# Assignment # 1

# Software Development Process Models

Abdul Saboor

20L-1113 (BDS-4A)

# Waterfall Model (Classical)

Application:

1. Requirements are very clear, well documented, fixed and not ambiguous.
2. Product definition is stable and technology is well understood.
3. The resources and requirements available are sufficient enough to support the project.

Advantages:

1. This model is simple and easy to understand.
2. This model is easy to manage as it is very rigid. The process is divided into phases where each phase has specific deliverables and a review process.
3. We focus on and complete one phase at a time.
4. There are clearly defined stages and well-understood milestones.
5. The whole process and results of development are well documented.

Dis-advantages:

1. There is no working software until the last stages of the life cycle.
2. There is a high number of risks and uncertainties involved in the process.
3. This is not a good model for long, ongoing, complex or object-oriented projects.
4. Measuring progress between individual stages is difficult.
5. There is no possibility of accommodating any change in the requirements or scope during the life cycle of the project.

# Waterfall Model with Prototyping

Application:

1. This model is used when the system is designed will need to have a lot of interaction with the end-users.
2. This is well-suited for systems like online systems and web interfaces, this model allows for ease of use and there is the minimum training required for the end-user.
3. The end users are constantly providing feedback which is incorporated in the prototypes. This means that this model is excellent for designing good human-computer interface systems.

Advantages:

1. The users are actively involved in system development.
2. The closer interaction with prototypes gives a better understanding of the system to the users.
3. Errors can be detected in the early stages of development.
4. The quick user feedback leads to better solutions.
5. Any confusing requirements or missing functions can be rectified at early stages.

Dis-advantages:

1. There may be incomplete and inadequate problem analysis.
2. There is a quick implementation but then it may lead to repairing the implemented system.
3. This is more costly and a longer time is required for the development.
4. The documentation is usually poor because there is a constant change in user requirements.
5. Sometimes a hurry to develop prototypes may lead to sub-optimal solutions.

# V-Model

Application:

1. The requirements are clear, well documented and fixed.
2. The product definition is stable.
3. The project is of short duration.

Advantages:

1. This model is highly disciplined and each phase is completed one at a time.
2. The project duration is short and requirements are very well understood.
3. The process is simple and easy to understand and use.
4. The verification and validation in the early stages of the project give a high probability of an error-free final product being delivered to the customer.
5. It enables efficient project management and tracking of progress.

Dis-advantages:

1. There is a high risk and uncertainty factor involved.
2. It is not suitable for complex or object-oriented projects.
3. This is not recommended for long and ongoing projects.
4. Concurrent events are not easy to handle.
5. It is difficult to go back and change functionalities once the application is in the testing phase.

# Phased Development

Application:

1. When requirements are known upfront.
2. When well-defined module interfaces are required.
3. Helpful when working with new technology as incremental resource deployment means fewer errors.

Advantages:

1. This process divides the project into many small deliverable parts.
2. There is a smaller time when the initial requirements are gathered and the first release is given to the customer.
3. Errors are identified in the early stages.
4. Allows for parallel development of the system.
5. The initial delivery cost is lowered.

Dis-advantages:

1. Huge peak resources may be required.
2. There may be fewer resources available for any particular module.
3. There is a high risk of total system failure.
4. The time between development and deployment maybe longer.
5. There is extensive use of temporary resources.

# Incremental (Iterative) Model

Application:

1. New technology is being used that is being learnt by the development team while working on the project.
2. Resources and required skill sets are not available and are being utilized on a contract basis for specific iterations.
3. There are some high-risk features and goals which may change in the future.

Advantages:

1. Some working functionality can be developed quickly and early in the life cycle of the project.
2. Changing requirements or scope is less costly.
3. Testing and debugging smaller iterations is easier.
4. Risks are identified and resolved during iterations meaning each iteration can be easily managed as a milestone.
5. This is well suited to large and mission-critical projects.

Dis-advantages:

1. More resources are required for the development.
2. More management attention is required.
3. System architecture and design issues may arise as all the requirements are not gathered at the beginning of the life cycle of the project.
4. Defining increments require the definition of the complete system.
5. Highly skilled resources are required for risk analysis.

# Prototyping Model

Application:

1. This is useful in systems having a high level of user interaction.
2. The ease of use means minimal training is required for end-users.
3. This is useful when there is a quick change in requirements.

Advantages:

1. There is increased user involvement in the product even before its implementation.
2. A working model of the system is displayed which gives the users a better understanding of the system being developed.
3. Since defects are detected earlier, the time and costs are reduced.
4. Quick user feedback leads to better solutions.
5. Confusing functions and missing functionalities can be identified early on.

Dis-advantages:

1. Large dependence on prototypes increases the risk of insufficient requirement analysis.
2. Confusion among prototypes and actual systems may arise among users.
3. The scope of the system may expand beyond original plans which may increase the complexity of the system.
4. Developers may try to re-use existing prototypes to build an actual system when it may not be technically possible.
5. If not monitored properly, the effort invested in building prototypes may be too much.

# Spiral Model

Application:

1. When budget constraints and risk evaluation is important.
2. When requirements are complex and we need to evaluate them for clarification.
3. When any new product lineup should be released in phases to get customer feedback

Advantages:

1. Changes in requirements can be accommodated.
2. This model allows extensive use of prototypes.
3. The requirements of the project can be captured more accurately.
4. Users can see the system early on in development.
5. There is better risk management as development can be split into parts where risky parts can be developed earlier.

Dis-advantages:

1. Project management is more complex.
2. This model can be expensive for small and low-risk projects.
3. The development process is comparatively complex.
4. The spirals may go on for an indefinite time and an end may not be known.
5. Extensive documentation is required for a large number of intermediate stages.

# Unified Process Model

Application:

1. It is suitable for applications where project is expected to customize and extend.
2. It is suitable for projects where we have rapidly changing requirements.
3. It is suitable for projects where we might expect major changes to architecture in future.

Advantages:

1. This model provides object-oriented system development approach.
2. This model can be easily tailored to adapt to organizational needs.
3. The iterative cycle allows to quickly adapt to changing requirements.
4. This model emphasizes on the need for an accurate documentation.
5. The model is very risk focused.

Dis-advantages:

1. There is a need of experts to develop the methodologies.
2. The process can be very complex and disorganized sometimes.
3. Utilizing new technologies may produce components that may not be reusable.
4. The implementation can be very challenging for small projects.
5. The integration in development process can have impact on fundamental activities during testing.

# Agile Methods

Applications:

1. When our highest priority is to satisfy the customer through delivery of valuable software.
2. When we may expect a change in requirements, even in the late stages of project.
3. When the system developers like to interact with the stakeholders for development.

Advantages:

1. Deployment of software is quicker and thus helps in increasing the trust of the customer.
2. Can better adapt to rapidly changing requirements and respond faster.
3. Helps in getting immediate feedback which can be used to improve the software in the next increment.
4. People and interactions are given a higher priority rather than process and tools.
5. Continuous attention to technical excellence and good design.

Dis-advantages:

1. In case of large software projects, it is difficult to assess the effort required at the initial stages of the software development life cycle.
2. The Agile Development is more code focused and produces less documentation.
3. Agile development is heavily depended on the inputs of the customer. If the customer has ambiguity in his vision of the final outcome, it is highly likely for the project to get off track.
4. Face to Face communication is harder in large-scale organizations.
5. Only senior programmers are capable of taking the kind of decisions required during the development process. Hence, it’s a difficult situation for new programmers to adapt to the environment.