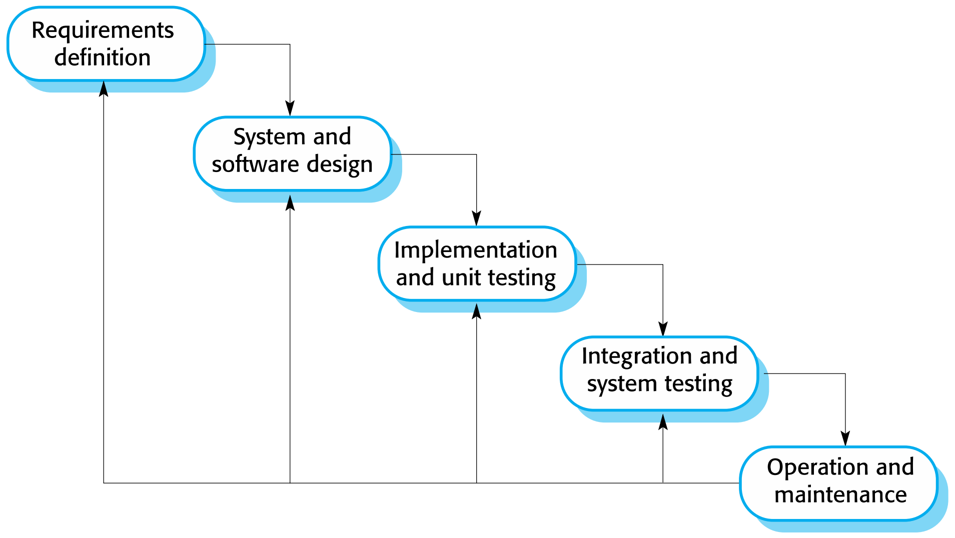
Tucker Moncey

5/23/2024

ELEC-3225-03B

Assignment 2: Part 1

Waterfall Model Diagram:


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

Phase Descriptions (General):

Requirements Definition: The goals, services, and constraints are defined.

System and Software Design: The overall system design and general software relationships and abstractions are defined.

Implementation and Unit Testing: The code is written, and components are tested.

Integration and System Testing: The entire system is tested.

Operation and Maintenance: The system is installed, used, and potentially updated.

Phase Descriptions (University Schedule System):

Requirements Definition:

Design a scheduling system for a university. The system will allow three different users (Student, instructor, and admin) to add courses, print schedules, search for courses, etc. Each user has different options in the system. The database could potentially work for 100 students, 10 instructors, and one admin.

System and Software Design:

There will be one base class (User) and three derived classes (Student, instructor, and admin). The user class will have three attributes (First name, last name, and ID), and methods to set each attribute and print all the attributes of the user. The derived classes will have no additional attributes. The main code will have if statements for each derived class selected by the user.

Implementation and Unit Testing:

Each class (User, student, instructor, and admin) is coded and tested. The main code is written and tested without the inclusion of the classes.

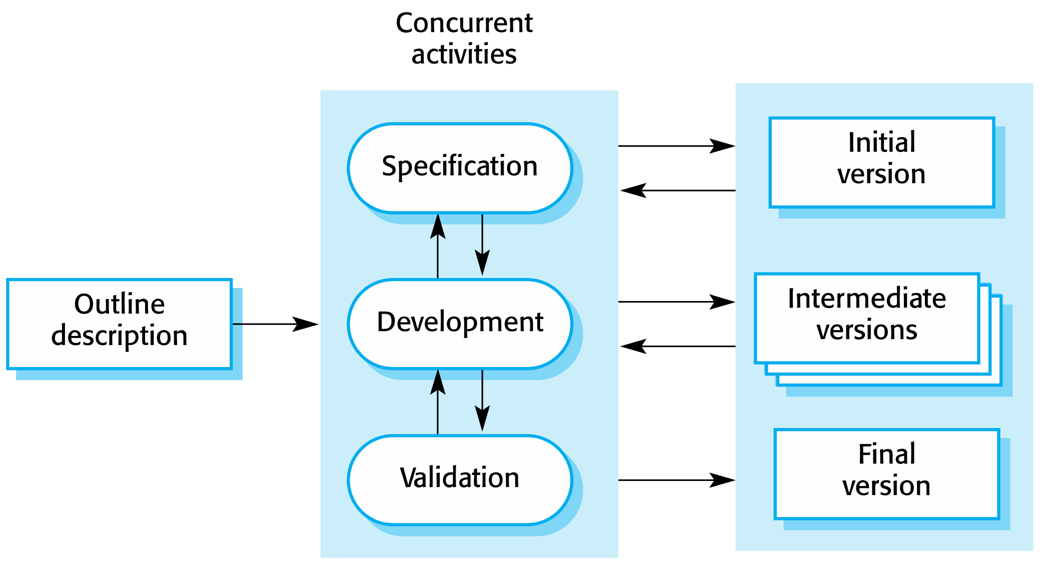
Integration and System Testing:

The classes are fully implemented in the main code and the entire code is tested using different test cases. Each option for each type of user is tested.

Operation and Maintenance:

The system is used by a student, instructor, or admin. Bugs could be fixed if a user reports a problem.

Incremental Development Model Diagram:



Phase Descriptions (General):

Each concurrent activity is tested before developing other iterations until the final system is completed. Sequential increments or versions add more functionality to the previous one.

University Scheduling System Implementation:

Basic versions of the main code and class files are tested first by the user. Based on the user's feedback, the overall system is changed. The system is split into three sections: the main code, base class code, and the derived class code. Newer versions of each of the sections are created at the same time based on user's input. For the first version, the user, student, instructor, and admin classes and main code are created with limited functionality. Based on user feedback, certain sections would be updated, and a second version would be created. Maybe the user wants more options for the student and instructor user or wants the user interface to improve in certain ways, so a second version would be developed to meet those needs. New versions would be made available to the user until the user is satisfied with the final system. Each new version will have more functionality than the previous version.

Integration and Configuration Model

General Idea:

An integration and configuration model incorporates previously existing program into a new system. Much of the code incorporates components that fit the structure of the new system. The components can be altered to better fit the system if necessary. The benefits of an Integration and Configuration Model are that the coder does not need to write code for the reused components, and the components are already fully tested.

Pieces That Can Be Potentially Used for University Scheduling System:

Code relating to a login system, user interface, or polymorphism could potentially be used for the university scheduling system. Tkinter is a Python library that can be used for the user interface for choosing the type of user and selecting different options. The user would have to log into the system, so a sample code of a login module could be used. The code could be altered to include the ID number and user type as well. To make sure the user inputs valid responses, the library PyInputPlus could be used. The system needs to display lists/tables like class lists and schedules, so the library PrettyTable could be used to easily display tables. The system would have to store the information of many different users, so a database could be used.

A Login Module in Python:

https://www.javatpoint.com/login-module-in-python

Login Interface using Tkinter:

https://www.geeksforgeeks.org/create-a-modern-login-ui-using-customtkinter-module-in-python/

PyInputPlus Description:

https://pypi.org/project/PyInputPlus/

PrettyTable Description:

https://pypi.org/project/prettytable/

Model Used for University Schedule Project:

For the university schedule project, a mixture of incremental development and integration and configuration models can be used. Using the incremental development model is ideal because the project will be completed as a group. If one group member wants to alter the code of another group member, he can use a certain version of the code uploaded by the other group member. If he alters the code in a way that damages the functionality and wants to undo the changes, the original version of the code is still available. When code is tested and approved by the group, a new version of the code can be uploaded to the repository.

Using the integration and configuration models allows the group to use pre-existing code, libraries, and databases in the system. Tkinter, PyInputPlus, and PrettyTable are Python libraries that could help develop the user interface. Many systems use a login and register module, so instead of taking the time to write and test new code, the code from a pre-existing program can be used instead.