Table 1: Revision History

Date	**Developer(s)**	**Change**
Sept 24	Jasper Leung	Team Meeting Plan, Team Communication Plan
Sept 24	Hongzhao Tan	Technology
Sept 25	Mengtong Shi	Coding Standard
Sept 25	Zabrain Ali	Workflow Plan
Sept 25	Mike Li	POC Demonstration Plan
Sept 25	Linqi Jiang	Project Scheduling

Development Plan
Software Engineering

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

September 26, 2022

This document will go over the Development Plan of the project Match GPS Traces to a Transportation Network.

1 Team Meeting Plan

1.1 Overview

Track a Trace will hold scrum meetings weekly. All members will be required to attend these scrum meetings. Along with these scrum meetings, the team will organize at least one informal work session per week with a select number of members, where we will sync our progress on the tasks we are working on, and work as a group to write documentation or develop code.

1.1.1 Scrum Meeting Structure

Due to inconsistent schedules, the time slots of these scrum meetings will vary. However, they will all be 15-30 minutes long, and usually held online on Microsoft Teams. The Scrum Master will start the meeting. Team members will speak in order, stating what tasks they are planning to work on, currently working on, and have finished since the last scrum meeting. The team will give feedback to each other based on what they have said during their turn.

1.1.2 Work Session Structure

The time slots of these work sessions will be vary based on the work that needs to be done. The general goal of these meetings is to have each participant of

the meeting state what they want to work on before starting the meeting, and after that, the team will come to a conclusion on what will be worked on during that work session. Once again, what happens during that work session will vary based on the work. After each work session, the team will collectively recap what was done during the work session. All work done will be documented by GitHub commits and code documentation.

2 Team Communication Plan

Track a Trace plans to hold in person meetings, as mentioned above, but also plans to use online communication methods.

2.1 Microsoft Teams

Microsoft Teams will serve as the team's main form of online communication. Microsoft Teams provides file sharing, message logging, and real-time communication, which will be useful assets for the team. Team members will use the Teams messaging feature to update each other on what they will be working on next, and to organize team meetings. Online meetings will be held on Teams as well.

2.2 Cell Phones

Cell phones will be used by the team to communicate with each other before meeting in person. Cell phones are more convenient to use than Teams on-the-go, and using having team members call each other will ensure that we are able to figure out how to meet while we are even if we are in transit.

2.3 GitHub

The team plans to utilize GitHub branches and pull requests, and through those, the team can give feedback on the changes that a member has made. This will allow for more detailed review of the work that each member has done, as well as making it easier to pinpoint specific issues compared to using Microsoft Teams. Also, GitHub will be used to store all of the team's documentation.

3 Team Member Roles

- Jasper Leung - Scrum Master, Developer
- Hongzhao Tan - Developer
- Mengtong Shi - Developer
- Zabrain Ali - Developer

- Mike Li - Developer
- Linqi Jiang - Developer

4 Workflow Plan

The process of the workflow will revolve around git. The first step of the workflow is to pull any changes from the master branch. The next step is to create a new branch (i.e. feature, bugfix etc.) from the master branch. As code changes are made, commits with detailed messages will be made. Git tags will also be used to mark stable releases or achievement of very important milestones. The next step is to add unit/integration tests to our code. After all the necessary code changes and tests are committed and pushed to a branch, the next step is to create a pull request to merge to the master branch. We will use CI/CD for the software development of this project to run automated tests using GitHub Actions once a pull request has been made. Furthermore, all developers will be added as a reviewer for each pull request and all developers must approve the pull request before merging to the master branch. Once a branch has been merged to master by the approval of all reviewers, the branch will be deleted and the workflow will repeat from the first step.

In the event there are any merge conflicts, here is the workflow to resolve it. First, identify where the merge conflicts occur and what developers were working on the same files. Next, developers will review the conflict together and make any necessary changes whether that is combining both code changes, removing one developers change, or rewriting the file entirely. After the file is edited, we will do a git add to stage the new merged file and commit it. Git will then create a new merge commit to finalize the merge.

5 Proof of Concept Demonstration Plan

The main risk for the success of this project is that ArcGIS might have a functionality that is too complex, or not replicable with free Python tools. Since we are replacing ArcGIS functionality with free tools to make GERT free of licensing ArcGIS, any ArcGIS function that is irreplaceable would require licensing. Another risk would be if it would take too long to replace all the ArcGIS functions with free libraries and tools. As long as there exists a single function using ArcGIS in GERT, whether it's from the function being too complex to replace, or if it's because we did not have enough time to replace it, a license needs to be bought, and the success of this project would be compromised. If during the Proof of Concept Demonstration, we demonstrate that all ArcGIS functionality can be replicated with free tools and libraries, and find a reasonable number of functions that require ArcGIS in GERT, we can convince ourselves that we can overcome the risks.

6 Technology

- Python and R will serve as the coding languages
- Pylint will serve as linter tool
- Pytest will serve as unit testing framework
- GitHub Actions will be used for Continuous Integration (CI) purposes, the main branch will be protected with branch protection rules including:
  - Require an approved pull request before merging
  - Require status checks to pass before merging
  - Require source branches to be up to date before merging
- Libraries that will be used include GeoPandas, numpy and Python xml library
- GitHub will be used for version control
- Visual Studio Code, RStudio and other potentially helpful IDEs will be used for implementation
- Latex and Overleaf will be used for editing and generating documentations

7 Coding Standard

- The Python code will follow the PEP 8 style guide.
- Black will be used to format the Python code.
- pycodestyle (formerly pep8) will be used to check the Python code against the PEP 8 style guide.
- Pylint will be utilized as the linter and help enforce the coding standard.
- The R code will follow Google's R Style Guide which can be found here.

8 Project Scheduling

- See Gantt chart at the following URL: Gantt chart