Software Requirements Specification for Software Engineering: PyERT

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

November 14, 2022

Contents

List of Tables

List of Figures

Date	Version	Notes
05/10/2022	1.0	Initial Software Requirements Specification
02/11/2022	1.1	Modified Functional Requirements and Naming Conventions
14/11/2022	1.2	Modified Functional Requirements

Table 1: **Revision History**

This document describes the requirements for PyERT. The template for the Software Requirements Specification (SRS) is a subset of the Volere template (Robertson and Robertson, 2012) with sections for System Description and Reflection added into it.

1 Project Drivers

1.1 Naming Conventions and Terminology

Acronym/Abbreviation	Definition
GIS	Geographic Information System
GERT	GIS-based Episode Reconstruction Toolkit
ArcGIS	A licensed GIS service used by GERT for data processing
PyERT	Python-based Episode Reconstruction Toolkit
Req.	Requirement
NFR	Non-Functional Requirement
CSV	Comma Separated Values
GPS	Global Positioning System
TUD	Time Use Diary: Survey data containing a chronological sequence of activities that respondents did over a time period
m/s	meters per second
SHP	shapefile: A data format for spreadsheet
LU	Land Use
PAL	Potential Activity Locations
CI/CD	Continuous Integration / Continuous Deployment
RCA	Route Choice Analysis

Table 2: **Naming Conventions and Terminology**

1.2 The Purpose of the Project

This project aims to re-implement the features in GERT (Dalumpines and Scott, 2018) that use ArcGIS Pro packages with open-source packages and libraries, and remove any use of ArcGIS in the project. Using open-source packages and libraries instead of ArcGIS will make GERT tools accessible. The changes that were made to GERT when replacing ArcGIS will be documented, as having detailed documentation for the modified GERT will ensure that existing users of GERT will be able to transition from using the ArcGIS-free version easily and understand the changes made, and will also be helpful for new users. The current project structure will be modified to be more organized and readable, because most of the code for the current project is stored in a single file, making it hard to read. Organizing the code and potentially splitting up the code into separate modules would be helpful for any users trying to use the code.

1.3 The Stakeholders

The stakeholders of this project include The School of Earth, Environment and Society of McMaster University, the supervisor of the project, Dr. Antonio Paez, 4G06 Capstone Project course teaching assistants, and the professor, Dr. Spencer Smith.

1.3.1 The Client

The clients of this project consists of The School of Earth, Environment and Society of McMaster University and the supervisor of the project, Dr. Antonio Paez. The primary concern for the clients is to see that the outcome of the project is an open-source and potentially upgraded version of the original GERT toolkit.

1.3.2 The Customers

The target demographic of this project is aimed at potential users who would develop projects based on GERT or refer to it, members of the school of The Earth, Environment and Society of McMaster University, and the supervisor of the project, Dr. Antonio Paez.

1.3.3 Other Stakeholders

There are many other stakeholders involved in this project. The development team is a major stakeholder, as they are responsible for re-implementing and upgrading the original GERT toolkit. The development team has their own desires for what the overall outcome should be, some of which include technical aspects such as efficiency, run time, organizing the structure of the project and stylizing the code.

1.4 Mandated Constraints

The main mandated constraints for this project involve the due dates for project deliverables. The final version of the project must not include usage of any package or library that need a license to be accessed. As this project is based off an existing toolkit, GERT, the inputs and outputs of this project must match the inputs and outputs of GERT. As such, this project shall only accept GPS or TUD data in CSV format, and shall return only CSV or SHP files after processing the user input if there are no errors.

1.5 Relevant Facts and Assumptions

1.5.1 Relevant Facts

The goal of the original GERT toolkit is to break the limit of previously developed procedures in terms of scalability, modularity and transferability. With this effective toolkit, we can take full advantage of GPS data for the following reasons: (1) it is difficult to adopt tools developed by other researchers because of the lack of transferability, (2) the limited ability to derive more information from GPS data for lack of an integrated set of modules, and (3) high computational costs and lack of automatic procedures in dealing with large data-sets.

1.5.2 Assumptions

- The project assumes the users will directly run the compiled .pyc files of the source code .py files with command line in console to use the product.

2 System Description

2.1 System Context

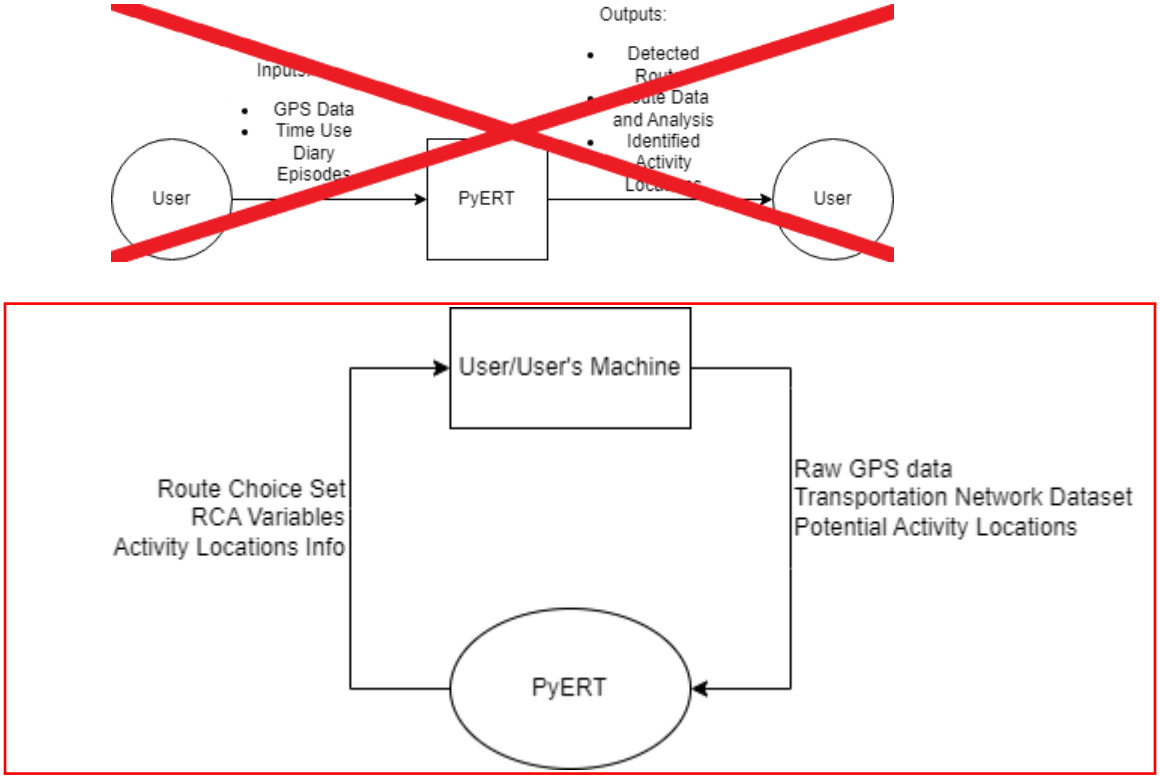

Figure 1: ~~Context Diagram~~ **Updated Context Diagram**

- User responsibilities:
  - Input a data-set containing GPS information in CSV format (mandatory) ~~and TUD episodes in CSV format (optional)~~.
  - Ensure the general format of the input data is correct.
- PyERT responsibilities:
  - ~~Filter faulty TUD episodes and GPS data.~~
  - Return detected routes, generated data for those routes, and identified potential activity locations in the correct format.

2.2 Normal Operation

This program is used to process geographical data. Users will use the program on a computer, which allows them to view and save the processed data. Users will then be able utilize the processed data in any manner they choose.

2.3 Behaviour Overview

The system requires user input in order to begin processing and returning data. The program will accept geographical data in the form of a CSV file. The user must provide a CSV file containing GPS points, and can optionally provide a CSV file with ~~TUD data~~ and SHP files with data on LU and PAL. ~~If no TUD CSV file is provided, the program will use its own generated data during processing. If a TUD CSV file is provided, the program will use the data in the provided file during processing~~. The program will filter faulty GPS points, and will return processed data on route choices in SHP and CSV files. Activity location data will also be returned with LU and PAL data appended if they were provided, in the form of a SHP file.

2.4 Undesired Event Handling

If an undesired event occurs during operation of the program, the program shall enter an error state. The program shall not process any data in this state to ensure corrupted or incorrect data is not used, and that it does not corrupt the user's inputted data. A reason for the error will be displayed so that the user understands the undesired event and potentially avoids it in the future. The program will need to be run again and will require the user to re-input their data, to ensure the undesired event does not affect the programs results.

2.5 User Characteristics

- The end user of PyERT should have an understanding of basic geographic concepts (GPS, latitude, longitude, etc.).
- The end user of PyERT should know the input formats of the GPS data-sets and ~~TUD Episodes~~.
- The end user of PyERT should understand the relevance of the GPS data-sets ~~and Time Use Diaries~~ being input into the program.
- The end user of PyERT should have an understanding of the Route Data and activity analysis provided by PyERT.

3 Functional Requirements

3.1 The Scope of the Work and the Product

3.1.1 The Context of the Work

The plan for the project is to re-implement the features of the original GERT toolkit, where packages require an ArcGIS Pro license, and implement it with open-source packages and libraries so that the toolkit can be independent from ArcGIS Pro. The re-implemented version of the toolkit (PyERT) should inherit ~~all functionalities of the original GERT toolkit~~ the main functionalities of the original GERT toolkit. ~~In addition, the structure of the implementation of the toolkit shall be reorganized into modules that are more readable and understandable for users who are trying to access the code~~. The plan for the project is to re-implement the features of the original GERT toolkit, where packages require an ArcGIS Pro license, and implement it with open-source packages and libraries so that the toolkit can be independent from ArcGIS Pro. The re-implemented version of the toolkit (PyERT) should inherit ~~all functionalities of the original GERT toolkit~~ the main functionalities of the original GERT toolkit. ~~In addition, the structure of the implementation of the toolkit shall be reorganized into modules that are more readable and understandable for users who are trying to access the code~~.

3.1.2 Work Partitioning

Event ID	Event Name	Summary
BE1	~~Matching GPS Data Point to Network with TUD Episodes Input~~	~~User executes PyERT toolkit using command line in console with input GPS data and TUD Episodes.~~
BE2	Matching GPS Data Point to Network without TUD Episodes Inputs	User executes PyERT toolkit using command line in console with input GPS data only.
BE3	Invalid Input from User	User executes PyERT toolkit using command line in console with invalid input.

Table 3: **Work Partitioning Part 1**

Event ID	Input	Output
BE1	User's GPS data, ~~TUD Episodes in CSV format~~ and LU and PAL information in SHP format	Route Choice Set and Activity Locations' Information in SHP file type and RCA Variables in CSV format
BE2	User's GPS data in CSV format and LU and PAL information in SHP format	Route Choice Set and Activity Locations' Information in SHP file type and RCA Variables in CSV format
BE3	User's data not in CSV format or user's CSV file but with data that cannot be processed by PyERT	Descriptive error message shall be shown to the user

Table 4: **Work Partitioning Part 2**

3.1.3 Individual Product Use Cases

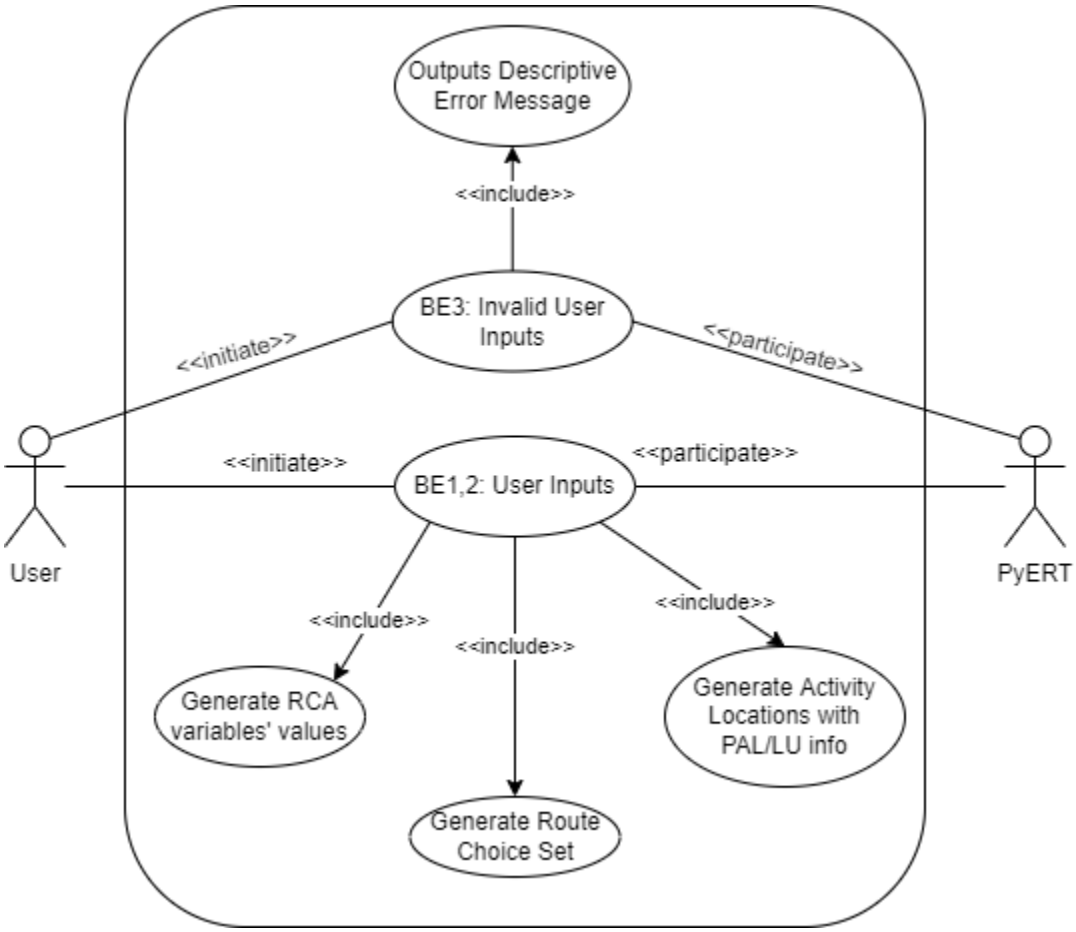

Figure 2: **Use Case Diagram**

3.2 Functional Requirements

Req. #: 1 **Req. Type**: Functional
Description: The product shall be able to read and generate CSV and SHP files.
Rationale: Since the product is a re-implemented version of the GERT toolkit, it shall be able to take and generate the same types of files the GERT toolkit used to take and generate as its inputs and outputs respectively.
Organizer: Software Engineer
Fit Criterion: After the user executes the product with valid input data in CSV and SHP formats, the product will be able to read the data from the input and generate required CSV and SHP files after processing the input data.
Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A
Priority: 1 **Dependencies**: None **Conflicts**: None
Supporting Materials: BE1, BE2
History: Created October 5, 2022
Plan to Complete by: January 6, 2023

Req. #: 2 **Req. Type**: Functional
Description: The product shall remove invalid GPS points from user's input. Invalid points include redundant points (points with the same coordinates) and outliers (with speed $\geq$ 50m/s).
Rationale: Since there could be some invalid GPS points in users' input GPS data which could affect the result of matching GPS points to network or the performance of the product. The requirement is needed for the product to determine and remove the invalid GPS points before actually processing them.
Organizer: Software Engineer
Fit Criterion: The product will remove points with the same coordinates and points that are with speed $\geq$ 50m/s.
Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A
Priority: 2 **Dependencies**: None **Conflicts**: None
Supporting Materials: BE1, BE2
History: Created October 5, 2022
Plan to Complete by: January 12, 2023

~~**Req. #**: 3 Requirement Removed~~ ~~**Req. Type:**~~ ~~Functional~~

~~**Description:**~~ ~~The product shall be able to divide GPS points into 24-hour trajectories and further divide the trajectories into clusters of adjacent points based on speed, distance, heading and change-in-heading thresholds.~~

~~**Rationale:**~~ ~~The requirement is needed since the GPS points from users' input need to be divided into clusters that represents the movement of each person-day or of even shorter time period so that the points can be further processed by the product.~~

~~**Organizer:**~~ ~~Software Engineer~~

~~**Fit Criterion:**~~ ~~The product will divide input GPS points into clusters of adjacent points based on predefined thresholds.~~

~~**Costumer Satisfaction:** N/A~~ ~~**Costumer Dissatisfaction:** N/A~~

~~**Priority:** 2~~ ~~**Dependencies:** None~~ ~~**Conflicts:** None~~

~~**Supporting Materials:** BE1, BE2~~

~~**History:** Created October 5, 2022~~

~~**Plan to Complete by:** January 12, 2023~~

Req. #: 4 **Req. Type**: Functional

Description: The product shall tag each valid input GPS point as a stop (stationary) point or a trip (moving) point.

Rationale: The requirement is necessary for the product to further classify the GPS trajectories into trip (moving) and stop (stationary) segments.

Organizer: Software Engineer

Fit Criterion: All valid input GPS points will be tagged as a stop point or trip point after being clustered.

Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A

Priority: 3 **Dependencies**: Req. #: 2, Req. #: 3 **Conflicts**: None

Supporting Materials: BE1, BE2

History: Created October 5, 2022

Plan to Complete by: January 18, 2023

Req. #: 5 **Req. Type**: Functional

Description: The product shall partition valid GPS trajectories into segments ~~and classify the segments into stationary activity (stop) episodes and travel (moving) episodes~~.

Rationale: The requirement is needed to fully capture the valuable information from the perspective of activity analysis.

Organizer: Software Engineer

Fit Criterion: ~~Modules that can automatically capture land interest points details will be developed.~~ The product will generate partitioned segments given valid GPS trajectories.

Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A

Priority: 4 **Dependencies**: Req. #: 4 **Conflicts**: None

Supporting Materials: BE1, BE2

History: Created October 5, 2022

Plan to Complete by: January 24, 2023

Req. #: 6 **Req. Type**: Functional

Description: The product shall extract trip segments (sequences of GPS points in travel episode) from valid GPS trajectories.

Rationale: The requirement is needed to fully capture the valuable information from the perspective of activity analysis.

Organizer: Software Engineer

Fit Criterion: ~~The availability of huge GPS data and the high potential to collect more make it a necessary procedure that can automatically extract information from these data.~~ The product will extract all the trip segments if the GPS trajectories are valid.

Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A

Priority: 5 **Dependencies**: Req. #: 5 **Conflicts**: None

Supporting Materials: BE1, BE2

History: Created October 5, 2022

Plan to Complete by: January 31, 2023

Req. #: 7 **Req. Type**: Functional

Description: The product shall extract activity locations (GPS points in stop episode or end points of trip segments) from valid GPS trajectories.

Rationale: The requirement is necessary for the further process to add LU and PAL information to the stationary GPS points.

Organizer: Software Engineer

Fit Criterion: If the user inputs the valid GPS trajectories, the product will extract the correct activity locations for the user.

Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A

Priority: 5 **Dependencies**: Req. #: 5 **Conflicts**: None

Supporting Materials: BE1, BE2

History: Created October 5, 2022

Plan to Complete by: January 31, 2023

Req. #: 8 **Req. Type**: Functional

Description: The product shall generate ~~alternative~~ one route~~s~~ for each trip segments in SHP format with digital road/pedestrian networks ~~and assign values to RCA variables for each of the alternative routes in CSV format based on road network attributes that are provided~~.

Rationale: ~~Both are~~ SHP is a cross-platform data format~~s~~ for ~~spreadsheet and~~ GIS applications~~, respectively;~~ thus, it makes it easy to manipulate GERT's outputs in other applications.

Organizer: Software Engineer

Fit Criterion: ~~GERT is designed to work on minimal inputs with generic variables such as location coordinates and time stamps. This feature allows implementation in environments without additional information aside from GPS data.~~ Given trip segments in SHP format with digital road/pedestrian networks, the product will generate alternative routes.

Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A

Priority: 6 **Dependencies**: Req. #: 6 **Conflicts**: None

Supporting Materials: BE1, BE2

History: Created October 5, 2022

Plan to Complete by: February 6, 2023

Req. #: 9 **Req. Type**: Functional

Description: The product shall generate SHP files for extracted activity locations with additional information about them when the information is available in given spatial data such as LU and PAL data in user's input.

Rationale: SHP is a standard format for spatial data across different GIS platforms,the outputs can be used to visualize activity episodes in a GIS or can be used as inputs to spatial analysis and modeling outside of GERT.

Organizer: Software Engineer

Fit Criterion: The file can be generated from spatial data such as land use and potential activity locations (PAL) using overlay analysis functions in GIS. PAL refers to points of interest that indicate the locations of government offices, shopping destinations, banks, and so on.

Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A

Priority: 6 **Dependencies**: Req. #: 7 **Conflicts**: None

Supporting Materials: BE1, BE2

History: Created October 5, 2022

Plan to Complete by: February 6, 2023

Req. #: 10 **Req. Type**: Functional

Description: The product shall ~~handle run-time errors and~~ generate descriptive error messages to the user when the user's input is invalid (not in CSV and SHP format, or data input cannot be processed by the product).

Rationale: It is necessary for the product to help the user understand the cause of error.

Organizer: Software Engineer

Fit Criterion: If there is a run-time error occurred, a descriptive message about the error will be shown to the user.

Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A

Priority: 1 **Dependencies**: None **Conflicts**: None

Supporting Materials: BE3

History: Created October 5, 2022

Plan to Complete by: January 6, 2023

Req. #: 11 **Req. Type**: Functional
Description: The product shall classify the segments partitioned from GPS trajectories into stationary activity (stop) episodes and travel (moving) episodes.
Rationale: The requirement is needed to fully capture the valuable information from the perspective of activity analysis.
Organizer: Software Engineer
Fit Criterion: The segments will be classified as stationary activity (stop) episodes or travel (moving) episodes.
Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A
Priority: 4 **Dependencies**: Req. #: 4 **Conflicts**: None
Supporting Materials: BE1, BE2
History: Created October 5, 2022
Plan to Complete by: January 24, 2023

Req. #: 12 **Req. Type**: Functional
Description: The product shall assign values to RCA variables in CSV format for each of the alternative routes generated based on the provided road network attributes.
Rationale: CSV is a cross-platform data format for spreadsheet applications; thus, it makes it easy to manipulate GERT's outputs in other applications.
Organizer: Software Engineer
Fit Criterion: Given the generated alternative routes, the product will assign the corresponding values to RCA variables in CSV format.
Costumer Satisfaction: N/A **Costumer Dissatisfaction:** N/A
Priority: 6 **Dependencies**: Req. #: 6 **Conflicts**: None
Supporting Materials: BE1, BE2
History: Created October 5, 2022
Plan to Complete by: February 6, 2023

4 Non-functional Requirements

4.1 Look and Feel Requirements

LF1. The interface of the program should be easy to follow, and clearly specify the user input.

- Rationale: A clear interface ensures that the program is easy to use and navigate through.

- Fit Criterion: At least 90% of the test users for the program will say that the program's interface is clear and concise.

LF2. The program shall give visual indication stating what stage the program is in.

- Rationale: Since the program output is not instant, there needs to be a visual indication of what stage of processing the program is in, to confirm to the user that it is running properly.
- Fit Criterion: At least 90% of the test users for the program shall report that the visual indicators displaying the program progress was helpful in determining what was going on within the program.

4.2 Usability and Humanity Requirements

UH1. The program shall be easy to install.

- Fit Criterion: 90% of users with no programming experience must be able to install the program by themselves.

UH2. The program shall be easy to use.

- Fit Criterion: 95% of users must be able to use the program and process data by themselves.

UH3. ~~The program shall be able to display instructions on how to use it.~~ Requirement removed.

- ~~Rationale: The program should be able to provide users with help, listing available commands and proper file types.~~
- ~~Fit Criterion: A help menu must be displayed to the user whenever a help command is called.~~

4.3 Performance Requirements

PR1. The average time between user input of data and output of processed data shall be relatively low.

- Rationale: To improve the user's experience, the processing delay should be as low as possible.
- Fit Criterion: The processing speed of filtering out invalid GPS points should be at least ~~80 percent as fast as GERT's reported performance of 9100 GPS points per second~~ 400 GPS points per second.

PR2. ~~The program shall be able to process data with up to 50 million GPS points.~~ Requirement Removed

- ~~Rationale: To allow users to use large data-sets.~~

- ~~Fit Criterion: The program successfully process a data-set with 50 million GPS points.~~

PR3. The user shall be able to use the program at any time of the day.

- Rationale: To allow the user to process data anytime that they want.
- Fit Criterion: The program shall be available 24 hours a day, 7 days a week.

4.4 Operational and Environmental Requirements

OE1. The program shall operate on any computer or laptop using an operating system that runs Python.

- Rationale: To provide support to nearly all computer systems so that more users can use the program.
- Fit Criterion: The program shall successfully run on any desktop and laptop using Windows or Linux with Python installed.

OE2. The program shall be able to run without having to download any external software.

- Rationale: In PyERT, the goal is to recreate all the features of GERT using open source Python packages, so external software will not be needed.
- Fit Criterion: When Python is installed on a system, the user shall be able to install GERT and process data without downloading any external programs.

OE3. The program shall list its dependent packages for the user to download.

- Rationale: Since PyERT relies on open source packages, the packages that it uses must be declared for the user to see.
- Fit Criterion: Users will be able to view all the dependent packages in a text file.

4.5 Maintainability and Support Requirements

MS1. Any features implemented in the source code can be easily undone via Git.

- Rationale: In case implemented features cause conflicts, the team may need to recall issues that caused the problem(s) individually.
- Fit Criterion: Every new feature implemented in the program shall be completed in a separate Git Branch.

MS2. Tickets to any issues of the program can be opened on GitHub.

- Rationale: Users should be able to notify the development team of any bugs or missing features.

- Fit Criterion: A test ticket will be raised by a member of the development team for the rest of the team to resolve.

MS3. The program will be portable.

- Fit Criterion: At least 90% of users who used the program on the bus report the same usability as when they used the program at home.

4.6 Security Requirements

SR1. The program shall not store a user's personal information.

- Rationale: To ensure that a user's information and privacy is maintained.
- Fit Criterion: When reviewing the code, the team will verify that the user's personal information is not being stored anywhere.

SR2. The program shall not use any files other than the ones the user has provided and the ones that are included with the program and Python.

- Rationale: To ensure the user's information and privacy is maintained, no external files should be accessed.
- Fit Criterion: When reviewing the code, there should not be any code which accesses files outside of the user-provided inputs and files included in Python and the program.

SR3. All revisions to the program shall be visible on GitHub.

- Rationale: To allow audits for all versions of the program.
- Fit Criterion: All users should be able to see and use every version of the program through GitHub.

4.7 Cultural Requirements

N/A

4.8 Legal Requirements

LR1. The program shall not use any libraries or packages that require licenses which the development team does not have.

- Rationale: Using libraries or packages that require licenses without owning the actual license is illegal.
- Fit Criterion: During code review, a list of all used packages and libraries shall be made and none of them will require licenses.

4.9 Health and Safety Requirements

N/A

5 Functional and Non-Functional Requirements Traceability

	LF1	LF2	UH1	UH2	UH3	PR1	~~PR2~~	PR3	OE1	OE2	OE3
Req. #: 1											
Req. #: 2											
~~Req. #: 3~~							~~x~~				
Req. #: 4											
Req. #: 5											
Req. #: 6							~~x~~				
Req. #: 7											
Req. #: 8											
Req. #: 9											
Req. #: 10											

Table 5: **Functional and Non-Functional Requirements Traceability Part 1**

	MS1	MS2	MS3	SR1	SR2	SR3	LR1
Req. #: 1							
Req. #: 2							
~~Req. #: 3~~							
Req. #: 4							
Req. #: 5							
Req. #: 6							
Req. #: 7							
Req. #: 8							
Req. #: 9							
Req. #: 10							

Table 6: **Functional and Non-Functional Requirements Traceability Part 2**

6 Project Issues

6.1 Open Issues

Since technology in these days evolves really fast, the hardware and software in the future could be extraordinarily different from what we have now. It is hard to forecast whether the product will work fine in the future or not and how should the product react to the coming changes. In addition, the product it is also not known if the product will work on new created operating systems that will be employed by majority of popularity in the future. Also, because the users' locals device will be hosting the product, the issues of the local machines would affect the product.

6.2 Off-the-Shelf Solutions

The original toolbox GERT is not open source, however we can use the source code to reverse engineer the toolbox. In this way we can reuse the same components of the original toolbox such as the modularity and general features, but designed with free open-source libraries in Python to define classes and functions in GERT.

6.3 New Problems

Since we are reverse engineering GERT that is using the ArcPro Geographic Information System and a proprietary sofware called ArcPro, we may have the problem of keeping the same level of performance as the original toolbox. This can pose a significant problem as currently GERT can scale up to large GPS data and that serves a great purpose of how and why it was designed. While we may be able to output the same meaningful information as GERT with our newly designed system, it will not be of much use if the system is not nearly as optimized as GERT. Other problems include potential user problems such as fatigue from prolong use of the product. This may result in muscle pain, dizziness, general discomfort, nausea, etc. Fatigue should not be a significant problem for this project as we suggest users to take periodic breaks when using the product for a large period of time.

6.4 Tasks

To ensure all of our tasks are met, we will use the V-Model of Software Development for our project planning and development. The design phase of the project has 5 deliverables: Problem Statement, Development Plan, Requirements Specification, Verification and Validation Plan, and Design Specifications.

- The Problem Statement and Development plan are scheduled to be completed by September 26 and can be completed in 1 week.
- The Requirements Specification is scheduled to be completed by October 5 and can be completed in 1 week.
- There is also a Hazard Analysis scheduled to be completed by October 19 and can be completed in 2 weeks.

- The Verification and Validation Plan is scheduled to be completed by November 2 and can be completed in 2 weeks.
- In between the Verification and Validation Plan and Design Specifications there is a Proof of Concept Demonstration scheduled to be presented between November 14-25 and that can be completed in 2 weeks.
- Next is the Design Specification that is scheduled to be completed by January 18 and that can be completed in 1 month.

Once the design phase is completed, the coding phase may start.

- Revision 0 Demonstration is scheduled to be completed by February 6-17 and can be completed in 3 weeks.
- After Revision 0, the Verification and Validation Report is scheduled to be completed by March 8 and that can be completed in 1 month.

Once we have gone through all the phases of the V-Model we will revise previous documentation and make any necessary changes.

- Then the Final Demonstration is scheduled to be completed by March 20-31 and that can be completed in 3 weeks.
- Lastly, the Final Documentation is scheduled to be completed by April 5 and that can be completed in 3 weeks as a lot of it will be based on the work we did using the V-Model.

The schedule of deliverables and tasks are all covered in the dynamic Gantt Chart of the development team which can be found from the following link: Gantt Chart

6.5 Migration to the New Product

When migrating the existing toolbox to our new product, the development team has to consider how the data might be modified or translated for the new system

6.6 Risks

Since the project will be processing such a large data-set, the development team should be aware of how much of a computer's resources the program will use, so no permanent damage is caused on the user's system. To reduce this risk we will explore several solutions such as progressive loading, using fast loading libraries, potentially change the data format, storing files in the #Parquet format, and many more.

6.7 Costs

As long as the development tools and documentation tools remain free, there will be no cost for developing the product.

6.8 User Documentation and Training

The development team will use quarto notebooks to document all code and give in-depth descriptions to all modules, classes and functions in the source code of the product written with Python. In addition to documentation on the source code, the development team will create a Readme that will describe how to use the finished product and what features are available.

6.9 Waiting Room

There are some requirements that will not be apart of the initial release of the new product. Examples of these requirements include:

- The program can only be used in the command line. In the future, a GUI can be developed for better accessibility.
- The program will have videos/manuals on how to use it that would make the tool more accessible.

6.10 Ideas for Solutions

The development team could define the data for the new product using GeoPandas which is an open source project to make working with geospatial data in Python easier.

It may be possible that GeoPandas will not be sufficient enough for this project. In that case, in the unlikelihood we cannot use GeoPandas then we can use the Pandas library and extend the datatypes of Pandas to match the original toolbox.

7 Likely and Unlikely Changes

This table outlines which functional and nonfunctional requirements are likely and unlikely to be changed in the future. Explanations for the likely to change requirements will be listed under the table.

	Likely To Change	Unlikely To Change
Req. #: 1	×	
Req. #: 2		×
Req. #: 3		×
Req. #: 4		×
Req. #: 5		×
Req. #: 6		×
Req. #: 7		×
Req. #: 8		×
Req. #: 9		×
Req. #: 10		×
LF1		×
LF2		×
UH1		×
UH2		×
UH3		×
PR1		×
PR2		×
PR3		×
OE1	×	
OE2		×
OE3		×
MS1		×
MS2		×
MS3		×
SR1		×
SR2		×
SR3		×
LR1		×

Table 7: **Likely and Unlikely Changes**

LC1. Change to Req. #: 1: The user may be allowed to use more than only CSV input files, and output more than only CSV and SHP files.

LC2. Change to OE1: The product will only run on Windows and Linux. Support for MacOS may be added in the future.

Appendix — Reflection

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

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?

Knowledge and Skills

- To gain a deep understanding of the problem, the team will need to acquire domain specific knowledge, such as knowledge of GIS and GPS data.
- To re-implement the features in GERT that use ArcGIS Pro packages, the team will need to learn about ArcGIS Pro collectively.
- In order to complete the documentation of the project, each team member will need to learn and further improve their LaTeX skills.
- In order to successfully manage the project on GitHub, each team member will need to acquire and further develop their skills in Git.
- For the purpose of CI/CD, the team members will need to learn to use GitHub Actions platform which is built into GitHub.
- For the development of Python project, the team members will need to learn about all necessary libraries utilized in the project, including GeoPandas, numpy, etc.
- For developing any necessary R project, the team members will need to acquire knowledge of R and learn to use RStudio as the integrated development environment for R.

Approaches

- For domain specific knowledge, the team members can do some secondary research and review the pertinent documents to acquire the necessary knowledge on their own. They can also choose to ask the supervisor of the project, Dr. Antonio Paez, about certain domain specific knowledge.

  - Jasper - This is a good approach because there is in-depth documentation for GERT, and other domain specific knowledge, that make concepts easy to understand for new users. Dr.Paez also has extensive experience using GERT and knowledge of geography and related concepts, so being able to ask him questions will be extremely useful for the group.

- To learn about ArcGIS Pro, the team members can go to the GIS labs located in the Burke Science Building try the software. They can also install it on their personal computers and learn about its features. It is also possible for the team members to ask the supervisor of the project if they have any questions.

  - Mike - This is a good approach because it is important to have hands on experience with GERT and ArcGIS Pro in order to fully understand their features. As PyERT is essentially replicating GERT without ArcGIS, having access to and using GERT will provide a valuable resource to compare the progress of this project to.

- To further develop the LaTeX skills, there are a lot of online resources available and the team members can use them to help refine the documents. It is also possible to learn from sample documents provided by the instructor of the course.

  - Linqi - This is a good approach since our document is very professional and requires high-level background knowledge to understand; besides, high-level scientific expressions are required to explain our project perfectly. Latex is such a tool that features a reliable program for typesetting, footnotes, bibliographic, images, captions, tables, and cross-references. Our team can generate more accurate documents by investigating online resources and current latex files.

- For Git and GitHub, the team members can learn from the tutorial of the course. They can also check any online resources that would be helpful, such as the reference manual for Git and the Pro Git book.

  - Hongzhao - This is a good approach since the tutorial of the course can teach about how to use Git with self-explained examples presented by the instructors and the examples can be followed easily, and the online resources about Git could help on some use cases of Git that the tutorial has not been covered but needed by the project

- To learn to use GitHub Actions, the team members review the tutorial on it and try to follow that to gain a better understanding. It is also use the GitHub Actions documentation for reference.

  - Mike - This is a good approach as having a continuous integration and continuous delivery pipeline allows the development team to send out changes to code quickly and efficient. It can improve the team workflow and also makes it easier to recover from any issues that arise, as it will be easier to locate the deployment that introduced a bug.

- For the Python libraries, it would be helpful for the team members to read through their documentation. There are also a lot of online tutorials and examples on how to utilize them to accomplish the goal of the project.
  - Zabrain - This is a good approach as there is a lot of in-depth documentation on each Python library that explains how everything functions and when to use them. Furthermore, if the documentation is difficult to understand then there are many external online resources such as video guides and example projects making use of different Python libraries that can be useful for better understanding
- For R and R and RStudio, the team members can turn to their documentation for help. There are also online videos and webinars that would help the team to gain a better understanding on how to use RStudio. It is also possible to ask the supervisor of the project if there is any specific problem regarding this project.
  - Mengtong - This is a good approach since the online videos and webinars are quite helpful for understanding how things work, especially when they demonstrate the process of using RStudio to accomplish their tasks. It is also easier to learn new things when there is a video to follow.

8 Changes to Volere Template

The team decided to add some sections to the template which we thought were fitting.

- System Description (Section 2.1-2.5): Our group thought that adding a system description would be useful for our project, since the project concept is not straightforward.
- Traceability Matrix (Section 5): The traceability matrix shows the dependencies between functional and non-functional requirements.
- Likely and Unlikely Changes (Section 7): The Likely and Unlikely changes will be useful in the future, and can be referenced by stakeholders if changes are made to the final product.
- Appendix - Reflection: This section will be useful for the team for planning, and assigning responsibilities to different areas of research that will be needed for this project.

References

Ron Dalumpines and Darren M. Scott. Gis-based episode reconstruction toolkit (gert): A transferable, modular, and scalable framework for automated extraction of activity episodes from gps data. Travel Behaviour and Society, 11:121–130, 2018. ISSN 2214-367X. doi: https://doi.org/10.1016/j.tbs.2017.04.001. URL https://www.sciencedirect.com/science/article/pii/S2214367X17300406.

James Robertson and Suzanne Robertson. Volere Requirements Specification Template. Atlantic Systems Guild Limited, 16 edition, 2012.