Mohamed Nezad

Applied Programming Concepts

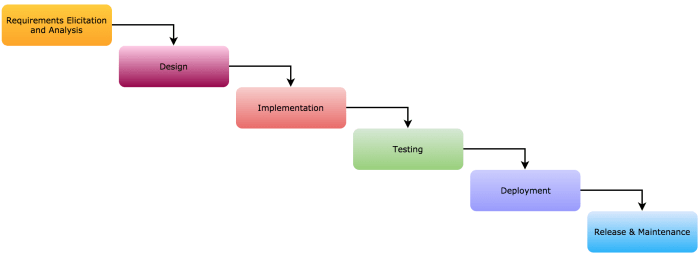
Andrew Lin & Jason Pham

05/30/2021

Process Model and Version Control

**WATERFALL MODEL**

* The result of each phase is one or more documents that are approved.
  + The following phase should not start until the previous phase has finished.
  + In practice, these stages overlap and feed information to each other.



**Requirements, analysis and definition**: establish the services, constraints and goals.

* Services are scheduling system that allows instructors, students, and admins to add courses, search for courses, print schedules, add/remove courses, add/remove students from a course, print class lists and rosters based on the specific role of the user.
* Constraints are that there can only be total of 100 students, 10 instructors, and 1 admin. We will, however, test with fewer than that.
* Goals are to have a scheduling system lie Leopardweb, and add more functionality to it, so that everyone is able to utilize it accordingly.

**System software and design**: establish the overall system architecture and define the fundamental software abstractions and relationships.

* Includes database of users, instructors, admin, such as their first name, last name, and ID. It will also include the course information, along with its course-registration number.
* Base class will be the type of user (student, instructor, or admin) and according to that, the functions will be added on the different types of utilization of the program they have.

**Implementation and unit testing**: write the code and test the components.

* Each code will be written accordingly in terms of who possesses what type of control ranging from a instructor to student to admin.
* It will also be tested individually by each component, to make sure that the type of access is accurate varying from different derived classes.

**Integration and system testing**: integrate the components into a system and test the system.

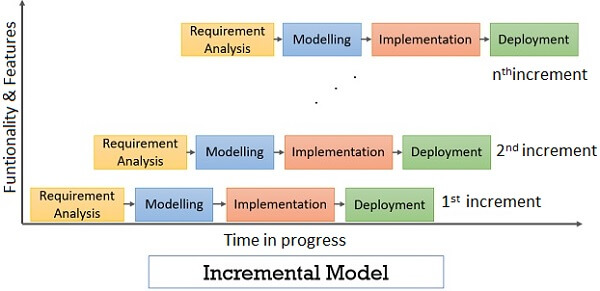
* Every component will be tested together, from different schedules to different courses. It will also be tested by entering user as either admin, instructor or student, and check if the accurate and corresponding controls are in place for each.
* It will also check whether the information being printed out is correct or incorrect.

**Operation and maintenance:** install and use, fix bugs, and update.

* Bugs and fixes will be made as different subjects are tested.
* Upon checking on whether any error occur or any information is misplaced varying from instructors to students, it will be later determined to be a bug and fixed accordingly.

**INCREMENTAL MODEL**

* Develop an initial implementation, allow user to test it 🡪 develop other iterations, testing each time, until a final product is developed.
  + Each increment or version adds more functionality.
  + Specification, development, and testing are interleaved rather than distinct.



**Initial implementation:**

* The scheduling system will be included as instructed, along with the different course information, as well as, requirements of first name, last name, and ID number.
* Instructors shall make changes to schedules, including printing the class list, along with the roster.
* Students shall be able to view their schedule, as well as, add and drop courses, and print their overall schedule.
* Admin shall be able to add or remove students from courses, along with adding or removing the courses themselves from the system.

**Alternate iteration:**

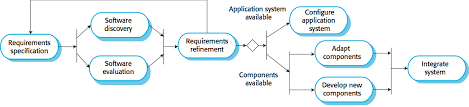
* Addition of multiple features with different functionalities from the previous, and implemented onto the initial implementation.
* Direct actions of printing schedules by admin, instructor, or student will be implemented.
* Initiation of multiple semesters, and different courses including different users, whether its student, instructors or admin will be implemented.

**Final product:**

* Combination of both the initial implementation and the alternate iteration to present a more comprehensive scheduling system with numerous new functionalities.

**INTEGRATION AND CONFIGURATION MODEL**

* Reuse-oriented.
* New intermediate stages
  + Component analysis: find and study existing software that does what you need
  + Requirement modifications: with components, revise original specs
  + System design (with reuse): use the components, modify/add as needed
  + Development and integration



**Component analysis:** find and study existing software that does what you need

* Wrike is a software I browsed online that had most of the functionality that I need in order to enhance my scheduling system for the assignment.

**Requirement modifications:** with components, revise original specs

* Clarify priorities based on the schedule each user has.
* Students can visualize their project plans, along with keep track of their work.
* Instructors may approve students work, and from there, students may accelerate based on given feedback.
* Students are also able to showcase projects they worked on as a team.

**System design:** use the components, modify/add as needed

* Clarify priorities based on the schedule each user has, along with print their courses or change their courses.
* Students can visualize their project plans, along with keep track of their work. They can also check with their instructors to be added or dropped from a course.
* Instructors may approve students work, and from there, students may accelerate based on given feedback. Admins can later add or remove a student if student didn’t improve.
* Students are also able to showcase projects they worked on as a team, while instructors may also be able to print their roster and class list and keep track of the teams.

**Development and integration**

* Initial modifications and system design can combine into one integration that can be developed overtime by numerous amount of testing and visualization of project.

https://try.wrike.com/scheduling-software/?utm\_medium=cpc&utm\_campaign=scheduling&utm\_content=listing&utm\_source=capterra&dclid=CPyhqMz38vACFZLrswodHjEOaA