**System Development Life Cycle (SDLC) + 3 development models**

What is System development Life Cycle?

System development life cycle consist of planning, implementation, testing, documentation, deployment/ maintenance and lastly maintaining.

Starting with the first is planning, which is the most important part of software development. Requirement gathering, or requirement analysis are usually done by the most skilled/ experienced software engineers in the company. After all the requirements are gathered from the client, the scope of the project is determined and documented.

Followed by implementation, the software engineers will start writing the code that’s follow by the client requirements. After the coding, we’ll begin testing, this process is to find defect or bugs in the software. Next step will be documentation, every step of the current project is being documented down for future reference and improvement of the software in the process.

Last few step, deployment and maintenance. To deploy the software after it been approve for release. Finally maintaining, software maintenance is required for future reference and improvement and new request by client.

3 Development models

Waterfall model, the very old model. Biggest flaw in waterfall model is that the testing phase only start after development is completed, as such many defects and failures are reported at the end. With this the cost of fixing these issues are high. However, waterfall model allows for early design changes. Also allow to adapt to shifting teams, well suited to large teams that may see members who come and go throughout the process of the project. The main idea of the waterfall model is to spend most of time, money and effort up front. Roughly twenty to forty percent in the first two phases, thirty to forty percent on coding / development, and the rest during implementation and maintenance.

Disadvantages of waterfall model is not well-suited for long or large-scale projects, lack of flexibility in the model means that projects prone to high levels of change may meet greater difficulty, the dependency of later stages upon earlier stages can make small changes in the project disproportionately challenging. Once the software reaches the testing stage, it can be difficult to go back to the development or requirements stage to fix something.

Advantages of waterfall model is it’s well-suited to projects where quality is emphasized over cost or time requirements, simple to implement, requiring relatively little in terms of resources and time. More transparent, such as demonstrable output can be seen by the client and project manager after every stage. Easy to communicate with clients, especially during the critical requirements and design phase.

Incremental model, this model is broken down into multiple standalone modules of the development cycle. In incremental development it is done in steps from analysis design to implementation to testing/ verification and finally maintenance. After each subsequent release of the system it will add function to the previous release until all designed functionality has been implemented. The advantages of this model is that software will be generated quickly, flexible and less expensive to change requirement and error are easily identified.

Disadvantages of incremental model is that his require good planning/ designing. Rectifying a problem in one unit requires correction in all units which will consumes a lot of time. Problems may arise pertaining to system architecture because not all requirements are gathered up front for the entire software life cycle, each phase of an iteration is rigid and do not overlap each other.

Advantages of incremental model is that it’s more flexible, less costly to change scope and requirements. Easier to manage risk because risky pieces are identified and handled during its iteration, generates working software quickly and early during the software life cycle.

So, when to use incremental model? This model is used where requirement are straight forward and clear which can implement by phase wise. By understanding he figure that the requirements is divided into small parts and being delivered accordingly.

Prototype model, this model is developed based on currently known requirement. Client can get an understanding of the system since the interactions with the prototype can enable the client to better understand the requirements of the system. Prototype is excellent for complicated and large systems for which there’s no manual process or existing system to help determining the requirements.

Disadvantages of prototype model that this methodology may increase the complexity of the system as scope of the system may expand beyond original plans and incomplete application may cause application not to be used as the full system was designed.

Advantages of prototype model are, errors can be detected much earlier, quicker user feedback leading to better solutions. Users are actively involved in the development and confusing or difficult functions can be identified. This methodology a working model of the system is provided, the users get a better understanding of the system being developed.

Best to use prototype is when the desired system needs to have a lot of interaction with the client/ end user. Also, usually online system web interfaces have high amount of interaction with client/ end user are best suited for this prototype model. Prototype model also ensures that client/ end user constantly work with the system and provide suggestion or feedback which will result in more usable system.