Basic Principles and Steps of Types of Software Models

Ramanpreet Kaur,
Department of Mathematics
Baba Farid group of Institutions,
Bathinda,India

Harvinder singh
Department of Mathematics
Baba Farid group of Institutions,
Bathinda,India

Satnam Singh
Department of Mathematics
Baba Farid group of Institutions,
Bathinda,India

Divya
Department of Mathematics
Baba Farid group of Institutions,
Bathinda,India

***Abstract*— Software development life cycle (SDLC) is a method by which quality software can be developed in the given time and according to the customer expectations. SDLC ensures quality product. Software Development Life Cycle Models are frameworks used to design, develop and test the software. They define a set of guidelines which are to be followed during the development. These models make sure that the software is designed systematically, according to the need of the customer and within the time schedule. Different types of software development life cycle models are waterfall, iterative, V-shaped, prototype and spiral model. Each of these models has its own benefits and drawbacks.**

***Keywords*— *SDLC, waterfall, iterative, prototype, spiral model*.**

I. INTRODUCTION

SDLC ensures quality product. All software development processes include various activities like requirements gathering and analysis, system analysis, system design, coding, testing, implementation. It is the choice of the developer or the team of developers to choose the SDLC model. Each SDLC model may have advantages and disadvantages in different situations. The challenge is to determine which model should be selected under certain circumstances.

II .TYPES OF MODELS

2.1 Waterfall Model:

It is also known as linear sequential life cycle model as it consists of sequence of phases. Once a development phase is completed, the development proceeds to the next phase in the sequence and there is no turning back to the previous phase. Thus it is not suitable for dynamic projects. Various phases in this model are Requirement gathering, system design, implementation, testing, deployment and maintenance.

Basic Principles

- Project is divided into sequential phases, with
- some overlap and splash back acceptable between phases. Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.
- Tight control is maintained over the life of the project via extensive written documentation, formal reviews, and approval/signoff by the user and information technology management occurring at the end of most phases before beginning the next phase.

The sequential phases in Waterfall model are:

1. Requirement Gathering and analysis: All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
2. .System Design: The requirement specifications from first phase are studied in this phase and the system design is prepared.This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
3. Implementation: With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing
4. Integration and Testing: All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
5. Deployment of system: Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
6. Maintenance: There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

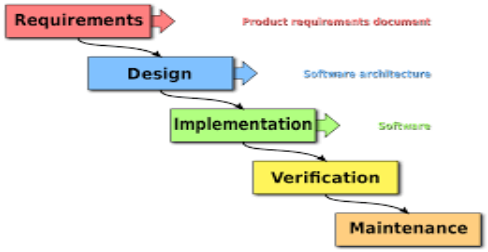


Fig.1 Waterfall Model

2.2 Iterative Model

In this model it is not required to start with the complete specifications. Instead, development starts by implementing a part which can then be reviewed and the next part can be planned according to the requirements. This process is repeated, giving new version of the software for each cycle of the model. In this model we can get user feedback. As this model proceeds step by step, it can be used when the project is big.

Basic Principles

- The problems with the Waterfall Model created a demand for a new method of developing systems which could provide faster results, require less up-front information and offer greater flexibility.
- Iterative model, the project is divided into small parts. This allows the development team to demonstrate results earlier on in the process and obtain valuable feedback from system users.

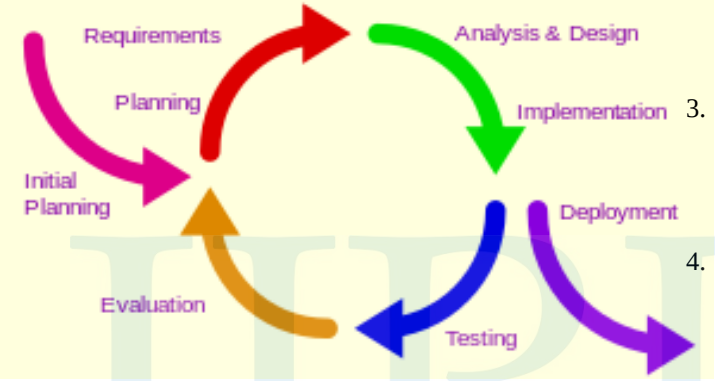


Fig.2. Iterative Model

2.3 Prototype Model

This model includes building a prototype before building the actual software. The prototype displays the functions of the product but may not actually hold the logic of the original software. It provides scope for understanding customer requirements at early stage and then proceeding accordingly. Also, errors can be detected much earlier. This model is used for applications which tend to have lot of user interactions.

Basic Principles

- Not a standalone, complete development methodology, but rather an approach to handling selected parts of a larger, more traditional development methodology.
- Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process.
- User is involved throughout the development process, which increases the likelihood of user acceptance of the final implementation.
- Small-scale mock-ups of the system are developed following an iterative modification process until the Prototype evolves to meet the users requirement .
- While most prototypes are developed with the expectation that they will be discarded, it is possible in some cases to evolve from prototype to working system.
- A basic understanding of the fundamental business problem is necessary to avoid solving the wrong problem.

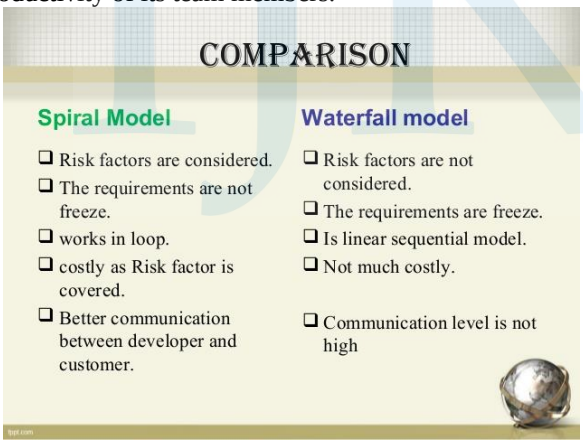
Following is a stepwise approach explained to design a software prototype.

1. Basic Requirement Identification This step involves understanding the very basics product requirements especially in terms of user interface. The more intricate details of the internal design and external aspects like performance and security can be ignored at this stage.
2. Developing the initial Prototype The initial Prototype is developed in this stage, where the very basic requirements are showcased and user interfaces are provided. These features may not exactly work in the same manner internally in the actual software developed. While, the workarounds are used to give the same look and feel to the customer in the prototype developed.
3. Review of the Prototype The prototype developed is then presented to the customer and the other important stakeholders in the project. The feedback is collected in an organized manner and used for further enhancements in the product under development.
4. Revise and Enhance the Prototype The feedback and the review comments are discussed during this stage and some negotiations happen with the customer based on factors like – time and budget constraints and technical feasibility of the actual implementation. The changes accepted are again incorporated in the new Prototype developed and the cycle repeats until the customer expectations are met. SDLC – Software Prototype Model SDLC 27 Prototypes can have horizontal or vertical dimensions. A Horizontal prototype displays the user interface for the product and gives a broader view of the entire system, without concentrating on internal functions. A Vertical prototype on the other side is a detailed elaboration of a specific function or a sub system in the product. The purpose of both horizontal and vertical prototype is different. Horizontal prototypes are used to get more information on the user interface level and the business requirements. It can even be presented in the sales demos to get business in the market. Vertical prototypes are technical in nature and are used to get details of the exact functioning of the sub systems. For example, database requirements, interaction and data processing loads in a given sub system.

Fig. 3 Prototyping Model

III.COMPARISION

A software development process is a structure imposed on the development of a software product. There are several models for such processes, each describing approaches to a variety of tasks or activities that take place during the process. It aims to be the standard that defines all the tasks required for developing and maintaining software. Software development teams, taking into account its goals and the scale of a particular project, and have a number of well-established software development models to choose from. Therefore, even though there are number of models each software Development Company adopts the best suited model, which facilitates the software development process and boosts the productivity of its team members.



IV. CONCLUSIONS

.Since the development team is familiar to the environment and it is feasible to specify all requirements of working environment. Iterative water fall model overcome the drawback of original waterfall model.It allow feedback to proceeding stage. Prototype model used to develop online systems for transaction processing. Since significantly reduce rework and lead to the creation of working model in lower capital cost. Spiral model is used for development of large, complicated and expensive projects like scientific Projects .Since spiral model approach enables the project term to address the highest risk at the lowest total cost.

REFERENCES

[1] Mumick I.S., Pirahesh H. “Implementation of Magic Sets in Starburst” Proc.of SIGMOD Conf., 1994.Authors

[2] Dayal, U., “Of Nests and Trees: A Unified Approach to Processing Queries that Contain Nested Subqueries, Aggregates and Quantifiers” Proc. VLDB Conf., 1987. 20 Murlaikrishna, “Improved Unnesting Algorithms for Join Aggregate SQL Queries” Proc. VLDB Conf., 1992

[3] Chaudhuri S., Krishnamurthy R., Potamianos S., Shim K. “Optimizing Queries with Materialized Views” Intl.Conference on Data Engineering, 1995

[4] Levy A., Mendelzon A., Sagiv Y. “Answering Queries Using Views” Proc. of PODS, 1995. 16 Yang H.Z., Larson P.A. “Query Transformations for PSJ Queries”, Proc. of VLDB, 1987.

[5] Kim W. “On Optimizing a SQL-like Nested Query” ACM TODS, Sep 1982. [18] Ganski,R., Wong H.K.T., “Optimization of Nested SQL Queries Revisited ” Proc. of SIGMOD Conf., 1987.

[6] O’Neil P., Quass D. “Improved Query Performance with Variant Indices”, To appear in Proc. of SIGMOD Conf., 1997.

[7] O’Neil P., Graefe G. “Multi-Table Joins through Bitmapped Join Indices” SIGMOD Record, Sep 1995.

[8] Harinarayan V., Rajaraman A., Ullman J.D. “ Implementing Data Cubes Efficiently” Proc. of SIGMOD Conf., 1996

[9] Gupta, A., I.S. Mumick, “Maintenance of Materialized Views: Problems, Techniques, and Applications.” Data Eng. Bulletin, Vol. 18, No. 2, June 1995. 9 Zhuge, Y., H. GarciaMolina, J. Hammer, J. Widom, “View Maintenance in a Warehousing Environment, Proc. Of SIGMOD Conf., 1995.

[10] Roussopoulos, N., et al., “The Maryland ADMS Project: Views R Us.” Data Eng. Bulletin, Vol. 18, No.2, June 1995

[11]]Swapanja Ingale “Comparative Study Of Software Development Model” International conf

[12] Sanjana Taya “Comparative Analysis of Software Development Life” ISSN:2229-4333(print),ISSN:0976- 8491(online),vol.2ISSUE 4,Oct-Dec2011