Moises Fernandes

Applied Programming Languages

Assignment 2

May 28th, 2022

**Incremental Model (Diagram)**

Initial version

Intermediate versions

Outline Description

Final version

**Description of each step (Incremental Model)**

**Outline Description:** This is a basic outline of the project and what needs to be accomplished.

**Specification:** This step of the process is where we decide what are the necessary specifications and how to implement them feasibly.

**Design & Development:** The development phase is where we start to code the necessary components.

**Testing:** The testing phase checks the performance of each existing functions and other components.

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

**Version 1, Version 2, Version N:** As we work; we can either continue to the next step or go back depending on what needs to be done after validation.

**Incremental Model Detailed Outline (LeopardWeb System)**

**Initial Version: LeopardWeb (Similar to Waterfall model)**

**Specification**

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.

**Design & Development:**

**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

**Testing**

**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.

**Validation**

* Is this system an overall desired outcome? Does it cover all the required specifications and work efficiently?
* If the answer is yes, congratulations we are done with the design process, and this is the final version of the system.
* If the answer is no, we need to re-evaluate (note that the validation step can be done in between other steps when following the Incremental design model)

**Intermediate Version:**

**Follows steps above (Specifications > Develop > Validation)**

* We can have an alert system to let users know when certain deadlines and important dates are coming by
* Have warning messages if courses overlap or if student hasn’t met prerequisite for a course they selected

**Final Version:**

**Follow Steps above (Specifications > Develop > Validation)**

* Create recommended class schedule for students
* Specific alerts sent to student such as course availability notifications
* CRN macro system