Reflection Report on yoGERT

Smita Singh, Abeer Alyasiri, Niyatha Rangarajan,
Moksha Srinivasan, Nicholas Lobo, Longwei Ye

Reflection is an important component of getting the full benefits from a learning experience. Besides the intrinsic benefits of reflection, this document will be used to help the TAs grade how well your team responded to feedback. In addition, several CEAB (Canadian Engineering Accreditation Board) Learning Outcomes (LOs) will be assessed based on your reflections.

1 Changes in Response to Feedback

Summarize the changes made over the course of the project in response to feedback from TAs, the instructor, teammates, other teams, the project supervisor (if present), and from user testers.

For those teams with an external supervisor, please highlight how the feedback from the supervisor shaped your project. In particular, you should highlight the supervisor's response to your Rev 0 demonstration to them.

1.1 SRS and Hazard Analysis

1.1.1 SRS

Our SRS document was modified multiple times based on the feedback from our peers, TA, and supervisor. The feedback we gained from our peers helped us to improve the communication style to clarify the misconceptions made early on in the project. Moreover, the feedback we received from the TA allowed us to make the necessary changes to improve the quality of the document from a professional point of view and add more details to the information documented. On the other hand, the feedback we obtained from out supervisor, Dr. Paez, was crucial to verify we have the correct requirement document and continuously validate that the requirements were understood correctly. In conclusion, the value added from these multiple forms of feedback shaped our document to be more consistent to the best engineering practices and in align with Dr. Paez vision for the project.

1.2 Design and Design Documentation

1.2.1 Software Architecture(Module Guide)

One of the major changes we made is to remove the main module(i.e. the command line interface that interact with users) based on the feedback from REV0 presentation. Besides, we correct some naming errors based on the feedback from TA.

1.3 VnV Plan and Report

1.3.1 VnV Plan

As part of our ongoing efforts to improve our VnV plan, we have recently made some important changes. Firstly, we have updated our traceability matrix to ensure that it accurately reflects our design. In particular, we have removed some non-functional requirements that are no longer relevant to our project, and have made sure that all remaining NFRs are properly aligned with the functional requirements that we are testing. Secondly, we have also updated and clarified our test input data and outputs. This includes providing more detailed information about the expected outcomes of each test.

1.3.2 VnV Report

To improve the quality and accuracy of our VnV report, we updated the traceability between our VnV plan and VnV report. This update ensures that every requirement in the VnV plan is properly mapped to its corresponding verification or validation activity in the VnV report, which helps to identify any potential gaps or discrepancies in our testing process.

2 Design Iteration (LO11)

The yoGERT team employed an iterative design process to structure the project's milestones and progression, resulting in significant evolution of the project's final state over time. Initially, the project lacked the necessary level of geospatial understanding and technical tools to succeed, which hindered the team's progress through the project's milestones. However, by following the iterative design process guidelines, the team was able to fill in any missing information, refine the project specification, and receive feedback to improve completed work before moving onto the next stage. For example, prior to implementation, the team held a formal SRS document review with the project supervisor to verify the completeness of the requirements and clarify the use cases for the final product. As a result, the team was able to make significant changes to the specification details to address certain business goals that the project needed to meet. Ultimately, the final state of the project successfully met all functional requirements and delivered an open source library that the target user base can utilize.

3 Design Decisions (LO12)

The main constraint that affected yoGERT's design was the open-source nature of the library as specified by the supervisor, Dr. Paez. First, there was a high focus on code readability to allow future developers to easily extend upon the existing work. This included robust documentation through every step of the project, including examples outlining how to use dependent libraries and providing wiki outlines of how certain calculations are made (ActivityLocations, Route Generation). Furthermore, along with doxygen-style comments, code was written to enhance readability. For example, if code reviewers had trouble understanding lengthy list comprehensions, they were often broken into more steps to make the code easy to read.

Additionally, the constraint of dependence on only open-source libraries and sources of data significantly changed the design of yoGERT. One example of this is the use of OpenStreetMaps in yoGERT instead of the Google Maps API to poll information about roads and services. This affected performance and as such function parameters needed to be changed to account for the user's tolerance for slow requests.

Next, based on early chats with Dr. Paez, yoGERT needed the capability to produce "sub-outputs", meaning each sub-step outputs its own file. This assumption differed significantly from the original toolbox, which provided low visibility of the intermediate steps of each main function to the user, only mapping the output on ARCGIS. This assumption led to a focus on the modularity of the code, making intermediate outputs available to view and making intermediate functions callable by the user. This ended up being a great design choice, while reviewing requirements with our supervisor, it was made clear that access to these intermediate data states can be highly useful in some use cases. Additionally, this focus on modular design subsequently helped with the open-source nature of our project as well. This makes it easier for other developers to contribute to the project and can help ensure the long-term maintainability of the codebase.

4 Economic Considerations (LO23)

Our product is catered for users who use geospatial tools for analysis of different GPS data. For marketing the product within the geospatial domain, we aim to increase the marketing of the product through reference. Professor Paez, who is currently part of the Faculty of McMaster has closely worked with us to create such an open source software used by his own team and his colleagues who face the same issue with licence costs.
For attracting people outside the industry, yoGERT has a number of extendable and future use cases. It can be used as part of Sobi bike routing, landscaping, etc.

By advertising potential uses for public purposes, we can increase engagement. Since the final product is approved by him, he could be a point of reference for others in the industry. Since, it is an open source software available from pip, anyone can use. The chances of monetizing is not a fore thought for such an open source edition.

5 Reflection on Project Management (LO24)

This question focuses on processes and tools used for project management.

5.1 How Does Your Project Management Compare to Your Development Plan

The development plan was followed closely with some minor deviations from the original plan.

The team followed the development plan by meeting at least once a week, however the specific time of the meetings varied depending on team member availability for specific weeks. All of our online meetings occurred on Microsoft Teams, and our in-person meetings would be held at Hatch or ITB project room. Lastly meeting agendas were established before meetings and meeting notes were taken during each meeting as well.

In terms of the communication plan, the team did use Microsoft Teams for a majority of the communication. Discord was rarely used(even under urgent situations).

Team member roles were not strictly followed like outlined in the development plan. Each team member would step up and take on different roles when necessary. Also, most team members became good at the roles established in the development plan so there was not a designated expert. However, each team member took more on an expert role on the module they were tasked to implement.

Workflow plan was followed to an extent. Whenever possible, team members would create their own feature branch and create a pull request to be merged into main. However many last minute changes were done directly to main without any pull requests or code reviews. The GitHub issue tracker was used very frequently to log issues and changes and assigned to the appropriate team member.

The technology we used was the technology we planned on using. The whole project was implemented in Python. The testing was done using Pytest as planned. Continuous Integration was unfortunately not implemented due to errors and insufficient time allocated to its implementation.

5.2 What Went Well?

Throughout our project management journey, we encountered several successes in terms of processes and technology. Firstly, we were able to utilize various

Python libraries such as OSMnx and Pandas to create our project, which allowed us to complete our project efficiently and effectively. Additionally, we were able to maintain our documentation throughout the project by constantly updating it based on feedback from our TA and the issues posted on Github, ensuring that all members of the team were up-to-date with the project's progress. Finally, we successfully met once a week, allowing us to discuss our progress and divide up work when necessary, which helped us stay on track and meet our project goals on time.

5.3 What Went Wrong?

One of the problem we resolved during our reversions is the implementation of the Main Module. We initially assume that we need a main module worked as the interface between the user and the program via command lines, but in the end, we decided delete this module and pack the remaining modules as a Python library so that the user could call yoGERT just like calling other functions.

5.4 What Would you Do Differently Next Time?

Next time, we may expand more features such as analyzing multiple routes. Besides, we may improve the algorithm we are using for generating shortest routes so that the time spent for large amount of data points can be reduced, which gives a better performance to the user.