 Referenced documents

1. Parable of Polygons, Revised Apr 18, 2015, https://github.com/ncase/polygons, Vi Hart and Nicky Case
2. Parable of Polygons Source Code, Revised Oct 25, 2015, https://github.com/ncase/polygons, Vi Hart and Nicky Case
3. Polygons, Revised Dec 9, 2014, https://github.com/dncnmcdougall/polygons, Duncan McDougall

3 Software Test Environment

3.1 Name of test site(s)

The software testing site shall be UMBC's campus, its library and Fine Arts Hall Room 215 being the specific locations.

3.2 Software items

All of Team SWIC will bring his or her laptop to conduct testing on individual ends; the operating systems are either Windows 7 and onwards, or Linux. Each laptop which, regardless of operating system, has either the latest (as of November 2015) version of Fire Fox, Internet Explorer, Chrome, or any combination of the aforementioned browsers. All laptops will also have access to GitHub, where the code is stored.

3.3 Hardware and firmware items

No other outside devices, such as Ethernet cables, are necessary.

3.4 Other materials

The Parable of The Polygons web site comes with its own manual/instructions and does not require additional reading to aid in understanding. Finally, neither The Parable of The Polygons nor related materials raise a security/privacy concerns due to project nature.

3.5 Proprietary nature, acquirer's rights, and licensing

The code will not need any licensing to use—it is readily available to view and use on GitHub. The code will also not gain any monetary profit.

3.6 Installation, testing, control

In order to set up the software testing environment, all laptops must have one or more of the latest (as of November 2015) versions of Fire Fox, Internet Explorer, and Chrome. All laptops used to code the software must also have the desktop or shell version of GitHub, as well as be a contributor to The Parable of The Polygons repository. All of these programs can be found on their official web sites. In order to "test"/prepare the software components, all that will need to be done is make sure Internet access is possible, and that laptop's GitHub can access, push, and sync to the correct repository. For maintenance, each laptop must update its browsers and/or GitHub if updates to either software are released.

3.7 Participating organizations

No other facility/team outside of Team SWIC will partake in the testing or preparation thereof.

3.8 Personnel

At least one person should test code, but potentially as many people as available can help to test the code. (However, the sandbox does not allow for multiple computers to see what one computer has for input.) The person must have the capability to read at the level of at least a fourth grader, perform simple fractions (e.g. 1/8, 2/5), and understand concepts such as bias, discrimination, and segregation. No other special skills will be needed. Between Team SWIC, all team members must be present from September 14th, 2015 to December 7th, 2015 to test code. Otherwise personnel from the team or otherwise will not be necessary.

3.9 Orientation plan

The Parable of The Polygons already provides a brief overview/training of how to use it. No outside sources will be needed to aid further understanding in the code, nor will a specialized team be needed to assist in understanding the code.

3.10 Tests to be performed

The tests to be performed are as follows: a red circle behaving like the yellow triangle and blue square (shakes, moves based on neighbor preference); three radio buttons which allow the user to select a desired algorithm; a triangularly-shaped slider which permits the user to change the relationships between the red circle, yellow triangle, and blue square; a second algorithm calculating how happy a polygon feels around its neighbors and moving to a spot where it would be happier; a third algorithm which is similar to algorithm 2, but also considers how its neighbors feel about it; and that the code works on the latest versions of Fire Fox, Internet Explorer, and Chrome.

4 Test identification

[redacted]General information

The red circle tests shall occur after they are developed and integrated into the system. After the red circles pass the test requirements without breaking the program, then our novel algorithm(s) shall be tested against the existing segregation method. Once our algorithm surpasses the existing one in average time to segregate, then the system will be complete and deliverable. When the system is complete, then it will be integrated and tested on separate web browsers to minimize software restrictions.

4.1.1 Test levels

The red circle shall be unit tested for identical functionality as the existing shapes in the Parable of Polygons. The algorithms shall be tested on the system level, utilizing every element of the automatic object. The web browser testing shall be system wide as well.

4.1.2 Test classes

The folllowing tests shall be performed on the red circle: unit tests for the added methods, three step tests on the added slider, limit tests on the slider, and integration tests with the other requirements and added features after these tests.

The following tests shall be performed on the new and old algorithms: timing tests, three step quality testing on their radio buttons, and the same tests on any new algorithms or features that address these requirements.

4.1.3 General test conditions

The red circle shall be proven equally functional as the tested blue square and yellow triangle. This includes each of their functions: dragging, reaction to new neighbors, and automatic movement. The algorithm must be proven to be faster than the random segregation method. The new objects involved in the interface of the red circle and algorithm shall be tested as well: the triangular slider and the radio buttons. The slider shall be proven to stay within its boundaries, and to be exact in selecting shape bias. The radio buttons shall prove to select the proper algorithm. They shall also prove to change properly after at least three new selected algorithms. All new methods added to the system after these tests shall be documented and tested for separately in new documentation.

4.1.4 Test progression

The red circle shall be integrated into the system. Then the interface for the red circle shall be proven flawless. The algorithm shall then be tested on the complete three-shape grid. Once the system runs on one web browser, other browsers will be integrated and tested to minimize software constraints.

4.1.5 Data recording, reduction, and analysis

The times of the algorithms during the timing tests shall be recorded on a digital spreadsheet, and the average times of each trial will be calculated and analyzed. The timing tests shall be complete when the average time to segregate is greater in the original algorithm than in our novel algorithm.

4.2 Planned tests

The following tests shall be performed on the system and are described in the subsections 4.2.x:

The Red Circle,

The New Algorithms, to add judgment to shapes based on the available spaces.

Google Chrome and Internet Explorer web browser functionality

4.2.1 The Red Circle

The red circle shall be a new polygon with equal functionality as the original two.

4.2.1.1 Drag Testing

a) *Objective:* The new shape must be able to be dragged and placed without crashing the program, and interact with neighboring shapes accordingly.
b) *Level:* The test is on the system level.
c) *Class:* Integration test
d) *Qualification method(s):* A red circle must be draggable, placed in a position where it would be satisfied, and its neighbors shall react to its placement based on their bias.
e) *Requirement(s) addressed:* "The system shall include a new polygon." and "The new polygon shall be a red circle."
f) *Special Requirements:* None
g) *Data Recorded:* Pass / Fail
h) *Analysis:* Pass / Fail
I) *Assumptions/Constraints:* The customer will be interested in dragging the red circle as they would on the original website.

4.2.1.2 Automatic Segregation testing

a) *Objective:* When the "Go" button is pressed, the red circles shall be dragged automatically as they become unsatisfied with their neighbors based on their bias.
b) *Level:* System
c) *Class:* Integration
d) *Qualification method(s):* The "Go" button on the automatic object shall be pressed, and the red circles shall behave as the other shapes to, eventually settling once every shape on the grid is satisfied.
e) *Requirement(s) addressed:* "The system shall include a new polygon." and "The new polygon shall be a red circle."

f) *Special Requirements:* none
g) *Data Recorded:* Pass / Fail
h) *Analysis:* Pass / Fail
I) *Assumptions/Constraints:* The automatic interface addresses the methods used by the shapes which are not used when dragging them manually.

4.2.1.3 Surroundings Judgment

a) *Objective:* The red circle must react accordingly when surrounded with shapes inside and outside of its preferred bias percentages.
b) *Level:* System
c) *Class:* Integration
d) *Qualification method(s):* Red circles shall be surrounded by other polygons manually by the user, and then their satisfaction shall be modified according to their given bias.
e) *Requirement(s) addressed:* “The system shall include a new polygon.” and “The new polygon shall be a red circle.”
f) *Special Requirements:* none
g) *Data Recorded:* Pass / Fail
h) *Analysis:* Pass / Fail
I) *Assumptions/Constraints:* The shape's reactivity is fully tested by the method listed above.

4.2.1.4 Slider Interface Testing

a) *Objective:* The slider for selecting bias in the three shapes shall not go out of bounds or be able to select a bias below the minimum and maximum. The slider must change the shapes' bias accordingly.
b) *Level:* System
c) *Class:* Integration, Bounds
d) *Qualification method(s):* The slider shall be modified for each shape and prove to select the bias it claims to for each of the three shapes. The boundaries of the slider shall stop the selector from moving beyond the bounds of the object.
e) *Requirement(s) addressed:* “The system shall include a new polygon.” and “The new polygon shall be a red circle.”
f) *Special Requirements:* none
g) *Data Recorded:* Pass / Fail
h) *Analysis:* Pass / Fail
I) *Assumptions/Constraints:* none

4.2.2 The New Algorithm(s)

The happiness algorithm, and in scope the collective happiness algorithm, shall be tested through the following methods:

4.2.2.1 Timing Testing

a) *Objective:* Prove that the algorithm(s) are more efficient than the original.
b) *Level:* System
c) *Class:* Timing

d) *Qualification method(s):* The times until complete segregation shall be recorded for each of 20 sessions. The average time shall be calculated and compared between algorithms.
e) *Requirement(s) addressed:* “Two new algorithms shall be added to the system.” and “The new algorithm shall be more effective than the original.”
f) *Special Requirements:* none
g) *Data Recorded:* Times in seconds
h) *Analysis:* The averages of each trial for each individual algorithm.
I) *Assumptions/Constraints:* none

4.2.2.2 Radio Button Quality Assurance

a) *Objective:* The radio buttons must change the algorithm selection according to the way they are labeled and selected by the user.
b) *Level:* System
c) *Class:* Integration
d) *Qualification method(s):* Debug messages shall be added for this test to display when a new algorithm is selected. The radio buttons will be selected no less than three times, the debug messages shall display accordingly, and when “Go” is pressed, the automatic.js object shall run properly.
e) *Requirement(s) addressed:* Radio buttons shall be added for selecting the running algorithm.
f) *Special Requirements:* none
g) *Data Recorded:* Pass / Fail
h) *Analysis:* Pass / Fail
I) *Assumptions/Constraints:* The algorithms are assigned before the segregation starts.

4.2.3 Web Browser Functionality

Google chrome, and possibly internet explorer, should be adapted for user accessibility to match the original program.

4.2.3.1 Google Chrome

a) *Objective:* The system must be fully functional on the Google Chrome web browser.
b) *Level:* System
c) *Class:* Integration, Software outside the system
d) *Qualification method(s):* All of the above tests shall be executed and analyzed on the Google Chrome web browser.
e) *Requirement(s) addressed:* “The system shall run on Google Chrome”
f) *Special Requirements:* none
g) *Data Recorded:* Pass / Fail
h) *Analysis:* Pass / Fail
I) *Assumptions/Constraints:* The above tests completely address the use cases for this browser, most users shall use the newest version of the web browser. The original system runs on this web browser.

4.2.3.2 Internet Explorer

a) *Objective:*
b) *Level: a) Objective:* The system must be fully functional on the Internet Explorer web browser.

b) *Level:* System
c) *Class:* Integration, Software outside the system
d) *Qualification method(s):* All of the above tests shall be executed and analyzed on the Internet Explorer web browser.
e) *Requirement(s) addressed:* "The system shall run on Internet Explorer"
f) *Special Requirements:* none
g) *Data Recorded:* Pass / Fail
h) *Analysis:* Pass / Fail
I) Assumptions/Constraints: The above tests completely address the use cases for this browser, most users shall use the newest version of the web browser. The original system runs on this web browser.

5 Test schedules

Test	Date	Location	Description	Hardware/software configurations	Actors
Red circle test (Observat-ional)	November 20, 2015 8 PM-10 PM	UMBC Library, on each team member's laptop	Test the red circle to see if it actslike the other shapes (e.g. moved if a certain amount of shapes were around it, and shook like the other shapes).	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops with Windows 7 or Linux	Team SWIC
Selected algorithm radio buttons (Observat-ional)	November 21, 2015 By 11:59 PM	Ian's laptop, dorm	Test the three, added radio buttons for each algorithm (random, happiness, collective happiness).	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops with Windows 7 or Linux	Team SWIC (Ian)
Random Algorithm Test	November 21, 2015 By 11:59 PM	All of Team SWIC's laptops	Test the Random Algorithm via the according radio button and make sure it functions as it did previously.	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops with Windows 7 or Linux	Team SWIC
Happiness Test	November 22, 2015 By 11:59 PM	Chris's laptop, dorm	Test the Happiness Algorithm and see if the polygons move to spot where they're happy.	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops with Windows 7 or Linux	Team SWIC
Collective Happiness Test	November 22, 2015 By 11:59 PM	All of Team SWIC's laptops	Test the Collective Happiness Algorithm and assess whether it works as intended.	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops	Team SWIC

				with Windows 7 or Linux	
Internet Explorer, Fire Fox, and Chrome Test	November 23, 2015 7 PM	All of Team SWIC's laptops	Tested to see if the code ran on the browsers listed.	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops with Windows 7 or Linux	Team SWIC
Work on Software Test Results (STR)	November 25, 2015 5 PM - 7 PM	All of Team SWIC's laptops	Assign parts of STR and finish any necessary testing	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops with Windows 7 or Linux	Team SWIC
3-sided Slider Test	November 2015	??	Test the slider on the interface and determine if polygon biases are accurately portrayed by ratio.	Latest version (as of November 2015) of Fire Fox, Internet Explorer, and Chrome on laptops with Windows 7 or Linux	Team SWIC

6 Requirements traceability

Red Circle Software Test Description:

The test cases herein acknowledge the requirements which state, "the system shall include an additional shape, to a total of three shapes," "the third shape added to the program shall be a red circle," and, "the bias of the new shape shall be represented with a triangular ui component, as illustrated in the SDP." The resulting software will be tested by unit, and eventually integrated in the automatic segregator component of Parable of Polygons.

Algorithm Software Test Description:

The test cases herein acknowledge the requirements which state, "the system shall include a sorting algorithm which is more efficient than the original," and "the system will provide radio buttons to assign the applied sorting algorithm." The resulting software will be unit tested, and integrated with the other system requirements, in the automatic segregator component of Parable of Polygons.

7 Notes

This section shall contain any general information that aids in understanding this document (e.g., background information, glossary, rationale). This section shall include an alphabetical listing of all acronyms, abbreviations, and their meanings as used in this document and a list of any terms and definitions needed to understand this document.

A. Appendixes

Appendixes may be used to provide information published separately for convenience in document maintenance (e.g., charts, classified data). As applicable, each appendix shall be referenced in the main body of the document where the data would normally have been provided. Appendixes may be bound as separate documents for ease in handling. Appendixes shall be lettered alphabetically (A, B, etc.).