IT314 Lab 1 Report
Keyur Govrani
202101498

Contents

1 Choosing Software Process Models

(a) A simple data processing project.

⇒ For a simple data processing project, a linear approach would be appropriate. *The Waterfall* model can be used because the requirements are well-defined and there are no major changes expected during the development process.

(b) A data entry system for office staff who have never used computers before. The user interface and user-friendliness are extremely important.

⇒ For a system having great importance of user interface (UI) and user friendliness, the *Iterative and Incremental Development* model or the *Prototyping* model would be appropriate. These models allow for continuous feedback from users, and frequent iterations can help in modifying the user interface and ensuring that it meets the needs of non-technical users.

(c) A spreadsheet system that has some basic features and many other desirable features that use these basic features.

⇒ For a system with evolving requirements and a focus on adding new features incrementally while building on existing ones, *Evolutionary Prototyping* model would be appropriate. This model fits perfectly for the given situation because it is an iterative approach where the prototype is developed, refined, and extended until it becomes the final product.

(d) A web-based system for a new business where requirements are changing fast and where an in-house development team is available for all aspects of the project.

⇒ In an environment where requirements are changing fast and where an in-house development team is involved, the Agile development model like *Extreme Programming (XP)* can prove to be beneficial by providing continuous communication and delivery.

(e) A Web-site for an on-line store which has a long list of desired features it wants to add, and it wants a new release with new features to be done very frequently.

⇒ For a web-based system with frequent releases and a focus on delivering new features quickly, the *Incremental* or *Spiral* model would be appropriate. These models perform iterative development thereby making it flexible to make the required changes and is also helpful in Time to Market.

(f) A system to control anti-lock braking in a car.

⇒ For safety-critical systems like this, the *Incremental Waterfall* or *Evolutionary Prototyping* model would be appropriate. These models allow iterative development thereby allowing early testing and validation of safety provided by anti-lock braking.

(g) A virtual reality system to support software maintenance

⇒ For a virtual reality system with evolving maintenance requirements, *Synchronize and Stabilize* model would be appropriate. It emphasizes continuous development and teams deliver stable releases at regular intervals.

(h) A university accounting system that replaces an existing system

⇒ When replacing an existing system, we have perfect understanding of the problem and there are no changes in the requirements. Hence, the *Waterfall* model would prove to be appropriate in this case.

(i) An interactive system that allows railway passenger to find train times from terminals installed in stations.

⇒ For an interactive system with UI concerns, the *Prototyping* model or the *Iterative and Incremental Development* model would be appropriate. These models allow for frequent user feedback and modification of the UI based on iterative development.

(j) Company has asked you to develop software for missile guidance system that can identify a target accurately.

⇒ For safety-critical systems like this missile guidance system, a highly structured and comprehensive model such as the *Spiral* model would be appropriate. These models emphasize risk management, thorough testing, and verification at each stage to ensure accuracy and reliability.

(k) When emergency changes have to be made to systems, the system software may have to be modified before changes to the requirements have been approved. Choose a process model for making these modifications that ensures that the requirements documents and the system implementation do not become inconsistent.

⇒ For emergency changes and situations where requirements might not be fully defined before implementation, the *Throw-away Prototyping* model would be suitable. It allows for rapid responses to changes and maintains close collaboration between the development team and stakeholders.

(l) Software for ECG machine.

⇒ For safety-critical medical devices like an ECG machine, a model that prioritizes thorough testing and verification, such as the *Spiral* model would be appropriate. It ensures proper validation and verification of the software to meet strict medical standards at each stage to ensure accuracy and reliability.

(m) A small scale well understood project (no changes in requirement will be there once decided).

$\Rightarrow$ For a small-scale, well understood project with stable requirements, the *Waterfall* model would be appropriate. It follows a linear approach, and with clear requirements, it can help in managing the development process efficiently.