Quang Vu

Assignment 2

Applied Programming Concepts 3225-01

Comments in Red: Alexander Puttre

**Waterfall**

1. *Requirements & Definitions: (1 Week)*

Our end game is to develop software that will be used by students, instructors, and admins. Each will have their own specifications on what they can and can’t do. We need to check the constraints that our clients want to have (ex. How many students can access the software).

1. *System & Software designs: (2 Week)*

A user interface needs to be well known so that everyone that has access knows that this website is for specific purposes. How can we structure this in a way where only Wentworth students/faculty can use this (Ex. We wouldn’t want a random person being able to access leopard web). Maybe we have an auto generated feature where students can hit one button and have a fully made schedule based on which course CRN they choose.

During this phase, we also need to determine how the database is going to be implemented.

1. *Implementation & Unit Testing: (2 Week)*

From step 2, we can first start small to make sure our code works. Doing so, we can find bugs to patch before it goes into production. Each member on the team will have an equal opportunity to work on each part of our code/implementations. A 2nd verification should be used to check that what we did is what that person intended it to be.

What do you propose the 2nd verification should be? Also, in this phase, we’re testing each component in isolation.

1. *Integration & System Testing: (3 Week)*

Making sure that functions such as students, instructors & admins can all do what they intend to. Are there loopholes in the software that will allow a student to log in as an instructor/admin? Is there a way for students to brute force themselves in a course that is already full?

This is also where we integrate the components we created in step 3 into a single system.

1. *Operation & Maintenance: (1 Week)*

In our GUI, we can implement a way that allows people to report a bug/feature request. If the servers were to go down, we would ask the user to send a crash report (OPTIONAL) that will help us improve our systems.

The bug report functionality is a good idea. We should also check in with the clients a few weeks after deploying the system to see if they have any feedback.

**Incremental Development**

* *Phase 1: (2 Week)*

Here, the development team will decide on what we want the system to look like. Would the UI & GUI interact with each other. What type of users are we looking to have in our project?

I think the GUI is less important than the core functionality. Can the users perform the essential functions? We can work on the GUI in a later phase, once this groundwork had been set.

* *Phase 2: (2 Week)*

Add some functionality that our users can do. Students will be able to add/drop their course. Instructor will be able choose which classes that can teach. And Admins will be able to control everything. A UI is also expected to be in this phase.

* *Phase 3: (2 Week)*

A GUI is now implemented where we will be able to test our work. We will also be adding crash log features where we can use it if something happens, and the user wants us to investigate. Create security features.

Ideally, we would have means of testing our work before creating the GUI. The crash logs are a good idea. What kinds of security features?

* *Phase 4: (2 Week)*

Our school system is now complete. Before ramping up production, we will have extensive testing to root out bugs and perform patches. After that we will send out our system to a 3rd party to beta test our system for a week.

Who could act as the 3rd party in our case?

* *Phase 5: (2 Week)*

Completion of school system. It is fully functional with no known bugs that currently exist.

**Integrate & Configure**

1. *Requirements & Specifications: (1 Week)*

Refer to the waterfall methods. It will be completely the same.

1. *Component Analysis: (1 Week)*

To have our dependencies work together. We will be using Python as our GUI since it offers a graphic library so we can just implement it into our code much easier.

Specifically, what libraries will we use for the GUI and database?

1. *Requirement Modifications: (1 Week)*

PyQt5 is the newest version of Qt and might make our functionality easier for us to use since it has over 35 extension modules and is supported throughout all OS platforms.

Here we determine if the libraries we have chosen need to be modified in any way (which they most likely do not, in this case.)

1. *System Design with Reuse: (2 Week)*

Intending to use our database as SQLite and our GUI with PyQT5. This is very similar to step 2 of the waterfall methods.

This is where we design the system, so we should have decided upon SQLite and PyQt5 before this step.

1. *Development & Integration: (3 Week)*

Design our system while integrating SQLite and PyQt5. Finding bugs and patching them.

We write the software in this phase as well.

1. *System Validation*: *(1 Week)*

Go into beta testing with more than 100 people and ask for feedback. Adding more people to use our system will also determine how many people can have at one time. Will it slow down? Crash overall?

It would be helpful to have a round of beta testing to find bugs as you suggest. We should also write test cases to put the program through as well.