ECSE 321 - Intro to Software Engineering
Backlog Week 1

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

February 17, 2017

1 Week 1

1.1 Main Tasks

- Harley Wiltzer
  - Co-formulated the prototype Requirements Document.
  - Drafted the first Requirements Document.
  - Designed Sequence Diagrams for Instructor Actions, Student Actions, Admin Actions.
  - Made small changes to Class Diagram.
- Camilo Garcia La Rotta
  - Co-formulated the prototype Requirements Document
  - Designed the prototype Class Diagram
  - Structured the GitHub file system
- Jacob Shnaidman
  - Designed Use Case Diagrams
  - Wrote Use Case Descriptions
  - Helped with the design of Sequence Diagrams
  - Wrote the Work Plan
- Robert Attard
  - Assisted in preliminary design of the use case diagrams
  - Designed and improved on the state diagram
  - Made wording and grammatical changes to Requirements Document
- Matthew Lesko
  - Designed the prototype Use Case Diagram
  - Co-designed Sequence Diagrams

1.2 Key Decisions

1. **Requirement Document**
2. **Use Cases**
   - Included four actors in the diagram, three of which are users and one of which is a service. The three users are: Student, Instructor and Department (has administrator privileges). The service is a Security Authentication.
   - Each user in the diagram acts upon multiple use cases respective to their functionalities mentioned in the Requirements and Specifications documents. Each use case describes functionality that its respective actor has.

3. **Class Diagram**

    - Trying to mimic the architecture of the event registration app, an early choice was to have a JobManager oversee the job transactions between Instructors and Students
    - Each user, whether it has administrator, instructor or student access rights, is obliged to have profile instance to facilitate identity access management
    - Later, it was also decided to have a ProfileManager to handle the actors' profiles and their authentication.
    - To enable an administrator to recreate actions available to instructors or student, we give the class administrator the power to create instances of the aforementioned users

4. **Sequence Diagram**

    - For convenience and concision, it was decided to design sequence diagrams for each main user action.
    - The sequence diagrams may contain actions from multiple user classes to emphasize the sequence. For example, in the Instructor sequence diagram, the action of an administrator verifying Instructor modifications to the TA/Grader hours is seen.
    - The current sequence diagrams include Student actions, Instructor actions, Administrator actions, and the authentication process.

5. **State Chart**

    - The state chart tracks the possible states of the job posting between being created and the position being filled.
    - The chart also indicates the triggers required to transition between job posting states.

1.3 Work Plan

In the following week, we will create more detailed sequence diagrams going into the details of the implementations on individual platforms. We will then develop prototype source code for the Publish Job Posting and Apply for Job use cases for the desktop and mobile applications respectively.

1.4 Work Hours

- Harley Wiltzer
    - Requirements document: 1 hours
    - Sequence diagrams: 3 hours
    - Class diagram: 30 minutes
    - Compilation of report: 4 hours
- Camilo Garcia La Rotta
    - Requirements document: 3 hours

  - Class diagram: 2 hours
  - State diagram: 2 hour

- Jacob Shnaidman
  - Use Case Diagrams: 4 hours
  - Use Case Description 1.5 hours
  - Helped with Sequence Diagrams 0.5 hours
  - Work Plan 0.5 hours

- Robert Attard
  - Requirements document: 0.5 hours
  - Use Case diagram: 3.5 hours
  - State diagram: 3 hours

- Matthew Lesko
  - Use case diagram: 4 hours
  - Co-design sequence diagrams: 4 hours

2 Week 2

2.1 Main Tasks

- Harley Wiltzer
  - Aided in the design of the revamped domain model
  - Designed and developed desktop application
  - Aided in the design of the mobile application
  - Created the Sequence diagram for the Desktop - Apply to Job Use Case
  - Helped create the Sequence Diagram for the Desktop - Publish Job Posting Use Case
  - Compiled the final report
- Camilo Garcia La Rotta
  - Designed and documented Sequence Diagrams for the Web and Desktop implementation of "Publish Job Posting"
  - Derived the PHP code implementation of "Publish Job Posting" and "Apply to Job Posting" based on the aforementioned design-level Class Diagram
- Jacob Shnaidman
  - gg
- Robert Attard
  - gg
- Matthew Lesko
  - Co-designed preliminary Architecture Block Diagram
  - Wrote draft for Architecture's Description and Rationale
  - Co-designed Detailed Design Class Diagram

2.2 Key Decisions

1. **Use Cases**
   - In order to correctly implement the Entity-Control-Boundary pattern and symbolism in the Use Cases, the naming of the cases from the first deliverable were reviewed and adjusted. Moreover, the relationships between each action was refined so as to reflect the following rules:
     - Actor only interacts with <<Boundary>>
     - <<Entity>> only interacts with <<Control>>
     - <<Control>> manages the interaction flow between <<Boundary>> and <<Entity>>
   - The overall structure of the Use Case for deliverable #2 focused solely on the Profile subclasses Admin and Instructor. To highlight this structure the Use Cases made a deliberate use of Generalization.

- The "Publish Job Posting" Sequence diagram had to be clear about the alternative actions based on the approval or disapproval of the Posting instance. As such, an ALT pane was implemented only for the Web Use Case, which didn't necessarily imply the posting would be automatically approved. As opposed to the Desktop application, only used by Admin which ensures the posting is automatically approved at the moment of submitting a publish form.
- It was decided that the Desktop app can handle the creation of all major entities. Starting with no data, the Desktop app has the power to create all necessary actors for all features of the program to be possible.
- For the purpose of Deliverable 2, the mobile app does not have the ability to create student profiles. Students register for jobs by specifying their username, as authentication has not been implemented yet.
- Since the mobile app does not support the creation of Students or Jobs, it was decided that it should programmatically create Students and Jobs if no current persistence XML file is found. This was decided solely for the convenience of the graders of this deliverable. If this is not desired, the grader may input his/her own XML to the appropriate directory.

2. **Domain Model**

   - ProfileManager, CourseManager, and ApplicationManager were added to the domain model to assist in the creation and persistence of Profiles, Courses, Jobs, and Applications.

2.3 Work Plan

gg

2.4 Work Hours

- Harley Wiltzer
  - Domain Model Modification: 2 hours
  - Java Desktop App: 9 hours
  - Android App: 2 hours
  - Sequence Diagrams: 2 hours
- Camilo Garcia La Rotta
  - Use Case Diagrams: 3 hours
  - Sequence Diagrams: 2 hours
  - PHP: 9 hours
- Jacob Shnaidman
  - gg

- Robert Attard
  - gg
- Matthew Lesko
  - Architecture: 3 hours
  - Detailed Design: