Verification and Validation Report: Software Engineering

Team 17, Track a Trace
Zabrain Ali
Linqi Jiang
Jasper Leung
Mike Li
Mengtong Shi
Hongzhao Tan

March 8, 2023

1 Revision History

Date	Version	Notes
March 5	1.0	Modified Symbols, Abbreviations and Acronyms
March 6	1.1	Modified Functional Requirements Evaluation and Nonfunctional Requirements Evaluation
March 7	1.2	Modified Unit Testing and Changes Due to Testing
March 8	1.3	Modified Automated Testing, Trace to Requirements, Trace to Modules and Code Coverage Metrics

2 Symbols, Abbreviations and Acronyms

symbol	description
T	Test
API	Application Programming Interface
csv	Comma Separated Values file type
OSM	OpenStreetMap, a free, open geographic database

See other symbols, abbreviations or acronyms in SRS (Ali et al., 2022).

Contents

List of Tables

List of Figures

3 Functional Requirements Evaluation

3.1 Generation of CSV and SHP Files

1. test-FR1-1:

   Initial State: The system is set up and ready to take over the user's input.

   Input: Valid GPS Points

   Expected Output: Processed Activity Location information in CSV and SHP, and Route Choice Analysis information in CSV and SHP.

   Actual Output: Processed Activity Location information in CSV and SHP, and Route Choice Analysis information in CSV and SHP.

   Result: Pass

3.2 Removal of Invalid Points

1. test-FR2-1:

   Initial State: The system is set up and ready to take over the user's input.

   Input: GPS Points with no redundant or outlying points.

   Expected Output: Processed GPS points with the same number of points.

   Actual Output: Processed GPS points with the same number of points.

   Result: Pass

2. test-FR2-2:

   Initial State: The system is set up and ready to take over the user's input.

   Input: GPS Points with redundant and outlying points.

   Expected Output: Processed GPS points with redundant and outlying points removed.

   Actual Output: Processed GPS points with redundant and outlying points removed.

   Result: Pass

3. test-FR2-3:

   Initial State: The system is set up and ready to take over the user's input.

   Input: GPS Points with only redundant points.

   Expected Output: One processed GPS point.

   Actual Output: One processed GPS point.

   Result: Pass

4. test-FR2-4:

   Initial State: The system is set up and ready to take over the user's input.

   Input: GPS points with only outliers

   Expected Output: No processed GPS points.

   Actual Output: No processed GPS points

   Result: Pass

3.3 Tag Creation for Valid GPS Points

1. test-FR4-1

   Initial State: The system has received valid GPS data and is ready to proceed.

   Input: Valid GPS Points

   Expected Output: GPS points tagged with their respective mode.

   Actual Output: GPS points tagged with their respective mode.

   Result: Pass

3.4 Partition of GPS Trajectories

1. test-FR5-1

   Initial State: The system has divided processed GPS points into trajectories and is ready to proceed.

   Input: Valid GPS Points

Expected Output: GPS points partitioned by mode.

Actual Output: GPS points partitioned by mode.

Result: Pass

3.5 Extraction of Trip Segments

1. test-FR6-1

   Initial State: The system has classified GPS point segments partitioned from trajectories.

   Input: Valid GPS Points

   Expected Output: GPS points of only mode Drive.

   Actual Output: GPS points of only mode Drive.

   Result: Pass

3.6 Extraction of Activity Locations

1. test-FR7-1

   Initial State: The system has classified GPS point segments partitioned from trajectories.

   Input: Valid GPS Points

   Expected Output: GPS points of only mode Stop.

   Actual Output: GPS points of only mode Stop.

   Result: Pass

3.7 Generation of Alternative Routes

1. test-FR8-1

   Initial State: The system has classified GPS point segments partitioned from trajectories.

   Input: Valid GPS Points

   Expected Output: Route of the extracted trip segments.

   Actual Output: Route of the extracted trip segments.

   Result: Pass

3.8 Generation of Activity Locations with Additional Information

1. test-FR9-1

   Initial State: The GPS data from stop segment, LU and PAL data that is needed for test has been prepared, the necessary Python libraries have been installed on local machine and ready to run the system that will be tested

   Input: Valid GPS Points, LU and PAL data fetched from OSM API

   Expected Output: The output SHP file should match the predetermined SHP file with the correct activity locations with additional information. If the output does not match, the nearest buildings that are determined for the input GPS points should be no further than 200 meters away from the GPS points.

   Actual Output: The output SHP file did not match the predetermined expected outputOut of 114 GPS points from the input, only 14 of those had their nearest buildings within 200 meters of distance from them

   Result: Fail

3.9 Generation of Error Messages

1. test-FR10-1

   Initial State: The system is set up and ready to take over user's input.

   Input: Invalid Data Format

   Expected Output: Raise InvalidDataException Error

   Actual Output: Raise InvalidDataException Error

   Result: Pass

2. test-FR10-2

   Initial State: The system is set up and ready to take over user's input.

   Input: Invalid File Format

   Expected Output: Raise InvalidFileFormatException Error

   Actual Output: Raise InvalidFileFormatException Error

   Result: Pass

3.10 Classification of GPS Points Segments

1. test-FR11-1

   Initial State: The system has partitioned trajectories into segments and is ready to proceed.

   Input: Valid GPS Points

   Expected Output: GPS points partitioned by mode.

   Actual Output: GPS points partitioned by mode.

   Result: Pass

3.11 Assignment of Values to RCA Variables

1. test-FR12-1

   Initial State: The system has already generated alternative routes for trip segments.

   Input: Valid GPS Points

   Expected Output: CSV file with assigned RCA variables for each alternative route generated.

   Actual Output: CSV file with assigned RCA variables for each alternative route generated.

   Result: Pass

4 Nonfunctional Requirements Evaluation

Many of the tests for non-functional requirements refer to a sample csv 'sample-gps-1.csv'. This csv can be found here.

The VnV Plan and many of the tests for the non-functional requirements also refer to a survey given to the supervisor, found here.

4.1 Look and Feel

1. test-LF-1

Initial State: The system was installed and run using sample-gps-1.csv and 500 points.

Input: The user was asked if the system's interface is easy to follow

Output: The supervisor was satisfied with the easiness of interaction with the interface

Result: Pass

2. test-LF-2

   Initial State: The system was installed and run using sample-gps-1.csv and 500 points.

   Input: The user was if the system's interface makes it easy to determine the required input

   Output: The supervisor was satisfied with the easiness of determination of the input

   Result: Pass

3. test-LF-3
   Initial State: The system was installed and run using sample-gps-1.csv and 500 points.

   Input: The user will be asked if the system's interface gave a clear indication of what stage the system was in while the system was generating an output

   Output: The supervisor says it is easy to visualize the stage of the system from it's interface when the system is generating output

   Result: Pass

4.2 Usability

1. test-UH-1

   Initial State: The system was available for download online. The user installed the system

   Input: The user was asked about their programming experience level, and if they were able to install the system without asking for help

Output: The supervisor says that the process of downloading the system is smooth, and the guidance is easy to follow.

Result: Pass

2. test-UH-2

   Initial State: The system was installed and run using sample-gps-1.csv and 500 points.

   Input: The user was asked if they were able to generate an output with given sample inputs

   Output: The supervisor said that the process of downloading the system is smooth, and the guidance is easy to follow.

   Result: Pass

4.3 Performance

1. test-PR-1

   Initial State: The system is was set up and ready to take the user's input.

   Input: The system was run using sample-gps-1.csv and 500 points.

   Output: The time it took for the system to print that it had entered mode detection stage was about 5 seconds. Since this is less than 20 seconds, this test was marked as passed.

   Result: Pass

2. test-PR-2

   Initial State: The system is was set up and ready to take the user's input.

   Input: The system was run using using sample-gps-1.csv and 500 points at two different times.

   Output: The geojson files, csv files, shp files generated by the two different runs were verified by to be the same by the developers. Since they were the same, the test was marked as passed.

   Result: Pass

4.4 Operational and Environmental

1. test-OE-1

   Initial State: The system was run using sample-gps-1.csv and 500 points

   Input: The supervisor was asked for their operating system and if they were able to run the system without any issue.

   Output: The supervisor reported that they were able to run the system, and that they were using a Windows computer to do so. The developers used Linux and Mac to run the system, and were able to get the same output as the supervisor. Thus, the test was marked as passed.

   Result: Pass

2. test-OE-2

   Initial State: The system was downloaded in a Python virtual environment with the required packages installed.

   Input: The system was run in the Python virtual environment using sample-gps-1.csv and 500 points.

   Output: The developers verified that running the system in the virtual environment produced the expected output. The test was marked as passed.

   Result: Pass

3. test-OE-3

   Initial State: The system was downloaded in a Python virtual environment without the required packages installed.

   Input: The system was run in the Python virtual environment using sample-gps-1.csv and 500 points.

   Output: The developers verified that a list of dependencies was provided inside a file in the folder. The developers then installed these dependencies into the virtual environment, ran the system, and verified that the system produced the expected output. The test was marked as passed.

   Result: Pass

4.5 Maintainability and Support

1. test-MS-1

   Initial State: The system is was set up and ready to take the users' inputs.

   Input: The system was run using using sample-gps-1.csv and 500 points by two developers on their own systems.

   Output: The geojson files, csv files, shp files generated by the two different runs were verified by to be the same by the developers. Since they were the same, the test was marked as passed.

   Result: Pass

4.6 Security

1. test-SR-1

   Initial State: Revision 0 of the system was completed.

   Input: The developers were given the system to review.

   Output: The developers confirmed during the code review session that there was no code that accessed the user's personal information. The test was marked as passed because of this.

   Result: Pass

2. test-SR-2

   Initial State: Revision 0 of the system was completed.

   Input: The developers were given the system to review.

   Output: The developers confirmed during the code review session that there was no code that accessed files outside of the user-provided inputs and files included in Python and the system. The test was marked as passed because of this.

   Result: Pass

4.7 Legal

1. test-LR-1

Initial State: Revision 0 of the system was completed.

Input: The developers were given the system to review.

Output: The developers confirmed during the code review session that there was no code that used packages or licenses that require libraries. The test was marked as passed because of this.

Result: Pass

5 Unit Testing

5.1 GPS Data Preprocessing Module

1. test_get_data

   Initial State: The system is set up and ready to take over the user's input.

   Input: sample-gps-1.csv Valid GPS Data Points

   Output: GeoDataFrame of processed GPS Points

   Result: Pass

2. test_filter_data

   Initial State: The system is set up and ready to take over the user's input.

   Input: sample-gps-1.csv Valid GPS Data Points only and sample-gps-8.csv Outliers GPS Data Points only

   Output: GeoDataFrame of processed GPS Points with the correct latitude, longitude, and geometry column with correct points

   Result: Pass

3. test_filter_data_redundant_only

   Initial State: The system is set up and ready to take over the user's input.

   Input: sample-gps-6.csv Redundant GPS Data Points only

   Output: GeoDataFrame of processed GPS Points with the correct latitude, longitude, and geometry column with correct points

   Result: Pass

4. test_filter_data_redundant_and_outliers

   Initial State: The system is set up and ready to take over the user's input.

   Input: sample-gps-7.csv Valid, Redundant and Outliers GPS Data Points

   Output: GeoDataFrame of processed GPS Points with the correct latitude, longitude, and geometry column with correct points

   Result: Pass

5. test_smooth_data

   Initial State: The system is set up and ready to take over the user's input.

   Input: sample-gps-1.csv Valid GPS Data Points only

   Output: GeoDataFrame of processed GPS Points with no points with a Speed_kmh greater than 50m/s or 180km/h

   Result: Pass

6. test_smooth_data_redundant_and_outliers

   Initial State: The system is set up and ready to take over the user's input.

   Input: sample-gps-7.csv Valid, Redundant, and Outliers GPS Data Points

   Output: GeoDataFrame of processed GPS Points with no points with a Speed_kmh greater than 50m/s or 180km/h

   Result: Pass

7. test_smooth_data_outliers_only

   Initial State: The system is set up and ready to take over the user's input.

   Input: sample-gps-8.csv Outliers GPS Data Points only

   Output: GeoDataFrame of processed GPS Points with no points with a Speed_kmh greater than 50m/s or 180km/h

   Result: Pass

5.2 ModeDetection Module

1. test_distance

   Initial State: The system is set up and ready to take over the user's input.

   Output: Checks if the distance between two different longitude/latitude coordinates is valid (matches what is found using this calculator and also checks if the distance between the same point is 0.

   Result: Pass

5.3 Extractor Module

1. test_fill_points

   Initial State: The system is set up and ready to take over the user's input.

   Input: Valid GPS Points

   Expected Output: GPS points tagged with their respective mode.

   Actual Output: GPS points tagged with their respective mode.

   Result: Pass

2. test_extract_trip_segments

   Initial State: The system is set up and ready to take over the user's input.

   Input: Valid GPS Points

   Expected Output: GPS points of only mode Drive.

   Actual Output: GPS points of only mode Drive.

   Result: Pass

3. test_extract_activity_locations

   Initial State: The system is set up and ready to take over the user's input.

   Input: Valid GPS Points

   Expected Output: GPS points of only mode Stop.

Actual Output: GPS points of only mode Stop.

Result: Pass

5.4 Route Choice Set Generator Module

1. test_map_point
   Initial State: The system is set up to be run. Test data for GPS points of trip segment and transportation network is ready to be used.

   Input: Valid GPS Points of Trip Segment, Transportation Network Data

   Output: The GPS points are matched to the correct streets

   Result: Pass

2. test_detect_and_fill_gap
   Initial State: The system is set up to be run. Test data for GPS points of trip segment and transportation network is ready to be used.

   Input: Valid GPS Points of Trip Segment, Transportation Network Data

   Output: The predetermined gaps(i.e. 2 consecutive GPS points that are more than 50 meters away from each other and not on the same line segment, or 2 consecutive GPS points that are not on the same street by street name) in the input GPS points have all been detected and the gaps are correctly filled by shortest path between the GPS points that involved.

   Result: Pass

3. test_route_choice_gen
   Initial State: The system is set up to be run. Test data for GPS points of trip segment and transportation network is ready to be used.

   Input: Valid GPS Points of Trip Segment, Transportation Network Data

   Output: A route that fits the GPS points in the trip segment

   Result: Pass

5.5 Route Choice Analysis Variables Generator Module

1. test_var_gen Initial State: The system is set up to be run. Test data for GPS points of trip segment and transportation network is ready to be used.

   Input: Valid GPS Points of Trip Segment, Transportation Network Data

   Output: The generated RCA variable has the correct value for each attribute

   Result: Pass

5.6 Activity Location Identification Module

1. test_identify_lu

   Initial State: The system is set up to be run. Test data for activity location information and valid GPS Points of Stop Segment.

   Input: Valid GPS Points of Stop Segment,Test data for activity location information

   Expected Output: The activity location landuse gdf has the correct value in lu_classification and lu_index

   Actual Output: The values are matched

   Result: Pass

2. test_identify_pal

   Initial State: The system is set up to be run. Test data for activity location information and valid GPS Points of Stop Segment

   Input: Valid GPS Points of Stop Segment,Test data for activity location information

   Expected Output: The distance between building points in GPS stop Segment and building polygons are less than 100

   Actual Output: All points are within 100 meters of the polygons

   Result: Pass

6 Changes Due to Testing

6.1 Change due to supervisor's feedback

Dr. Paez, who is the supervisor of this project, has pointed out during the Revision 0 demo that rather than finding the nearest building for every single GPS point in stop segment(s), it would be better for the product to find all the buildings that are within certain distance to the points of the stop segment(s). To follow the feedback of Dr. Paez, the implementation and design of the Activity Locations Identification module will be changed so that the corresponding method in the module will output all the buildings that are within a distance from user's input in unit of meters(100 meters by default) to the points of the stop segment(s). An input prompt will be added in the Main module to ask for user's input for the distance.

6.2 Change due to reviewing the questions in the usability survey

In the VnV plan, it states that our group will review the questions in the usability survey as part of non-functional testing. One of the questions in the usability related to how easy it was to determine the current state of the system, and that was used for requirement and test LF-3. While the supervisor responded that it was easy to determine the stage of the system (pre-processing, mode detection, etc.) from the print statements being shown in the console, the team decided that this state could be more clear, since the individual progress of each stage of the system is not shown. The team decided that adding a progress bar to the interface of the system would give a better indication of the current stage of the system, rather than just print statements.

6.3 Change due to failing system test test-FR9-1

The reason of failing the test-FR9-1 has been found to be that within the PAL data that is fetched from OSM API, lots of the buildings that are around the input GPS points had neither addresses nor names of the building included in the fetched PAL data, while the method in the Main module for fetching PAL data from OSM API has been implemented to filter out all the data of buildings that have neither address nor name information. Hence,

the nearest buildings that was predetermined for the GPS points were not recognized by the Activity Locations Identification module. To fix the issue, the implementation of the method for fetching PAL data will be changed so that the data of buildings that have neither address nor name specified in OSM will be kept to be used by the Activity Locations Identification module.

7 Automated Testing

The automated testing framework being used is PyTest. PyTest allows created test cases to be as simple as writing a Python Function. Statements such as "assert" are used to verify if the expected output is the same as the actual output. PyTest also enables test parametrization at several levels with the use of the builtin pytest.mark.parametrize decorator. This has made the experience with automated testing much faster and easier to use.

8 Trace to Requirements

Test ID	R1	R2	R4	R5	R6	R7	R8	R9	R10	R11	R12
test-FR1-1	×										
test-FR2-1		×									
test-FR2-2		×									
test-FR2-3		×									
test-FR2-4		×									
test-FR4-1			×								
test-FR5-1				×							
test-FR6-1					×						
test-FR7-1						×					
test-FR8-1							×				
test-FR9-1								×			
test-FR10-1									×		
test-FR10-2									×		
test-FR11-1										×	
test-FR12-1											×

Table 1: **Functional Requirements Traceability**

Test ID	LF1	LF2	UH1	UH2	PR1	PR3	OE1	OE2	OE3
test-LF-1	×								
test-LF-2	×								
test-LF-3		×							
test-UH-1			×						
test-UH-2				×					
test-PR-1					×				
test-PR-2						×			
test-OE-1							×		
test-OE-2								×	
test-OE-3									×

Table 2: **Non-Functional Requirements Traceability Part 1**

Test ID	MS1	MS2	MS3	SR1	SR2	SR3	LR1
test-MS-1			×				
test-SR-1				×			
test-SR-2					×		
test-LR-1							×

Table 3: **Non-Functional Requirements Traceability Part 2**

9 Trace to Modules

Test ID	M2	M3	M4	M5	M6	M7	M8
test-FR1-1							×
test-FR2-1	×						
test-FR2-2	×						
test-FR2-3	×						
test-FR2-4	×						
test-FR4-1		×					
test-FR5-1		×					
test-FR6-1			×				
test-FR7-1			×				
test-FR8-1				×			×
test-FR9-1						×	×
test-FR10-1							×
test-FR10-2							×
test-FR11-1		×					
test-FR12-1					×		×

Table 4: **Implemented Modules Traceability Part 1**

Test ID	M2	M3	M4	M5	M6	M7	M8
test-LF-1							×
test-LF-2							×
test-LF-3							×
test-UH-1	×	×	×	×	×	×	×
test-UH-2							×
test-PR-1							×
test-PR-2							×
test-OE-1	×	×	×	×	×	×	×
test-OE-2							
test-OE-3							
test-MS-1	×	×	×	×	×	×	×
test-SR-1	×	×	×	×	×	×	×
test-SR-2	×	×	×	×	×	×	×
test-LR-1							

Table 5: **Implemented Modules Traceability Part 2**

References

Zabrain Ali, Linqi Jiang, Jasper Leung, Mike Li, Mengtong Shi, and Hongzhao Tan. Software requirements specification for software engineering: Pyert, 2022. URL https://github.com/paezha/PyERT-BLACK/blob/main/docs/SRS/SRS.pdf.

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection. Please answer the following question:

1. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)

   (a) One change that we made to the VnV plan was the test group for the usability survey. Originally, our plan was to have at least 10 test users run the system, and give feedback. We decided to change this to just having the supervisor do the survey, because once we completed our system, we felt that it would be hard to get valuable feedback from users not familiar with the premise of the system, and it would take too much time to introduce each user to the system. We decided that feedback from the supervisor would be more relevant and easy to obtain. In future projects, we will make sure to take into account how complex the system is before using usability surveys as a form of non-functional testing, to anticipate whether new users will be able to use the system easily and provide relevant feedback.

   (b) Another change that we made to the VnV plan was testing over 1 million unique points during the preprocessing phase. We decided after talking with our supervisor it was not necessary to test that many points, nor was there a valid sample gps data for us to use. In future projects, we will set our testing goals to be more realistic and discuss with any supervisor beforehand what is a feasible data set to use for accurate testing.

   (c) A change that we have made to the VnV plan was we have removed the tests for requirement 3 which was to cluster GPS points into 24-hour trajectories because the requirement has been removed.

We made such decision after talking with the supervisor of the project Dr. Paez that it was necessary to GPS data that goes across multiple days and there was no such long valid sample GPS data for us to use. In future projects, we will set our requirements and testing goals to be more realistic and discuss with any supervisor beforehand about what requirements the stakeholders would have and what is a feasible data set to use for accurate testing.