SDLC Report

Faizal-

Software Development Life Cycle (SDLC) is the process of building a new software from scratch until it becomes a fully functional software. It starts from the phase of coming up with the software idea and ends at the deployment of the software. There are many process models to follow which elaborates the steps to take to complete the SDLC. 3 of which are Waterfall model, Prototyping and Unified Process.

The Waterfall model goes through Requirements, Design, Implementation, Testing and Deployment in order.

Requirements engineering is the step for the consolidation of details of what the software intended to build, needs to do (i.e. functions of the software). This is the first and most important step as any misunderstandings or shortcomings in the requirement gathering process will affect all the other steps and may cost time and money. The 2nd step is the Analysis and Design. Analysis and Design is where the layout or design of the software is decided. Layout includes the Graphic User Interface (GUI) and also the structure of the software underneath. They are the architectural design and the database design. The 3rd step is the Implementation. Basically this is where all the ideas, requirements and design representations of the software is converted into actual programming codes. Lastly comes the Testing and Deployment. Testing is the phase of checking if the software is up to requirements and client’s or customer’s satisfaction. Deployment is the official release of the software either to a customer or a group of customers.

Prototyping is building a scaled down version of the target software using given specifications. It has 2 different approaches.

1st is the Throwaway prototyping. It is an approach where the prototype is built with the outline specification and is checked with the user (e.g. Client or customer) and is redone until it meets the user’s satisfaction. With that a new set of specification is delivered and the software is built based on that. The prototype is being thrown away in the end. 2nd is the Evolutionary Prototyping where the prototype is built with the initial specification and is checked with the user and is refined and implemented with more parts of the software until it meets the user’s satisfaction and the software is complete.

Unified Process (UP) uses an iterative and i­­­ncremental development process. Each and every iteration consists of all the steps of the SDLC. However at the start of the iterations, the beginning steps such as business modelling will be done more than the later steps. As the number of iterations increase, the later steps will be done more than the beginning steps. UP has 4 phases which are Inception, Elaboration, Construction and Transition. This 4 phases are similar to that of the ­­waterfall model. Inception is the gathering of the requirements and specifications. Elaboration is the establishment of the architecture, interface and database design of the software. Construction is the implementation of the software which is the coding of it. Lastly Transition is the deployment of the completed software.

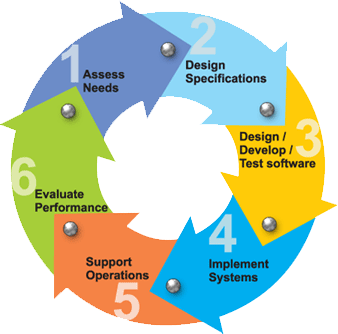
Syazwani-

**Definition of Software Development Life Cycle**

Software Development Life Cycle also known as SDLC in short is fundamentally a sequence of steps that provide model for the development and lifecycle management of an application or software. SDLC is used by systems engineering, information systems and software engineering to describe a process for planning, creating, testing, and deploying an information system.

The life cycle defines a methodology for improving the quality of software and the overall development process. Even though the organizing system may differ throughout different industries and organizations, ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required and represent processes that establish a life cycle for application or software to provide development, acquisition, and configuration on maintaining the software.

**Phases of Software Development Life Cycle**

****

There are 6 phases in every SDLC Model. Following are the phases:

1. Requirement gathering and analysis
   * In order to hold this phase to determine the requirement, they need to do meetings with managers, stakeholders and users. During the requirement-gathering meeting they will be questioned – Who is going to use the system? How will they use the system? What data should be input into the system? What data should be output by the system?

These questions were analyzed for their validity and the possibility of incorporating the requirements in the system. Lastly, a Requirement Specification Document is created.

1. Design
   * The system and software design is prepared from the specification of the first phase. The System Design would help to specify hardware, system requirements and helps in defining the overall system architecture.
2. Implementation
   * The work is divided into units followed by starting of the actual coding. During this phase, the code is created for the developer. This phase would be the longest and the most crucial.
3. Testing
   * The codes are then tested against the requirements to validate that the product met the requirements addressed during the first phase. During this phase of unit testing, integration, testing, system testing and acceptance testing were done.
4. Deployment
   * Once successful, it is delivered or deployed to the customer for their use.
5. Maintenance
   * In this phase, it is normal for customers to receive problems from the product from time to time. Therefore, maintenance should be resolve when actual problems arise.

**Software Development Life Cycle Models**

There are numerous SDLC models defined and designed that are followed during software development process. Each processed model shadows a series of steps unique to its type, in order to ensure success in process of software development.

* Waterfall Model



* + The Waterfall Model was the first Process Model to be introduced also known as a linear-sequential life cycle model.  This model is very easy to understand.  In a waterfall model, each phase must be fully completed in order for the next phase to begin and phases do not overlap. This type of model is used for smaller project, as there are no indeterminate requirements. At the end of each phase, there will be a review to verify if the project is on the right path and to decide whether or not to continue or discard the project.

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| Simple and easy to use. | Not suitable for the projects where requirements are high risk of changing. |
| Easy to manage. | No working software is produced. |
| Phases completed one at a time. | High amounts of risk and uncertainty. |
| Smaller projects as requirements are well understood. | Not a good model for complex and object-oriented projects. |
|  | Poor model for long and ongoing projects. |

* V-Model

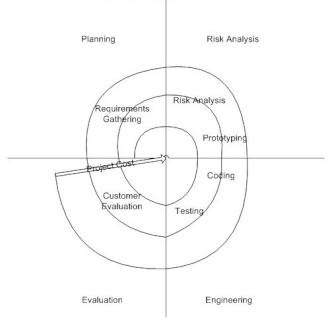


* + The V-Shaped life cycle is a sequential path of implementations of processes. Each phase should be completed before the next phase begins.  Testing of the product is planned in parallel with a corresponding phase of development.

**Both** Business Requirement Specifications (BRS) and System Requirement Specifications (SRS) requirement are needed to begin the life cycle model. However, before development starts, a system test plan is created.  This test plan focuses on meeting the functionality specified in the requirements gathering.

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| Simple and easy to use. | Very rigid and least flexible. |
| Testing activities like planning, test designing happens well before coding. | No working software is produced. |
| Smaller projects as requirements are well understood. | If any changes happen in midway, then the test documents along with requirement documents has to be updated. |
| Proactive defect tracking. |  |
| Avoids the downward flow of the defects. |  |

* Spiral Model



* The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation Phases. A software project repeatedly passes through these phases in iteration. The baseline spiral, starting in the planning phase, requirements are gathered and risk is assessed.

Planning Phase: BRS and SRS are gathered during this phase.

Risk Analysis: A process is commenced to identify risk and alternate solutions. A prototype is produced at the end so that if any risk is found during the then alternate solutions are suggested and implemented.

Engineering Phase: Software and testing is developed.

Evaluation phase: This phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral.

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| High amount of risk analysis hence, avoidance of risk is enhanced. | Costly. |
| Good for large and mission-critical projects. | Risk analysis requires highly specific expertise. |
| Strong approval and documentation control. | Project’s success is highly dependent on the risk analysis phase. |
| Additional functionality can be added at a later date. | Doesn’t work well for smaller projects. |
| Software is produced early in the software life cycle. |  |

Ronald-

SDLC stands for Software Development Life Cycle. A Software Development Life Cycle is essential as it shows the series of phrases that provide models for the development and lifecycle management for an application or software. The benefit of SDLC is that the process will help to developed software that is cost-efficient, active and higher quality.

Software Development Life Cycle has a few phrases from the start to the end. It will begin from an inception of ideas within the team of developers to requirements engineering will be used to find out what the user wants to do with the software. Following by analysis & design phrase that is a phrase where developers uses a combination of text and diagrams to display out the requirements for the software and creating blueprints of architectural design, user interface design, database design to show the client’s After that moving to the phrase of implementation of coding which is converted whatever the design into instructions written into programming languages to backbone the whole software. After the development of the software, the software will be in testing phrase whereby developers must ensure all the requirements are met and the software reliable to be used without any technical errors. Lastly, deployment phrase is the distribution of software to a group of selected customers before prior official. Enabling the clients to get first hand experience of what the software is capable of use of. Clients are able to use the newly develop software to solve their problems or improve their day-to-day process or to analysis certain business pattern to further increased the productivity of an enterprise.

Models use by Software Development Life Cycle like Waterfall, Unified and agile methods are step-by-step methods to take when developing a project. It can be use to analyze and monitor the process of the software which are currently being developed and make adjustments.

Firstly, Waterfall model a commonly use model in Software Development Life Cycle. The main aim of the waterfall model is to divide task into smaller components making it more manageable. This will ensure each task produces quality results to the project as a whole. Waterfall models are easy to managed, as activity are deal one at a time. This will also prevent errors from stockpiling and hinder the software development as developers will have to spend time figuring out the errors made. This model is most suited when developers already have similar software in the past and would like to revamp the whole software or improving it by developing another one. By using this model developers must be aware of all its domains and the rough idea of outcome of the new software that is about to be developed.

Secondly, Unified Process (UP) another commonly use model in Software Development Life Cycle. This model leads the software development process in iterations. As the model use an iterative fashion to accommodate ongoing risk. Making each phrase consisting of one or more iteration. Each of the iteration will consist of analysis, designs, coding and testing. Using the iteration method in UP models allows software first developed in small scale and all the steps taken are carefully monitor. Then, on every iteration, more features and modules will be designed, coded, tested and added to the software. Each cycle will produces a better version software compared to the last, and this will continue until the software is completed and ready to be deployed. The benefit of using this model for Software Development Life Cycle is that after one iteration is over, the developing team can work on the risk management and prepare for the next iteration. As the cycle will include a small portion of the whole software process, this will make it easier for them to manage the development process but however consuming more resources. The iteration in the UP model has 4 phrases that which is inception, elaboration, construction and transition. Starting from inception will allow sufficient studies to access whether the system is worthwhile developing. Following right after is elaboration, listing out a complete list of requirements, design architecture and develop the project and risk plans. Then the process will move into a construction phrase of developing the system from the design architecture. Lastly when the development is done and the system is stable for deployment it will be distributed to its clients to put the system into use.

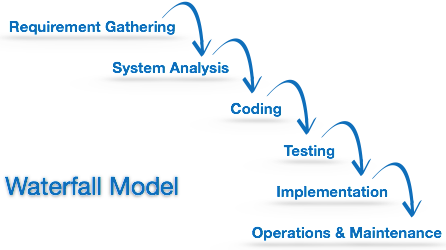
Finally the agile method used in Software Development Life Cycle. Agile development provides opportunities to assess the direction throughout the development lifecycle. Regular cadences of work also know as sprints, at the end of teams will have to present a potentially shippable product increment. Focusing on the repetition of abbreviated work cycles as well as functional product the team wish to develop. Agile method is iterative compared to waterfall, where development team is only got a single shot at getting all the aspect right in track. In agile paradigm, every phrase of development from requirements to design is continually revisited. Stopping the process of the development of the software every 2-month to re-evaluates the direction of a project every two week, before steering to another direction. Using “inspect and adapt” approach in this Software Development Life Cycle model, it reduces the development cost and time to market. Teams are able to develop the software at the same time gathering requirements, analysis paralysis that is likely to impede a team from making progress. Team’s work cycle is limited to two weeks, as stakeholders have recurring opportunities to calibrate the release for success into the real world. Agile development helps corporates to build the right system/software. Instead of committing to market a piece of product that has not been written yet, agile will be empower earns to repeatedly re-plan their release to optimize its value throughout the development keeping the company competitive.

Mervin-

The Software Development Life Cycle (SDLC) is a description of the process of planning, creating, testing, and deploying an information system and an ambiguous, organized sequence of stages to develop a software product. It efficiently designs and develops a software product by providing a list of steps which includes Communication, Requirement Gathering, Feasibility Study, System Analysis, Software Design, Coding, Testing, Integration, Implementation, Operations & Maintenance and Disposition. Developers can select a strategy to develop the software with the aid of the software development paradigm. Three examples of the paradigms are the Waterfall model, Big Bang model and Spiral model.

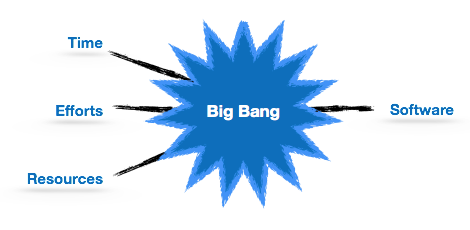
The Waterfall model, being the simplest model, says the phases of SDLC will function one after another in linear manner and assumes all processes are carried out smoothly and does not think that the past issues will arise in the next phase. Hence, when there are issues in the previous phase, the model will not work out smoothly.

Waterfall Model



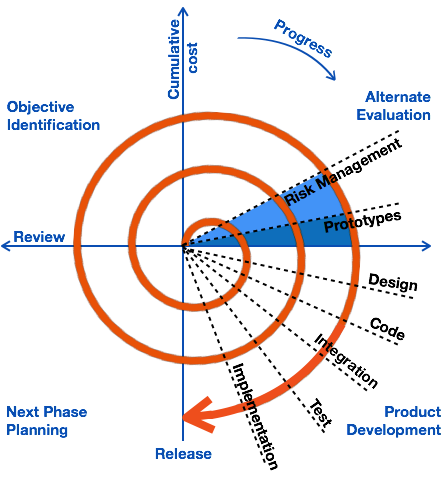
The Big Bang model is the simplest model in its form, does not need much planning, programming and funds and uses the concept of the big bang of universe. There are no processes required and the input requirements are arbitrary. This model helps in learning and experimenting but does not suit big software projects.

Big Bang Model



The Spiral model is a blend between iterative and one SDLC model. It analyses risks which may be overlooked by most other models.

Spiral Model



Resources:

<http://en.wikipedia.org/wiki/Waterfall_model>

<http://en.wikipedia.org/wiki/Unified_Process>

<http://en.wikipedia.org/wiki/Software_prototyping#Throwaway_prototyping>

<http://en.wikipedia.org/wiki/Systems_development_life_cycle#Phases>

<http://www.veracode.com/security/software-development-lifecycle>

<http://www.tutorialspoint.com/sdlc/sdlc_overview.htm>

<http://istqbexamcertification.com/what-are-the-software-development-life-cycle-sdlc-phases/>

<http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/>

<http://istqbexamcertification.com/what-is-v-model-advantages-disadvantages-and-when-to-use-it/>

<http://istqbexamcertification.com/what-is-spiral-model-advantages-disadvantages-and-when-to-use-it/>

<http://www.tutorialspoint.com/software_engineering/software_development_life_cycle.htm>

<http://en.wikipedia.org/wiki/Systems_development_life_cycle>