ECSE 321 - Intro to Software Engineering
Deliverable 6 Report

Harley Wiltzer
Camilo Garcia La Rotta
Jake Shnaidman
Robert Attard
Matthew Lesko

April 11, 2017

1. Implementation

All the requirements formally identified by an ID are references to the Deliverable 1.

- **Functional:** Total implementation of the R. 1.1.1.1 to 1.1.1.4 with the caveat of serialization to text file instead of an xml format for the Web platform. This was a design choice made during the 2nd deliverable to use the same persistence methodology seen in Event Registration. R. 1.1.1.5 was not accomplished, a database was voted as too much of a frame for the access and retrieval of data. A much lightweight approach of storage in text files was preferred. R 1.1.1.6 is partially accomplished. Mobile and Desktop are fully capable of using each others xml outputs. Web remained incompatible due to its plain text output serialization.

- **Non-Functional:** From section R. 1.2.1 only R 1.2.1.3 will ship with the final report. All other requirements in this section were dropped from development as they related to an extra scheduling algorithm design that the team lacked time to develop. From section R. 1.2.2, all platforms provide a secure authentication procedure linked to a profile. The only requirement not achieved was 1.2.2.6, as NTRU cryptosystems required more research, and not enough time was available. From section 1.2.3 all requirements were achieved, providing a reliable cross platform version of the product was one of the main goals of the team. From section 1.2.4, all applications work under the same colour pallet, but as a caveat Mobile is the only platform without light/dark theme mode. It was chosen to rely on androids native accessibility options to change color gradients on the cellphone.

- **Desktop:** From section R 1.1.2 all but 1.1.2.9 was accomplished. As demanded by the client, the Desktop application is fully able to perform the tasks of an Instructor, Department and Student Profile which can be found in more depth in the Mobile and Web section of this report.

- **Web:** The web service is capable of dual profile actions. An administrator can log in and create courses instances and instructor profiles as well as linking them. An instructor can log in and publish job postings and manage the received applications as well as evaluating TAs.

- **Mobile:** The shipped application provides the capabilities of appying for a job, accepting/rejecting offers, creating and modifying the user's student profile, and viewing instructor evaluations.

2. Usability of Application

- **Desktop:** The desktop program shall be run by double clicking on the .jar file of the program. All saved data (persistence data) is stored in the $HOME/.tamas/output directory, which is generated by executing the init-desktop target of the build script. In order to run the desktop application, this persistence folder *must* be created. Thus must only be done once, when using the app for the first time, and must be done subsequently any time the persistence folder has been deleted.

- **Web:** The Web application is accessed through the web and doesn't host any permanent or temporary files client-side. The user is required to access the server that hosts the website.

- **Mobile:** The mobile program shall be released as an apk file that may be installed on an Android phone. Once the apk file is installed, the user may use the application on the phone. Persistence from the desktop application may be imported into the mobile app by placing it in the mobile device's `/storage/emulated/0/Android/obb/ca.mcgill.ecse321.group10.TAMAS` directory.

3. Testing of Applications

The unit, integration and system testing strategies devised for deliverable 3 were implemented for the desktop application to ensure that maximal performance and coverage as well as to ensure proper functionality of the system would be achieved. The testing strategy needed to be adjusted somewhat for the web app due to the differnt way that each functionality was implemented in php. As the android app depended on the java code made for desktop, it was suficient to test units and integration on dekstop and not on android.

- **Unit Testing:** A plethora of unit tests were written for the desktop application, and currently offer 100% coverage of the controller classes. In similar fashion to the desktop unit tests, all controllers were tested separately during unit tests for the web application. Coverage on the unit tests is similar to that of the desktop application.

- **Integration Testing:** Because the desktop app has more functionality than the web app, the integration testing of it involved a more comprehensive list of test cases than the desktop app did. As the web app only handles things from the perspective of a teachers point of view, which means to look at things in terms of jobs, it was sufficient to test that all functionality and error-detection involved in job creation was covered.

- **System Testing:**At the system level, all three applications were extensively tested after each new iteration of the source code. Due to the sheer size of the applications, it is difficult to measure the exact system test coverage, though it is believed that the state coverage on all platforms was close to 100%.

4. Release Pipeline

The release pipeline shall follow the plan as in deliverable 4's report. Every release shall follow semantic versioning: MAJOR.MINOR.PATCH. With the first release being V1.0.0. During deployment, the Travis script will invoke the scripts: ant export-desktop, ant export-web, and ant export-mobile. These shall export the necessary jar, apk, and web files. All source code will be available as well.

- **Desktop:** The Executable Desktop JAR archive will be released. The APP/Desktop directory will be available in the source code.

- **Web:** The Web Application zip archive will be released. The APP/Web directory will be available in the source code.

- **Mobile:** The apk file of the android program will be released. The APP/Mobile directory will be available in the source code.

It is interesting to note that a major part of the release pipeline time budget was allocated to the implementation of build script. Apart from the base functionalities it is also capable of packaging the web app and loading it in a browser, creating an executable and running the desktop application, and installing the mobile app automatically to a connected android device. This strongly increased the development team's productivity.

5. Responsibilities for Each Phase

- **Harley Wiltzer: (∼80h)** Sole developer and principal tester of the Desktop app. Designed the logic and UI for each use case. Designed and implemented the multi-platform build script to completion.

- **Camilo Garcia: (∼70h)** Sole web developer. Co-designed the umple model. Designed Sequence Diagram implementations on web. Designed and implemented all the unit tests cases form web app.

- **Jacob Shnaidman:(∼52.5h)** Wrote 95% of the mobile application. Wrote integration tests. Contributed to design of system architecture and model.

- **Robert Attard: (∼48h)** Main tester for the web app. Assisted in system testing on the desktop. Kept all tests for the web app up-to-date. Assisted with the layout and style of the UI.

- **Matthew Lesko: (∼40h)** Tester for Desktop App. Involved in SW Design. Implemented Travis CI as a CI tool. Involved in report documentation.