Moises Fernandes

Applied Programming Languages

Assignment 2

May 28th, 2022

**Integration and Configuration**

**Description of each step (Integration and Configuration)   
Requirement Specifications:** This is the phase where we come up with initial requirements for the system

**Component Analysis:** Find and study existing software’s that have similar if not the same specification as the desired system you want to create

**Requirement Modification:** Revise the currently established requirements to match what you found after the component analysis phase

**System Design with Reuse:** Use the components, make modifications as you work

**Development and Integration:** The development phase is where we start to code the necessary components

**System Validation:** This is the phase where we test the components and reach a conclusion on whether it is ready, or we need to adjust.

**Integration and Configuration Detailed Outline (LeopardWeb System)**

**Requirement Specifications**

Feasibility study: Looking at the criteria and understanding what needs to be done, I conclude that this project is possible and can be satisfied with current technology. For this project the best approach will most likely be using Object Orientated methods such as classes, inheritance, polymorphism etc.

Requirements elicitation and analysis: Understanding that we are emulating a university course system we can look at other models that are used for out project to lessen the workload, doing so we can also make necessary changes.

Requirement Specifications:

* Data base of users: The system needs to operate for 100 students, 10 instructors and 1 admin
* Data base of courses: This is going to contain all the information needed for course related searches such as, CRN codes, course names, time, and the instructor per course
* Three different types of users:

1. Student – can register, see available courses and their own schedule
2. Instructor – can see available courses, and their own course roster
3. Admin – can see everything, can edit courses/ users/ schedules

* The system should include multiple semesters, print out of schedules, schedule preferences.
* The system needs to be thoroughly testing before being released
* The base class is the User, which shall accommodate these specifications:

1. Attributes: First Name, Last Name, User ID
2. Methods: Set function for each individual attribute, and a corresponding function to print all the information of a desired user.

* Derived Classes:
* All derived classes must contain any additional attributes appropriate to the specific class and needs to have “set/get” methods per specific attribute.
* **Student:** The students will be able to search for courses, add/drop courses, and print out their schedule
* **Instructor:** The instructors should be able to print their schedules and print their class list and search for courses
* **Admin:** The admins should be able to add courses to the system, remove courses from the system, add/remove users, add/remove students from a course, search, and print both rosters and courses.

Requirement Validation: These requirements make sense, and they also match with what the user wants the system to do.

**Component Analysis:**

* Are there existing systems that we can draw from to make our job easier?
* Since we are creating a university cite there definitely is an existing system that we can model our system after to lessen the workload and to ensure the best possible results

**Requirement Modifications:**

* After finding another site to model our system after, we then make the necessary edits to our requirements to match the requirements of the already existing system. This will help make our model more efficient.

**System Design with Reuse:**

**System Software and Design**

**Software validation**

* Components: Looking at what needs to be done a lot of the functions are similar in nature (add/drop, search, and print) so many of the components can be reused throughout the project.
* Program testing: Using made up course names and CRN codes to do an initial test of the system.
* Component testing: Frequent testing of all the stated components.
* System testing: Integrate databases, code, and user interface to test the entire system
* Acceptance testing: Using actual course names and CRN codes in our program

**Development and Integration:**

**Implementation and Unit testing**

Architectural Design: What components – Functions, classes and objects, data base   
Interface design: (1) Interfacing sub-components (2) the user interface (GUI will most likely be used) Data base design: how many tables are needed? One for all users; possibly a table per species, etc.? What should be in each table.

What is needed

Students: All derived attributes from user class

Instructor: All derived attributes from user class

Admin: All derived attributes from user class

Information will be stored: Class times, Instructors linked Courses,

Component design: Looking at what needs to be done a lot of the functions are similar in nature (add/drop, search, and print) so many of the components can be reused throughout the project.

Implement: Code, databases, user interface

**Integration and System Testing**

* Integrate all components and data bases and test them in a system.
* Testing all the different types of users, functions etc.

**System Validation:**

* Is this system an overall desired outcome? Does it cover all the required specifications and work efficiently?
* If so, then the system is complete and is ready for use
* If not, we continue to work on the system

**Online Resources to Integrate from:**

* Video by Prototype Project <https://youtu.be/-5iRWqUtBT8>
* GitHub course scheduler project <https://github.com/oist/course-scheduler>

**Stages of Iteration for our Project:**

For the first version of out project, I propose that we follow the video by Prototype Project to get the skeleton of our system completed. After that I think it would be best to use the GitHub course scheduler project since it has much more details and a database that we can use to model our own type of database. For example, in the GitHub repository it contains files that can be useful for our LeopardWeb project; one of the files contains a list of students ordered with attributes Student ID, Name and class taken. There are also files that help draw out and make a schedule which I believe will be helpful moving forward.