Software Life-Cycle models

As software development gained popularity, there were a number of incidents where the Software failed to work as intended and it was found that major software crisis happened due to human error. Hence a strong need to come up with a standard model to develop software raised [1].

Waterfall Model

Principle:

The waterfall model involved separate and distinct phases of specification and development [2].This was proposed by Winston Royce in his article "Managing the Development of Large Software Systems,"1970 [3]. He recommended that when a computer program is being written, it should be written twice and the second version to be delivered to the customer [3].The same approach has been developed into today's waterfall concept with sequential requirement analysis, design and development phases. The present water fall model has the following phases defined [2]:

- Requirements analysis and definition
- System and software design
- Implementation and unit testing
- Integration and system testing
- Operation and maintenance

Strengths:

1. This model provides high visibility for effective and efficient management [1]
2. If the requirements are fixed during the project's initial requirement analysis phase and are not subjected to change, this process is very apt for those situations [1]
3. The process is document driven and involves formal change management [1]

Weakness:

1. Some of the phases may overlap and clear distinctions cannot be drawn [1]
2. If the requirements are changed by the customer, it will have a huge cost and work overhead [1]
3. The system performance and feedback are available at the very end of the project phase and would require a lot of rework to make it better [1]

Applicability:

1. One can use this method when the requirements are frozen [1]
2. Project is of short duration and scope is clearly defined [1]
3. Very large, complex system development that requires extensive documentation [1]

Incremental Model

Principle:

According to this Software process, instead of delivering the whole software solution as a single system delivery, the development and delivery is broken down into increments called builds. Each

build involves delivering a part of the total list of functionalities. Customer requirements are prioritised and the highest priority requirements are developed and delivered in the initial builds. Once the development of an increment is started, the requirements are frozen [2].

Strengths:

1. This model results in lower software failure as the highest priority features are delivered initially and extensive testing will happen after each increments. This will help make sure that the main requirements are tested thoroughly [2]
2. The customer can start using the software with the first few increments [2]
3. Customer can use the early increments as prototypes and gain experience that controls their requirements for later system increments [2]
4. Lower risk of overall project failure [1]

Weakness:

1. Can be more expensive [1]
2. Each and every build or increment should deliver a functionality and should not mess up the previous increments [2]
3. It will be difficult to map customers requirements onto the increments as the project progresses [2]

Applicability:

1. One can use this method when the requirements are prioritised and have different delivery deadlines [1]
2. Project is of long duration and customer requirements are partially available [1]
3. Very large, complex system development that requires phase by phase development by adding functionalities [1]

Practical Implementations of incremental model:

In 1957, Project Mercury which was the seed bed of IBM Federal Systems Division started the tradition of incremental development by assembling the new share operation system [3]

The Light Airborne Multipurpose System, part of the US Navy's helicopter-to-ship weapon system was incrementally delivered as a part of four-year 200-person-year effort involving millions of lines of code, LAMPS project in 45 time-boxed iterations [3]

Microsoft Office project was also a product of incremental development where additional features were added around the base functionalities and releases with a new version licences, Eg: MS Office 98, MS Office Me, MS Office 2010 etc. [4]

RUP Model

The RUP model is a modern process derived from the UML and Associated process [2]. It has 3 perspectives:

- A dynamic perspective that shows phases over time [2]
- A static perspective that shows process activities [2]
- A practice perspective that suggests good practice [2]

This process also divides the development process into different phases but unlike waterfall model, in this process the activities are divided into workflows and multiple activities can be carried out in

parallel. The IID process practice like developing individual component incrementally and iteratively is adapted in this model. This promotes change management and improves software quality [2].

References:

[1] Prof. Casper Lassenius, T-76.3601, Introduction to Software Engineering-Software Process Slides

[2] Software Engineering, 8th edition, Ian Sommerville, NC, 2006, Chapter 4

[3] Iterative and Incremental Development: A Brief History by Craig Larman and Victor R. Basili

[4] Software Development on Internet Time by David B.Yoffie and Michael A.Cusumano