Easy Travel With the Waterfall SDLC model

Level-3 Semester-I Project Report

Course Code: CSE 305
Course Title: Software Engineering

Submitted by
Md.Obaidullah Al Mahdy
ID: 2002010

Submitted to
Pankaj Bhowmik
Lecturer, Department of CSE

Department of Computer Science and Engineering
Hajee Mohammad Danesh Science and Technology University, Dinajpur-5200

Software Development Process for Easy Travel

SDLC Model: Waterfall

---

1. Introduction

- **Project Title:** Easy Travel
- **Objective:** The goal of the Easy Travel project is to develop a C++-based travel management system that allows users to easily manage travel bookings, schedules, and related information. The system enables users to book travel, view schedules, and access travel-related data through an intuitive user interface.
- **SDLC Model Used:** Waterfall Model
- **Reason for Choosing Waterfall Model:** The Waterfall Model is ideal for this project because it provides a clear, structured approach with distinct phases that are easy to manage. The project requirements are well-defined, and the sequential nature of the Waterfall Model ensures thorough completion of each phase before moving on to the next.

---

2. Waterfall Model Stages

2.1 Requirement Analysis

- In this phase, the primary objective was to gather and analyze user requirements for the system.
- **Activities:**
    - Identify the functional requirements for travel booking, schedule management, and information retrieval.
    - Understand user needs for a simple, user-friendly interface.
    - Collect data through user interviews, feedback, and brainstorming sessions with stakeholders.
- **Outcome:** A detailed set of user requirements that formed the foundation for the next phase of system design.

2.2 System Design

- This phase involves planning and designing the architecture and components of the system.
- **Activities:**
    - Design the overall architecture of the system using C++ modules to handle different functions like booking, cancellations, and travel information management.
    - Define the database structure to store user data, travel bookings, and schedules.

    - Create wireframes and a flowchart for the user interface to ensure a smooth interaction between the system and the end-user.

- **Outcome:** A set of design specifications, including the database schema, system flow diagrams, and a clear outline of the system's structure.

2.3 Implementation

- In this phase, the system is developed based on the designs from the previous phase.
- **Activities:**
    - Develop the system's core functionalities using C++. This includes coding for booking, schedule management, and travel information retrieval.
    - Implement the backend logic for handling user inputs, generating travel schedules, and processing booking requests.
    - Use **Code::Blocks** as the integrated development environment (IDE) and the **GCC Compiler** for building and running the system.
    - Database: SQLite for secure and efficient data storage.
- **Outcome:** A fully functional travel management system with all the required features implemented.

2.4 Testing

- Testing ensures the system functions as expected and meets the defined requirements.
- **Activities:**
    - Perform **Unit Testing** to verify the functionality of individual modules such as the booking system, cancellation features, and schedule management.
    - Conduct **Integration Testing** to ensure that all modules work together seamlessly as part of the whole system.
    - Execute **User Acceptance Testing (UAT)**, where real users interact with the system to ensure it meets their needs and is intuitive to use.
- **Outcome:** A validated system that is free from critical bugs and performs as intended.

2.5 Deployment

- In this phase, the system is deployed for user access and real-world use.
- **Activities:**
    - Deploy the software on local systems, ensuring that users can easily access and interact with the application.
    - Provide installation files, setup instructions, and user manuals to facilitate the installation process.
    - Conduct **live testing** to ensure the system works correctly in the deployment environment.

- **Outcome:** A deployed system available for user access and operational in real-world conditions.

2.6 Maintenance

- Once the system is deployed, ongoing maintenance is needed to fix bugs, update features, and ensure long-term functionality.
- **Activities:**
    - Monitor user feedback and address any issues or bugs reported.
    - Optimize system performance to handle a larger number of users and data.
    - Add new features, such as integration with third-party APIs for better travel management or online booking capabilities.
- **Outcome:** A continuously improving system that adapts to user needs and remains operational over time.

---

3. Diagram

Below is the visual representation of the Waterfall Model stages:

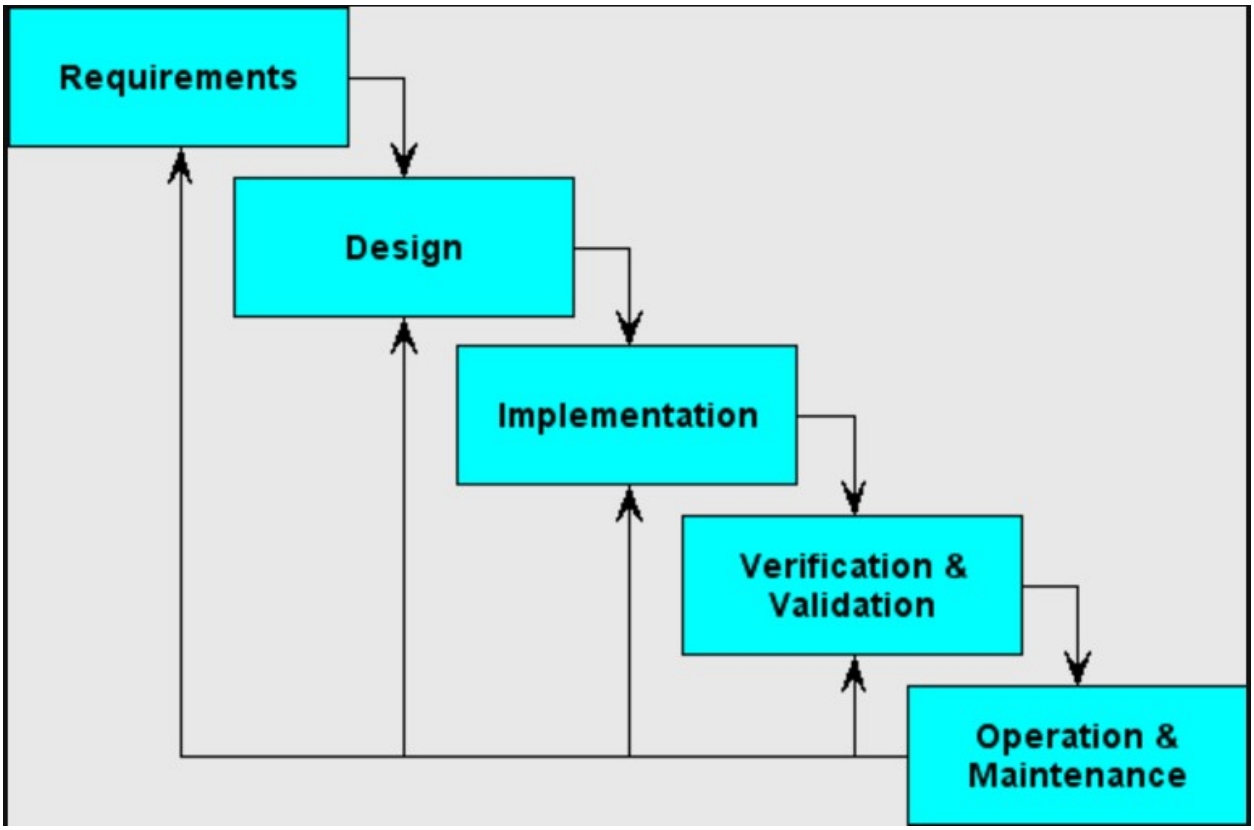

fig:waterfall SDLC model

---

4. Conclusion

The Waterfall Model allowed for a systematic, step-by-step approach to the development of the Easy Travel system. Each phase was thoroughly completed before moving to the next, ensuring the project stayed on track and met its objectives. The outcome is a reliable, user-friendly travel management system that meets the needs of its users and is adaptable for future improvements.