Software Test Description (STD)

Software It Counts (SWIC)

CMSC 447

Updated on November 20, 2015

Table of Contents

1 Scope

This section shall be divided into the following paragraphs.

1.1 Identification

For this project, we will be working with the web application, Parable of the Polygons. This software simulates segregation levels between two different shapes, allowing them to move to different locations on the board in an attempt to be happy. The website itself is programmed in HTML while the application is programmed in JavaScript.

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 testing of these requirements in order to observe their completion or not. The subsequent test plans for the project will thoroughly detail how each requirement is to be handled. This document will (like all documents) be uploaded to the project's repository at https://github.com/Kirkas1/polygons.

1.3 Document overview

This document will test all the requirements we must fulfil for the project as previously discussed by Professor Cain. The test plan will have three software tests to address all of the requirements on an individual basis. It will also discuss the preparation, input, and expected outputs of all the tests to be carried out. The test procedure and handling shall be carried out by only the 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 Test preparations

This section shall be divided into the following paragraphs. Safety precautions, marked by WARNING or CAUTION, and security and privacy considerations shall be included as applicable.

3.1 (Project-unique identifier of a test)

This paragraph shall identify a test by project-unique identifier, shall provide a brief description, and shall be divided into the following subparagraphs. When the information required duplicates

information previously specified for another test, that information may be referenced rather than repeated.

3.1.1   Hardware preparation

This paragraph shall describe the procedures necessary to prepare the hardware for the test. Reference may be made to published operating manuals for these procedures. The following shall be provided, as applicable:

- a. The specific hardware to be used, identified by name and, if applicable, number
- b. Any switch settings and cabling necessary to connect the hardware
- c. One or more diagrams to show hardware, interconnecting control, and data paths
- d. Step-by-step instructions for placing the hardware in a state of readiness

3.1.2   Software preparation

This paragraph shall describe the procedures necessary to prepare the item(s) under test and any related software, including data, for the test. Reference may be made to published software manuals for these procedures. The following information shall be provided, as applicable:

- a. The specific software to be used in the test
- b. The storage medium of the item(s) under test (e.g., magnetic tape, diskette)
- c. The storage medium of any related software (e.g., simulators, test drivers, databases)
- d. Instructions for loading the software, including required sequence
- e. Instructions for software initialization common to more than one test case

3.1.3   Other pre-test preparations

This paragraph shall describe any other pre-test personnel actions, preparations, or procedures necessary to perform the test.

4   Test descriptions

4.1   (Project-unique identifier of a test)

Red Circle Test will be used to identify the test through observation.

The Random Algorithm Test will be used to ensure that the algorithm is functional.

The Collective Happiness Test will be used to verify the algorithm is correct.

The Slider Test is used to identify the test to verify the added sliders.

4.1.1   (Project-unique identifier of a test case)

Red Circle Test – observation of the red circle within the system.

The Random Algorithm Test – Execute the algorithm on the fresh system and observe the movement of the polygons.

The Collective Happiness Test – Execute the algorithm on the system and observe the movement of the polygons.

The Slider Test – Move the slider in different combinations verifying through observation that each function correctly.

The Radio Buttons Test – Select each button algorithm (Random algorithm selected by default) and observe that the algorithm actually changed.

4.1.1.1 Requirements addressed

The Random Algorithm Test – addresses the requirement of implementing the option for the user to select the system to randomly sort itself until a valid state is found

The Collective Happiness Test – addresses the requirement of implementing the option for the user to select the system to sort itself based on the Collective Happiness algorithm.

The Slider Test – addresses the requirement of modifying the sliders currently in the system to accommodate for the addition of the red circle.

The Radio Buttons Test – addresses the requirement of implementing uniquely, selectable radio buttons, one for each of the three algorithms.

4.1.1.2 Prerequisite conditions

The computer must have installed and able to run Internet Explorer, Firefox, or Chrome.

4.1.1.3 Test inputs

This paragraph shall describe the test inputs necessary for the test case. The following shall be provided, as applicable:

a. Name, purpose, and description (e.g., range of values, accuracy) of each test input
b. Source of the test input and the method to be used for selecting the test input
c. Whether the test input is real or simulated
d. Time or event sequence of test input
e. The manner in which the input data will be controlled to:
   1) Test the item(s) with a minimum/reasonable number of data types and values
   2) Exercise the item(s) with a range of valid data types and values that test for overload, saturation, and other "worst case" effects
   3) Exercise the item(s) with invalid data types and values to test for appropriate handling of irregular inputs
   4) Permit retesting, if necessary

4.1.1.4 Expected test results

For the testing of the red circle, the project's game board interface shall contain yellow triangles, blue squares, and the inclusion of red circles. The expected result will be a game board with the new shape (red circle) and its according functionality among the polygons.

For the testing of the three-sided, polygon-behavior slider, the project's user interface shall support the functionality of a custom-made slider used to control the inter-behavior polygon percentages. The slider is expected to have an adjustable behavior setting that allows the game board to understand which shapes are happy together and at what ratio.

For the testing of the "happiness" and "collective happiness" algorithms, the project's user interface shall have those options (along with the default algorithm) in a set of radio buttons near the board simulation. With only one algorithm being allowed to be chosen during a given simulation, the expected result of the "happiness" algorithm will be to adjust polygons, on an individual basis, until they have found a location where they are happy. For the expected result of the "collective happiness" algorithm, the polygons will be examined by their 8 shape groups, and adjust accordingly to make the group (in total) happy. The above algorithms should be carefully examined as the board state changes accordingly to match the previous statements' polygon behaviors, characterized as an observational test result.

For the testing of the Radio Buttons, the project's user interface shall have 3 selectable buttons corresponding to the "Random", "Happiness", and "Collective Happiness" algorithms. The radio buttons

are expected to be uniquely selectable (meaning only one button selected at a time), and that each button shall implement its corresponding algorithm onto the board interface.

4.1.1.5   Criteria for evaluating results
Throughout the testing of the red circle, the polygons of the game board should be exactly yellow triangles, blue squares, and red circles. Any third shape that is not a red circle will be a failure of this requirement. The game board shall work as intended with the other two shapes, and not be broken in any way. The simulation shall also work with the red circle and the other two shapes must acknowledge the existence of red circles in the game board.

During the testing of the customized, three-sided slider, the polygons on the board should accurately model the happiness levels concurrent on the slider. The slider should include exactly triangle-to-circle, circle-to-square, and square-to-triangle happiness ratios. The final output of the simulated game board of polygons should also match the slider's happiness percentages for each shape in the grid. The slider should also not inhibit the user interface or break any functions within the simulation as well. If the ratios within the final result of the simulation are close to the anticipated shape ratios of the slider, then this result will be counted as a successful output.

In the testing of the two new algorithms ("happiness" and "collective happiness"), the algorithms should only be selected one-at-a-time via radio buttons. The "happiness" algorithm should result in the simulation placing shapes in locations where they are happy with their surroundings. The "collective happiness" algorithm should result in the simulation placing shapes from a group of 8 polygons in locations where its neighbors are most happy. The system must not break when either of these algorithms is being tested, otherwise it has failed. The duration of both the algorithms in order to complete the simulation shall not matter, unless the act results in an infinite loop or bugs the interface/board. The observed results will be interpreted by our customer Russ Cain as to whether the algorithm passes the requirements being tested.

In the testing of the radio buttons ("random", "happiness", and "collective happiness"), the buttons should only be selected one-at-a-time and be placed next to the "Start" and "Stop Movin'" buttons on the board interface. The system must not break when any of the buttons are being tested, otherwise it has failed. The radio buttons shall not bug the board interface and must correspond to each specified algorithm it is mapped towards, otherwise it has failed. The observed results will be interpreted by the group's software tester.

4.1.1.6   Test procedure
   a. Test operator actions and equipment operation required for each step:
      1) open webpage, check if the red circle is there
      2) Run the simulation with the red circle, and see if it behaves like the other polygons
      3) If the red circle is there, and behaves properly we have succeeded in adding the red cicle
      4) Open the webpage with the happiness algorithm active
      5) Run the simulation with the happiness algorithm.
      6) If the simulation runs properly, and is than the random, then we have succeeded in implementing the happiness algorithm
      7) Repeat steps 4 through 6 for the collective happiness algorithm
   b. Actions to follow in the event of a program stop or indicated error:
      1) Step through the code line by line
      2) Analyze where the algorithm or circle produced the error

3) Fix that piece of code

4.1.1.7 Assumptions and constraints

This program must be tested in internet explorer as well as chrome and fire fox such that it works properly on each browser. We can assume that every machine trying to run it has javascript installed and the running will be uniform across machines.

5   Requirements traceability

5.1 Red Circle Requirements

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.

5.2 New Algorithm Requirements

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.js component of Parable of Polygons.

6   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.).