VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Name: Apurba Koirala

Reg no: 22BCE3799

Subject Code: BCSE301P

Course Title: Operating Systems Lab

Lab Slot: L45 + L46

Guided by: Dr. PEREPI RAJARAJESWARI

Lab Assessment 1

Department of Software systems

School of Computer science and Engineering

LAB ASSESMENT 1

A) You are interacting with the MIS department of a very large oil company with multiple departments. They have a complex regency system. Migrating the data from this legacy system is not an easy task and would take a considerable time. The oil company is very particular about processes, acceptance criteria and legal contracts

Answer the following questions.

Which model is best suited and, justify?

Briefly explain about the suitable model?

Answer

For the given scenario, where the task involves migrating data from a complex legacy system for a large oil company, the **Waterfall Model** is the most suitable software process model. This model is particularly effective because it is designed for projects where requirements are clearly defined, processes are structured, and documentation is a critical part of the workflow. The oil company, being very particular about processes, acceptance criteria, and legal contracts, necessitates a development approach that prioritizes well-documented requirements from the outset. The Waterfall Model ensures that all requirements are fully documented and agreed upon before any development begins, leaving little room for ambiguity. This alignment with the company's preference for clarity and precision makes it the ideal choice.

The sequential nature of the Waterfall Model is another significant advantage for this scenario. Migrating data from a complex legacy system is a challenging and time-consuming task that requires meticulous attention to detail. The Waterfall Model's phase-by-phase approach ensures that each stage is completed fully before moving on to the next. This sequential process is particularly critical for maintaining data integrity and ensuring compliance with the company’s stringent regulatory and legal requirements. By following a structured and linear progression, the Waterfall Model minimizes the risk of errors that could arise from overlapping or unfinished phases.

Additionally, the Waterfall Model emphasizes documentation, which is a key requirement for the oil company due to its focus on legal and regulatory compliance. Each phase of the model is accompanied by detailed documentation, providing a clear record of progress, decisions, and deliverables. This extensive documentation aligns perfectly with the company’s demand for accountability and traceability, ensuring that all stakeholders are on the same page throughout the development process. Furthermore, the rigid and structured nature of the Waterfall Model works well in an environment where flexibility is not a primary concern.

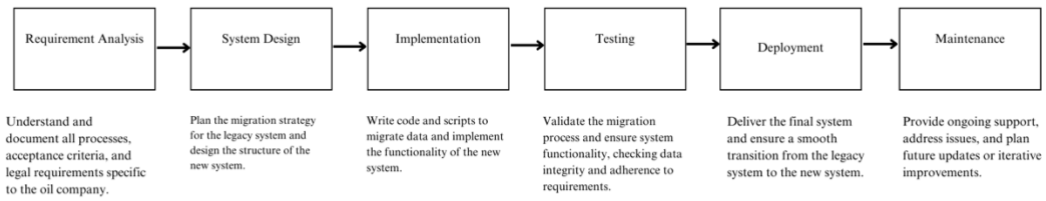
Given that the company's processes and acceptance criteria are fixed, the need for adaptability is minimal, making the Waterfall Model a natural fit.

The Waterfall Model is a traditional software development lifecycle (SDLC) model that follows a linear and sequential approach, with each phase dependent on the deliverables of the previous one. Progress flows in one direction, much like a waterfall, ensuring that each phase is completed fully before the next phase begins. The model consists of six distinct phases: **Requirement Analysis**, **System Design**, **Implementation**, **Testing**, **Deployment**, and **Maintenance**.

1. **Requirement Analysis**: In this phase, all requirements are gathered from stakeholders. This involves understanding and documenting all processes, acceptance criteria, and legal requirements to ensure there is no ambiguity in the project's scope or objectives.
2. **System Design**: This phase involves planning the migration strategy for the legacy system and designing the structure of the new system. It includes defining data structures, system architecture, and necessary modules to ensure the migration is seamless and well-organized.
3. **Implementation**: During this phase, the actual coding and scripting are carried out. These scripts are designed to migrate data and implement the functionality outlined in the system design phase.
4. **Testing**: Once the implementation is complete, the system is thoroughly tested to validate the migration process and ensure the functionality meets all specified requirements. Testing is critical for maintaining data integrity and compliance with the company's standards.
5. **Deployment**: After successful testing, the system is delivered to the client for use. This phase ensures a smooth transition from the legacy system to the newly implemented system.
6. **Maintenance**: Post-deployment, this phase handles any issues that arise, providing support and updates to ensure the system remains functional and efficient over time.

Each phase in the Waterfall Model has clearly defined deliverables and a review process, making it simple and easy to manage. For this scenario, the phases can be mapped to specific activities in context. For instance, during **Requirement Analysis**, the team would focus on understanding and documenting all processes, acceptance criteria, and legal requirements. In the **System Design** phase, the migration strategy for the legacy system would be planned and structured. **Implementation** involves writing the scripts and code required for data migration and system functionality, while **Testing** validates the accuracy and integrity of the migrated data. During **Deployment**, the system is handed over to the client for real-world use, and finally, **Maintenance** ensures ongoing support and updates as needed.

In conclusion, the Waterfall Model is best suited for this scenario due to its clear, structured, and sequential approach. It ensures that all requirements are thoroughly documented and understood, that data integrity is maintained, and that legal and regulatory compliance is met. Its emphasis on documentation and its rigid structure makes it ideal for a project where flexibility is not a priority, but precision and accountability are critical.



B) You have been appointed as a project manager for a small software products company. Your job is to build a breakthrough product that combines virtual reality hardware with state-of-the-art software. Because competition for the home entertainment market is intense, there is significant pressure to get the job done. What non-agile software process model(s) would you choose? Justify your answer for choosing the model and for not choosing the other models.

For the scenario of managing a project to develop a groundbreaking product that combines virtual reality hardware with state-of-the-art software, the **Spiral Model** is the most appropriate non-agile software process model. This model is highly suitable for projects involving complex and innovative technologies because it focuses on iterative development, risk management, and continuous refinement. In a competitive industry like home entertainment, where both technical challenges and time-to-market pressures are significant, the Spiral Model provides a structured yet flexible framework. Each iteration, or "spiral," allows the team to progressively refine the product while also addressing potential risks at every stage. A key feature of this model is its emphasis on prototyping during each phase, which is critical for testing and validating the integration of VR hardware and software early in the development cycle. Furthermore, the Spiral Model incorporates regular customer feedback, enabling the team to align the product with evolving user expectations and market trends. This iterative feedback loop helps ensure that the final product meets both technical and user requirements, reducing the likelihood of costly changes later.

Other models, such as the **Waterfall Model**, are less effective for this type of project due to their rigid and linear structure, which does not allow for flexibility or iterative development. The Waterfall Model requires all requirements to be fixed and fully documented upfront, which is unrealistic in a project involving cutting-edge VR technology, where requirements are likely to evolve. The **Incremental Model**, while offering phased delivery, does not provide the comprehensive risk management capabilities that the Spiral Model offers, nor

does it emphasize prototyping to the same extent. Similarly, the **Big Bang Model** is entirely unsuitable because it involves minimal upfront planning and relies heavily on a last-minute, all-at-once integration of components. This approach is far too risky for a project involving the intricate integration of hardware and software. Lastly, the **V-Shaped Model** is excluded because its sequential focus on testing after development makes it less adaptable for projects requiring iterative prototyping and frequent adjustments, which are essential for innovative and complex systems like this VR-based product.

In conclusion, the Spiral Model stands out as the best choice due to its iterative nature, strong emphasis on risk management, and ability to accommodate changes in requirements. Its support for prototyping and customer feedback ensures that the product is continuously refined to meet technical challenges and user needs. In the highly competitive and fast-evolving home entertainment market, where precision and adaptability are key, the Spiral Model provides the necessary balance between structure and flexibility, making it the ideal framework for developing this cutting-edge VR product.