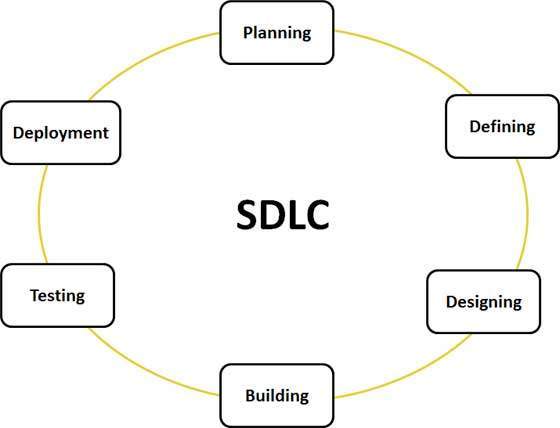
Quinones Mark Joseph Gallardo

1404555c

***Software Development Life Cycle (SDLC)***

SDLC is used in the development of an IT project it helps to describe the stages that is involved in creating the project from the idea generation trough the completion of the project.

It is also the process that is used by the software industry to test, create and design a top tier software that can meet and even exceed the customer’s expectations either through completion time or even the cost budget.

**Figure 1: Software Development Life Cycle**

***Phases of SLDC***

* **Requirement gathering and analysis**

This phase is important as it lets the developers have common ground with the users in what they want to create, thus preventing any misunderstanding. The requirements are gathered by developers and analyst from users, clients and sponsors.

There are two different types of requirements, functional and non-functional. Functional being how the system behaves while non-functional is not directly related to the functions provided by the system.

* **Design**

In this phase, the system and software design is prepared from the requirement specifications that was gathered in the first phase. Developers would then brainstorm various ideas and would present them to important stakeholders to be assessed. Once a design is chosen testers would then come up with test strategy that shows what they will test and how. The system design specifications would then serve as an input for the next phase.

* **Implementation or coding**

The longest phase of the cycle in which the actual development and completion of the product occurs. This phase can only be started once the design documents and workload allocation has been done after that actual coding begins by following coding guidelines that are outlined by the organization and programming tools that is used to generate the code. Programming language is chosen based on the type of software being developed.

* **Testing**

This phase makes use of the test strategy that was created on the design phase to test the codes using functional testing against the requirements to ensure that the product is able to meet all the needs that was addressed on the requirement gathering phase.

Any defects or problems with the product is then reported, tracked, fixed and retested until it reaches a quality of standard.

* **Deployment**

This phase only takes effect after the product is successfully tested and is deemed to have met the requirements. The product is first given to customers to do beta testing in which developers look for feedback regarding any bugs or if any changes is necessary. Once the changes and bugs are fixed, the final deployment then occurs.

* **Maintenance**

This phase takes care of the process in which the product is taken care of by fixing any new problems that comes up from time to time and providing product improvements.

There are two different types of maintenance, Remedial and Adaptive. Remedial is usually caused by inadequate testing and is used for correcting errors in the system. Adaptive is due to the fact that the software is deemed “soft” that’s why it is easy to modify so that they would then improve, change or add new features.

**Software development Models**

**Waterfall Model**

The first widely used model in software engineering to ensure a successful project. In this approach, the process of software development is separated into different phases. With this model, usually, the result of one phase acts as the input for the next phase.

**Figure 2: Waterfall Model Diagram**

The phases cascades in which progress is seen flowing downwards like a waterfall through the phases. The next phase could only start after a defined set of criteria is met for the previous phase and is signed off. This model doesn’t allow the phases to overlap.

Pros

* Easy to control and monitor
* Each task produces a well-defined deliverable
* Task are smaller and more manageable

Cons

* Product is only seen at the end. No opportunity to validate user requirements at the early stages of development.
* Problem in an earlier stage can’t be fixed.
* Does not stress the need for anticipating changes

**Prototyping**

Prototype is a working model of the software that is drastically scaled down and has a limited functionality. The main aim of creating a prototype is to clarify requirements, design user interface, demonstrate feasibility and verify that the new technology will work.

Steps to design a software prototype:

1. Basic Requirement Identification

2. Developing initial the prototype

3. Review of the Prototype

4. Revise and enhance prototype

There are multiple Software prototyping types that are used in the industry the following are used widely.

* Throwaway

-This type of prototyping requires minimum effort and requirement analysis to develop a prototype. The moment the actual requirements are understood, the prototype is discarded and the actual system is developed with a more clear understanding of the requirements of the user.

- Some of the techniques for this method is making use of a high level language, reusing components, ignoring error handling, omitting features and ignoring functions.

* Evolutionary

-This type of prototyping is based on creating actual functional prototypes with minimal functionality at the start. The prototype developed is the heart of future prototypes in which the entire system is built. Only well understood requirements are included in the prototype and requirements are only added once they are understood.

-Useful when requirements are hard to specify.

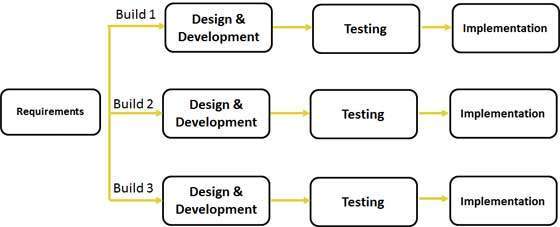
Pros of Prototyping

* Able to clarify user requirements
* Specifications can be developed incrementally, giving users opportunity to change their mind
* Reduces time and cost as defects are detected earlier.
* Quicker user feedback is available which then leads to better solutions

Cons of Prototyping

* Changes is costly and difficult
* Difficulty in estimating, planning and managing a prototype since there is no regular deliverables.
* User can get confused by the prototypes and actual system.
* Can increase complexity of the system and risk expanding the original scope of the project.

**Iterative Model**

**Figure 3: Iterative Model Diagram**

This is a popular iterative and incremental model and is often used under the following requirements

* Requirements of the complete system is clearly defined and understood
* There is a time to market constraint
* New technology is used and is being learnt by the team working on the project
* Project contains high risk features and goals that can change in the future

**Agile Model**

**Figure 3: Agile Model Diagram**

This model is popular due to its flexibility and adaptability. The model follows the following principles.

Individuals and interactions - Self-organization, motivation, interactions like co-location and pair programming are important.

Working Software – The demo of a working software is the best means to communicate and build an understanding of the customer requirements instead of only relying on documents.

Customer Collaboration – Continuous interaction is important with the customer to ensure that the gathered product requirements is as accurate as possible

Responding to change – agile development is focused on quick responses to change continuous development.

**Application to project**

After doing this report, I was able to increase my understanding of the Software Development life cycle with my new found knowledge I now know how important it is to gather the requirement of the user accurately so that the entire project wont be sabotaged. I will now ensure that I research more about the system that my team would build and what exactly does the user requires. Furthermore, my team would ensure that the design of the system is built that in such a way it would be easier to change to cope with the ever changing needs of the user.

Shaun Chua

1404150C

**SDLC**

The SDLC is a term used in systems engineering, information systems and software engineering to describe the process of planning creating testing and deploying an information system. This also applies to a range of hardware and software configurations.

The first software development model is The Waterfall Model. The waterfall model is the first process model introduced and it is simple to understand and use as it is basically just having each phase completed before being able to proceed to the next phase.

The second software development model is The V Model. The v model is the verification and validation model which is like the waterfall model, the v shaped life cycle is a sequential path of execution of processes.

Lastly the third software development model is the Incremental model. For incremental model, the whole requirement is divided into various builds. Multiple development cycles take place here and so making the life cycle a multi –waterfall cycle.

Parmesher Singh

1405306J

**Software Development Life Cycle (SDLC)**

SDLC is applied during the development of a software project. It comprises of the phases which are associated with the creation of the project. SDLC depicts processes such as planning, implementing, testing and deploying a system.



**Figure 1: Stages of SDLC**

**Stages of SDLC**

* **Requirements analysis (Specification)**

It is the most essential and crucial in SDLC. This stage is usually done by the leading members of the project team. They will retrieve information from the customer, conduct surveys in the particular industry. The information which has been obtained is used to work out the necessary path to be taken throughout the course of the project. The product specifications will now be clearly documented and will be given the green light by the customer or user.

* **Design**

This stage is carried out with reference to the first phase which is the requirements analysis. The members of the development team will then go on and conceptualize ideas and approaches to be taken. This is all done in a Design Document Specification (DDS). The DDS is then considered by important stakeholders. The important stakeholders will then come to a conclusion and a design approach has now been chosen for the product.

* **Development**

This is where the development of the product begins and the product has been actually created. Members of the development team will have to abide by the guidelines they have been presented with by their organization as to how the code is going to be generated. As for the programming language, it depends on the type of software product being developed by the software development team.

* **Testing**

Now that the product has been built, it is time for the product to go through some test to see if the product fulfils the customer’s requirements which was mentioned earlier in the requirements analysis stage. Should there be any flaws with the product, it will be reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

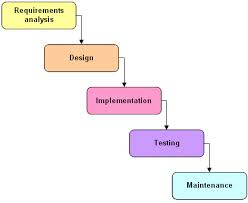
* **Deployment**

The deployment of the product is only taken place when the testing of the product has been successful and have met the product requirements. The customer will then use the product to see if they encounter any issues with the product. The customer will then provide feedback to the software development team. The development will then make changes accordingly if any.

**SDLC Models**

* **Waterfall model**

The waterfall model flows downwards like a waterfall through the stages of software development. This goes to show that any stage in the process can only start after the previous stage has been dealt with. Waterfall model does not allow members to go back to previous stages.



**Figure 2: Waterfall Model**

**Advantages**

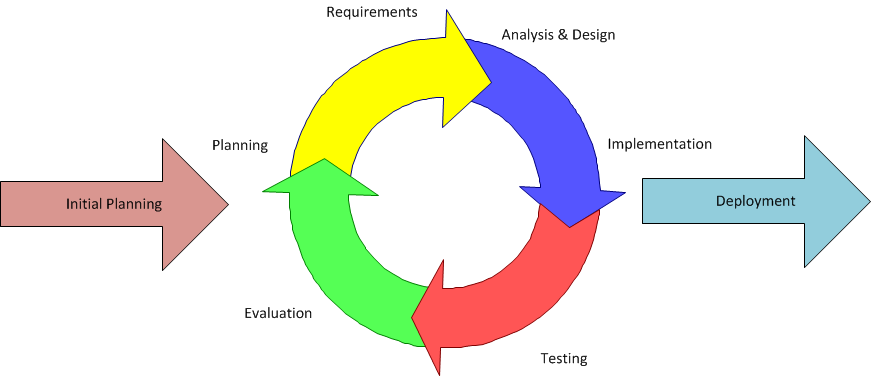
1. It is easy to explain to the user.
2. Helps to plan and schedule the project.
3. Ensures early detection of errors during the verification of each stage.

**Disadvantages**

1. Very difficult to go back to a stage when it has been completed.
2. Little flexibility.
3. Costly and more time is required.

* **Iterative and Incremental Method**

This model is created to overcome the vulnerabilities of the waterfall model. Beginning with an initial plan and ending at deployment. The purpose of this model is to come up with a system through looped cycles which fulfils the iterative section which will be in smaller portions at a time which fulfils the incremental section of this method. This enables developers to make full use of what was utilized during the development of previous versions of the product.



**Figure 3:Iterative and Incremental Method**

**Advantages**

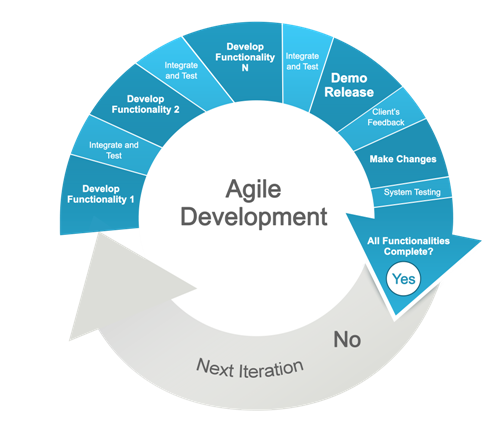
1. Allows some changes between increments.
2. Prioritises customer value than linear approaches.
3. Problems can be discovered at an early stage.

**Disadvantages**

1. Needs heavy documentation.
2. Little flexibility as a fixed set of processes have to be followed.
3. Defines increments based on function and feature dependencies.

* **Agile Development**

Agile development is based on iterative and incremental development, where the specifications and solutions emerge through association between cross-functional terms. It can be utilized with any kind of project. But the customer needs to slightly more involved and interactive.



**Figure 4: Agile Development**

**Advantages**

1. The time needed to avail some of the system features has been lessened.
2. Face to face communication and ongoing inputs from customer leaves no space for work based on guesses and estimations.
3. End product is the high quality and stellar software in the least possible time, satisfying the customer.

**Disadvantages**

1. Customer find it difficult to express their needs.
2. Documentation is only carried at later stages.
3. Reduces the usability of various components.