STUDY OF SOFTWARE DEVELOPMENT PROCESS AND DEVELOPMENT MODELS

Sachin Gupta *

Ankit Aggarwal **

ABSTRACT

This paper is concerned with explaining software development process and various development models. Software development process presents the task or activities used to produce a software product. A software development life cycle model is either a descriptive or prescriptive characterization of how software is or should be developed. There are tons of SDLC models. Each model contains specific activities to be performed to develop software. Many companies adopt their own model. The aim of this paper is to provide a brief study about software development life cycle, common activities to develop software and various development models.

Keywords: *Software Development, Software Development Life Cycle, Development models.*

* Software Engineering, Kurukshetra Univeristy, Kurukshetra, India.

** Computer Science & Engineering, Kurukshetra Univeristy, Kurukshetra, India.

I. INTRODUCTION

Software development life cycle models describe phases and the order in which those phases are to be executed so as to develop software. The common phases are Requirement gathering, Specification, Designing, Coding, Testing and Maintenance, etc. Each phase produces deliverables required by the next phase in the life cycle. Requirements and specifications are translated into design. Code is produced during implementation that is driven by the design. Testing verifies the deliverable of the implementation phase against requirements. There exists variety of development models each having their specified procedures and steps. The goal of this paper is to explore the description about the software development life cycle, common activities to develop software and various software development life cycle models. [1]

II. SOFTWARE DEVELOPMENT PROCESS/LIFE CYCLE

Software development life cycle is the entire life from beginning to completing the software development process. The development process is the task or activities used to produce a software product. A process can generate the other process. It is a set of activities, methods, practices, and transformations that people use to develop and maintain software and its associated products. [6]

III. COMMON ACTIVITIES/PHASES TO DEVELOP SOFTWARE

The classic software life cycle models usually include some version or subset of the following common activities:

- ***System Initiation/Planning:*** Where do systems come from? In most situations, new feasible systems replace or supplement existing information processing mechanisms whether they were previously automated, manual, or informal.
- ***Requirement Analysis and Specification:*** Identifies the problems a new software system is suppose to solve, its operational capabilities, its desired performance characteristics, and the resource infrastructure needed to support system operation and maintenance.
- ***Functional Specification:*** Identifies and potentially formalizes the objects of computation, their attributes and relationships, the operations that transform these objects, the constraints that restrict system behavior, and so forth.
- ***Partition and Selection (Build vs. Buy vs. Reuse):*** Given requirements and functional specifications, divide the system into manageable pieces that denote logical subsystems, then determine whether new, existing, or reusable software systems correspond to the needed pieces.

- ***Architectural Design:*** Defines the interconnection and resource interfaces between system subsystems, components, and modules in ways suitable for their detailed design and overall configuration management.
- ***Detailed Component Design:*** Defines the procedural methods through which the data resources within the modules of a component are transformed from required inputs into provided outputs.
- ***Coding/Implementation***: Codifies the designed components into operational source code implementations and validates their basic operation.
- ***Software Integration and Testing:*** Assures the overall integrity of the coded components, verifying the consistency and completeness of implemented modules, verifying the resource interfaces and interconnections against their specifications, and validating the performance of the system and subsystems against their requirements.
- ***Documentation Revision and System Delivery:*** Packaging and rationalizing recorded system development descriptions into systematic documents and user guides, all in a form suitable for dissemination and system support.
- ***Deployment and Installation:*** Providing directions for installing the delivered software into the local computing environment, configuring operating systems parameters and user access privileges, and running diagnostic test cases to assure the viability of basic system operation.
- ***Training and Use:*** Providing system users with instructional aids and guidance for understanding the system's capabilities and limits in order to effectively use the system.
- ***Software Maintenance:*** Sustaining the useful operation of a system in its host/target environment by providing requested functional enhancements, repairs, performance improvements, and conversions. [2]

IV. SOFTWARE DEVELOPMENT LIFE CYCLE MODELS

There are plenty of development models. Some of these are explained as below:

4.1 Linear Sequential Model: The Linear Sequential model is the earliest method of software development. It provides the theoretical basis for other development models. The linear sequential model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall), through the phases: Analysis, Requirements specification, Design, Coding, Testing, and Maintenance. [2]

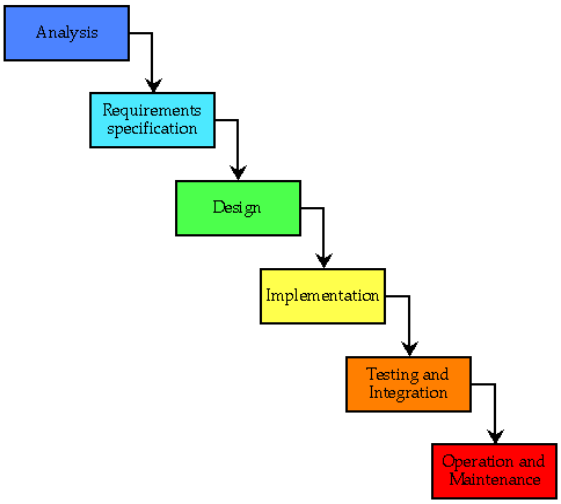

Figure 1: Linear Sequential Model [5]

4.2 Prototype model: The purpose of a prototype is to allow users of the software to evaluate developers' proposals for the design of the eventual product by actually trying them out, rather than having to interpret and evaluate the design based on descriptions .Prototyping can also be used by end users to describe and prove requirements that developers have not considered, and that can be a key factor in the commercial relationship between developers and their clients. The process of prototyping involves the following steps:

1. Identify basic requirements
2. Develop Initial Prototype
3. Review : The customers, including end-users, examine the prototype and provide feedback on additions or changes
4. Revise and Enhance the Prototype: Using the feedback both the specifications and the prototype can be improved. Negotiation about what is within the scope of the contract/product may be necessary. If changes are introduced then a repeat of steps 3 and 4 may be needed. [3]

4.3 Spiral Model: The spiral model is similar to the incremental model, with more emphases placed on risk analysis. The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations. The baseline spiral, starting in the planning phase, requirements is gathered and risk is assessed. Each subsequent spiral builds on the baseline spiral. Requirements are gathered during the planning phase. In the risk analysis phase, a process is undertaken to identify risk and alternate solutions. A prototype is produced at the end of the risk analysis phase. Software is produced in the engineering phase, along with testing at the end of the

phase. The evaluation phase allows the customer to evaluate the output of the project to date before the project continues to the next spiral. In the spiral model, the angular component represents progress, and the radius of the spiral represents cost. [1,4]

Figure2: Spiral Model [8]

This model of development combines the features of the prototyping and the waterfall model. The spiral model is intended for large, expensive and complicated projects. [8]

4.4 Incremental Model: The incremental model is an intuitive approach to the waterfall model. Multiple development cycles take place here, making the life cycle a "multi-waterfall" cycle. Cycles are divided up into smaller, more easily managed iterations. Each iteration passes

through the requirements, design, implementation and testing phases. A working version of software is produced during the first iteration, so you have working software early on during the software life cycle. Subsequent iterations build on the initial software produced during the first iteration. [1]

4.5 Rapid Application Development Model (RAD): RAD is a development lifecycle designed to give much faster development and higher-quality results than those achieved with the traditional lifecycle. RAD model has the following phases**:** Business Modeling, Data Modeling, Process Modeling, Application Generation, Testing and Turn over. [7]

4.6 Agile Model: Agile software development is a style of software development that emphasizes customer satisfaction through continuous delivery of functional software". The Process of Agile Software Development involves the following:

1. Starts with a kick-off meeting
2. The known requirements are understood and prioritized. The development is plan is drawn accordingly.
3. Relative complexity of each requirement is estimated
4. Sufficient design using simple diagrams is done
5. Test Driven Development (TDD) approach may be used. TDD emphases on "writing test first and then writing code to pass the test". It can help in avoiding over-coding.
6. Development is done, sometimes in pairs, with lot of team interaction. Ownership of code is shared when pair programming is done.
7. The code is tested more frequently. Sometime a dedicated "Continuous Integration" Server/Software may be used to ease the integration testing of the code.
8. Depending on the feedback received, the code is refractor. Refactoring does not impact the external behavior of the application but the internal structure may be changed to provide better design, maintainability. Some ways of refactoring may be add interface, use super class, move the class etc.

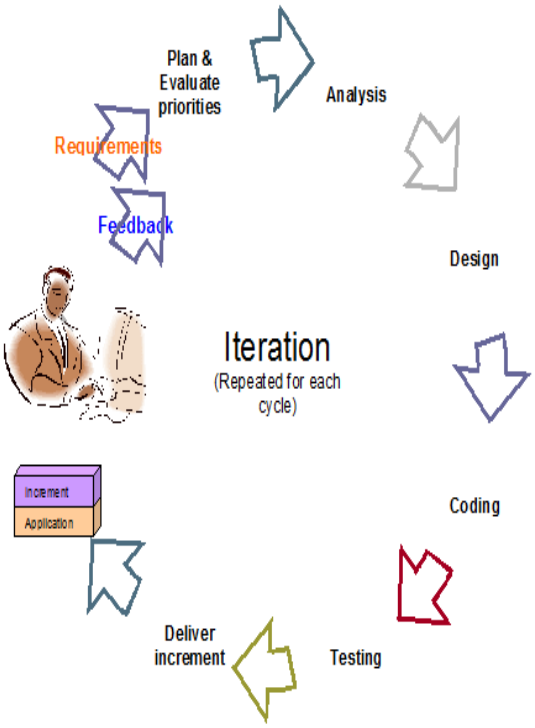

Figure3: Agile SDLC [9]

V. CONCLUSION

A study has been given which helps one to understand about software development process, common activities to develop software and various development models. Tons of development models exist. This paper explained six different models out of those. First one is Waterfall model which provides base for other development models. Then its enhanced models are explained. Prototyping model, Spiral model, Incremental model, Rapid Application Development model and finally, Agile development model.

VI. FUTURE SCOPE

The work presented in this paper is survey work of existed software development models. Nothing in the world which can be said complete. There are a lot of limitations in existing models. In future, A lot of work can be done to produce a new software development model.

VII. REFERENCES

1. Raymond Lewallen, “Software Development Life Cycle”, 2005
2. Dr. Winston, W. Royce, www.informatik.uni-bremen.de. Retrieved 27 Oct 2008.
3. UCertify, “Software Prototyping”, http://www.ucertify.com/article/software-prototyping.html, 2010.
4. Kichan Park, Murad Ali and Françoise Chevalier, “A spiral process model of technological innovation in a developing country: The case of Samsung”, African Journal of Business Management Vol. 5(13), pp. 5162-5178, 4 July, 2011
5. http://www.csse.monash.edu.au/~jonmc/CS E2305/Topics/07.13.SWEng1/html/text.html
6. Laurie Honour Werth, “Lecture Notes on Software Process Improvement”, document number CMU/SEI-93-EM-8, copyright by Carnegie Mellon University, “Introduction to Software Process Improvement”, 1993.
7. James Martin, “Rapid Application Development Model (RAD)”, Software Engineering, August, 2009.
8. Boehm B, "A Spiral Model of Software Development and Enhancement", ACM SIGSOFT Software Engineering Notes", "ACM", 11(4):14-24, August 1986
9. Prerana Patil, “Quick Introduction to Agile Software Development”, www.indicthreads.com, June 16, 2008