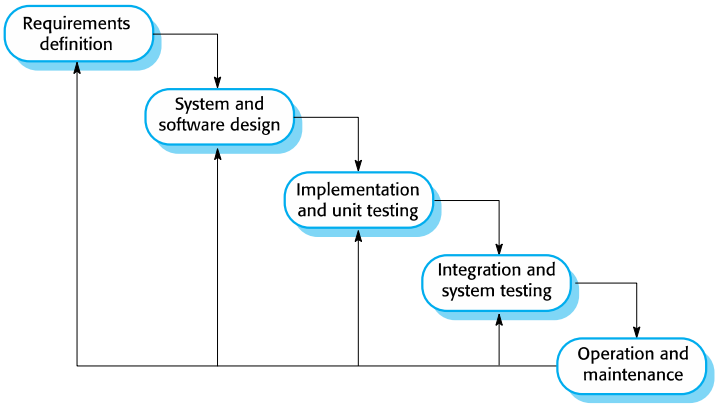
Assignment 2.1 – Process Models

ELEC 3225 – Applied Programming Concepts

Sakdipat Vijitvoravong

Waterfall Model: an approach to software development where each phase must be completed before the next one begins.




**Requirements Analysis:** Gather and document all requirements from stakeholders.

**System and Software Design:** Design the architecture and components of the system.

**Implementation:** Write the code and convert the design into a working system.

**Integration and Testing:** Combine all modules and test the system for errors and bugs.

**Operation and Maintenance:** Perform ongoing maintenance to fix bugs and make improvements.

University Scheduling System

**Requirements Analysis:**

Gather detailed for students, instructors, admin requirements. Document requirements for user authentication, course management, scheduling, and user roles.

**System Design:**

Design the database for users and courses. Plan the user interface layout and navigation.

**Implementation:**

Develop the user authentication module. Implement the database and develop operations for users and courses. Code the functionalities for students, instructors, and admin roles.

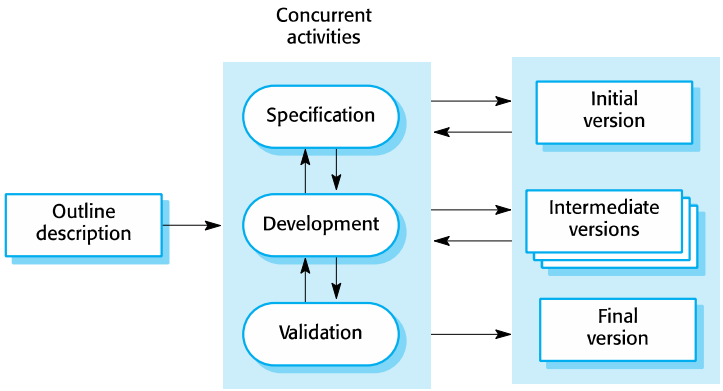
**Integration and Testing:**

Integrate modules for user authentication, course management, and scheduling. Perform testing on individual components. Conduct system testing to ensure all components work together.

**Maintenance:**

Regularly update the system based on user feedback. Fix any bugs reported by users. Add new features as requested.

Incremental Development Model: developing the system in small, manageable increments. Each increment adds a part of the functionality until the full system is implemented.




**Initial Planning:** Define the system's overall architecture and the first increment's requirements.

**Increment 1:** Develop the first set of features.

**Testing Increment 1:** Test the first increment thoroughly.

**Increment 2:** Add more features based on the next set of requirements.

**Testing Increment 2:** Test the second increment, integrating it with the first.

**Repeat until Final System:** Continue the process until the entire system is developed.

University Scheduling System

**Initial Planning:**

Define the overall architecture and identify core functionalities. Plan the increments, starting with the most critical features.

**First Increment:**

Subsystem: User Authentication and Basic User Management.

Testing: Perform tests on user login and registration. Conduct testing to ensure user data is stored correctly.

Features: User login, registration, and basic profile management.

**Second Increment:**

Subsystem: Course Management for Admin.

Testing: Perform tests on course addition, removal, and modification. Integrate with user management and test.

Features: Admin can add, remove, and edit courses. Basic course search functionality.

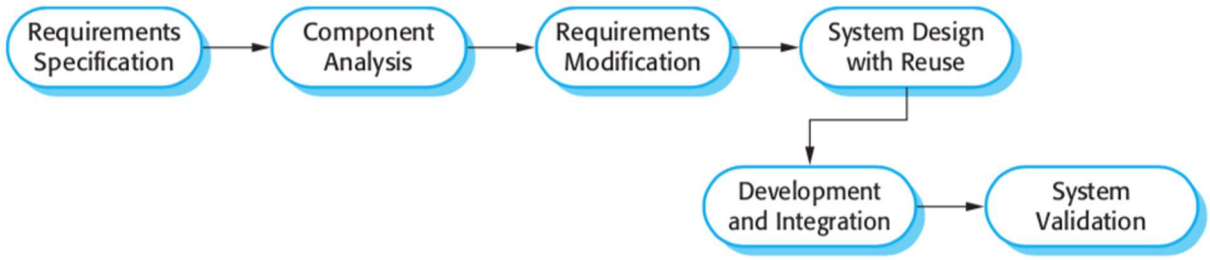
**Final System:**

Subsystem: Student and Instructor Functionalities.

Testing: Perform tests on course enrollment and schedule viewing. Integrate all subsystems and perform system testing.

Features: Students can register for courses and view schedules. Instructors can view course rosters and their schedules.

Integration and Configuration Model: integrating existing software components and configuring them to meet specific needs. This approach reduces development time by leveraging pre-built software.




University Scheduling System

**Software Libraries:** Use libraries for user authentication, database interaction and web frameworks.

**Databases:** Use a relational database like SQL to manage user and course data.

**User Interfaces:** Utilize front-end frameworks to build the user interface.

**Integration:** Integrate the selected libraries and frameworks, configuring them to support the specific needs of the scheduling system.

**Configuration:** Configure user roles, permissions, and course scheduling rules based on requirements.

**Best Model for University Schedule Project**

A mixture of Incremental Development and Integration and Configuration would be best for the university scheduling project. This allows the system to be manageable increments, with existing libraries and frameworks to reduce development time. This provides the flexibility to adapt to requirements.