ISSN 2231-3842 (Print) ISSN 2277-8691 (Online)

DIAGNOSTIC STUDY OF AN ASSORTMENT OF PROCESS MODELS IN SOFTWARE ENGINEERING

Mrs. Anjali Sharma
Assistant Professor
School of Computer Science and Applications, IIMT University, (India)
E-mail: Anjali.shail@gmail.com

Ms. Priyanshi, Mr Lavesh, Mr Aniket, Mr. Yash Chauhan, Mr. Lareeb
Students
School of Computer Science and Applications, IIMT University, (India)

Abstract

In Software Engineering, software process models can be measured at two different stages actions: one at the begin stage actions which is related to gather the information which based on the analysis, development of the required aspects, and last is maintenance of the concern software, in other second stage action is related with the dentition of the software, functioning about the software, measurement, and upgrading according demands software process itself. This paper nearby the proportional study of various process models in software engineering which based on various parameters and also planned different aspects for software process models in the throughout the world of software development.

Keywords: Software Engineering, Software Process Model, Software Crisis.

1. Introduction

Software development starts to and goes by a sequence of path that report for the beginning, initial development, and lively operation, remains, and exit from one production to another. This paper show the variety of software process model and also examine a various methods for describing model with how we can develop the software systems. At the initial stage its starts with background idea and labeling of well-known software life cycle process models (Van der Aalst, 1999).

This paper is divided into four divisions.

1. Details of all Software Process Models.
2. Discuss the Comparison chart of various model based on usage in now a day in organizations.
3. Comparison Table of Various Process Model on the basis of Parameters
4. Finally, the conclusion gives a summary of the findings of this research

2. Software Process Model

A software process model is the way through we can take a conceptual representation of processes which are using to develop the software according user demands (Kindler, 2006). It simply follows the software development life cycle method which includes various phases like

Analysis, Design, Implementation or coding, Testing, and Maintenances.

2.1 Waterfall Model

Waterfall model is the oldest software process model. This model was introduced by Dr. Winston W Royce in 1970.it is used to create systems with linear and sequential approach, so easy to use. This model is named waterfall as it is developed systematically from one phase to another in a downward fashion like a waterfall. The model has different phases where output of every phase act as input of next phase, and there is no overlapping between two phases. This type of models is used when client is crystal clear about his requirements and his expectations out of the project.

Advantages of waterfall model

- The model is simple, easy to use and understandable.
- A systematic order is followed each phase is completed before starting to next phase.
- Quality assurance test are performed at each step to validate the quality.

Disadvantages of waterfall model

- The waterfall model is not appropriate for complex projects.
- The modifications can be only performed with ongoing phase.
- Documentation consumes a lot of time.

2.2 Prototyping Model

It is one of the most popular models used in software development life cycle. The prototype is the preliminary version of the final project. The basic idea behind the model is to build a prototype model before proceeding to design and coding, based on currently known requirement. After using the prototype user can understand better about his actual requirement and expectations about the project and can describe a clear idea to the developer. The model is well suited for large complicated system, where there is no previous experience of the system.

Advantages

- User’s involvement reduces the risk of crisis, and leads to improved quality of the final project.
- A single prototype can be reused as a resource based upon the user’s requirements.
- Errors can be detected at early stage and can be rectified timely.

Disadvantages

- Requirement analysis is insufficient as the developer relies on prototype.
- If the prototype implemented is incorrect then a lot of time is wasted
- Poor documentation because the requirements of the user are continuously changing.

2.3 Incremental Model

Incremental process model is also known as the Successive version model. In this model requirements divided into multiple modules of software development cycle. First, a simple working system implementing only a few basic features is built and then that is delivered to the customer. Each of the iterations in the Incremental model passes through four different phases. They are the requirements phase, design phase, coding phase, and testing phase. When a project has a lengthy development schedule then incremental model is used (Preuner, Conrad & Schrefl, 2001). We are achieving 100% objective of the software with this model. There are two types of incremental model i.e., **Staged Delivery Model and Parallel Development Model.**

Advantages

- It is easy to test and debug.
- This module will charge less cost to change scope and requirements of the whole system.
- Different module can be completed at different time.
- It is more flexible in nature.

Disadvantages

- The main disadvantage of this system is that it needs a clear and complete definition of the whole system.
- The total system cost is greater than the waterfall model.
- It consumes a lot of time.

2.4 Spiral Model

Spiral model is one of the most important Software Development Life Cycle models, because it provides support for **Risk Handling.** It is similar to the incremental development for a system, with more emphasis placed on risk analysis. This model does not follow the sequential stages. The spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, or prototyping (Rozinat & Van der Aalst, 2008). The spiral model is a combination of sequential and prototype models.

Advantages of spiral model

- We can easily change requirements at later phases.
- Strong approval and documentation control.
- There are higher chances of customer satisfaction because at each phase customers gets to see.

Disadvantages of spiral model

- Because of multiple phases it is **not suitable for a simpler and smaller** project.

- It is quite expensive.
- Highly skilled technicians are required to perform risk analyses.

2.5 RAD Model

RAD model is defined as Rapid Application Development model. It is a software development process based on prototyping without any specific planning. It uses an element-based construction approach to emphasizes a concise development cycle. If the project scope is a constraint, the RAD process enables (Aleem, Fernando Capretz & Ahmed 2016). It emerged as developers discovered that the conventional waterfall development approach was not particularly successful. There are five phases in rapid application development (RAD) model which are following:

1. Business modeling
2. Data modeling
3. Process modeling
4. Application generation
5. Testing and turnover

Advantages of RAD Model

- It increases reusability of components.
- RAD ensures high quality of the product.
- It uses less developer so it is cost effective.

Disadvantages of RAD model

- It only suits small and medium size project so it is not suitable for all projects.
- It is not an appropriate solution when the technological risk is large.
- If the team is not sufficiently motivated then this method cannot be success.

2.6 Agile Model

Agile development is may be one kind of iterative and Incremental based development process model in software engineering. In this model Software is developed incrementally with rapid cycles. This provides small-small increments that release time to time according demands of users, with each and every release increments building on previous functionality. In each release small increment is systematically tested to guarantee software quality is maintained or not. Today's the most well-known agile development life cycle model is Extreme Programming (XP) and another method are scrum, crystal Methodology, Mobile-D.

3. Comparison chart of various model based on usage

There are various software process models for software development. Fig 1 shows various process models which we described in this paper in previous section. It Describe which process model how much used by organizations. Also, some of model are highly efficient but not used much and some is much complex but gives good Efficiency and used highly in the world of software development. This paper Describes which process model satisfied which kind of parameter.

Figure 1: Usage Percentage of Various Models

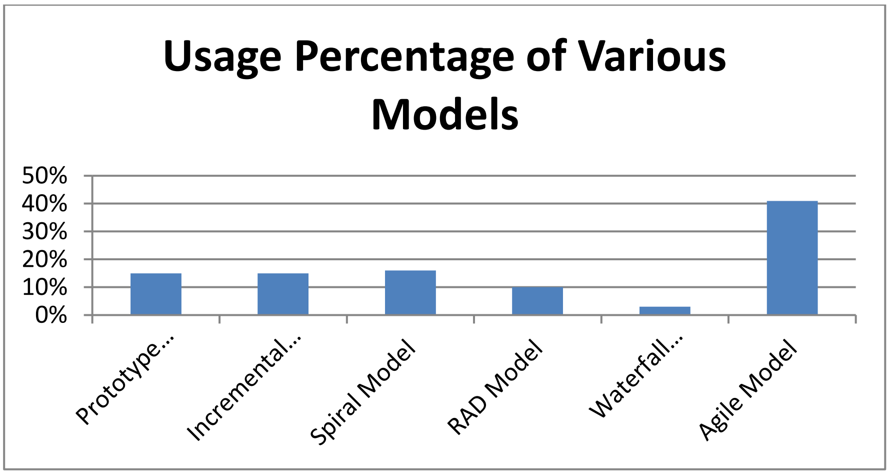

Table 1: Comparison Table of Various Process Model on the basis of Parameters

Parameter	Waterfall Model	Incremental Model	Prototype Model	RAD Model	Spiral Model	Agile Model
Requirement and Specification	Initial time	Initial time	initial level	initial	initial	Change accordingly
Changing Possibilities	low	high	medium	no	high	high
Idea about Project	low	low	high	low	medium	high
Risk Chances	Initial level	no	no	no	yes	yes
Usability	low	medium	high	medium	medium	use nowadays
Cost	low	low	high	very high	expensive	expensive

Understandability	simple	medium	medium	medium	hard	complex

4. Conclusion

This paper discussed what the role of software process model is and how they perform according user requirement. In this paper we described the various process models, also evaluate them with different constraint and underline the factors according choosing it todays in industries through usages which show by chart. on the other hand, the existing model still can be recovering and adapted with less cost, time with highly efficient. After this study some aspects will show from the point of view by Developer that should find out 1. Study market analysis that why Agile Model is accepted now a days. 2. How we can advance the effectiveness of suggested models.

References

Van der Aalst, W. (1999). Formalization and Verification of Event-driven Process Chains. *Information and Software Technology*, *41*, 639–650.

Kindler, E. (2006). On the semantics of EPCs: Resolving the vicious circle. *Data & Knowledge Engineering*, *56*, 23–40.

Preuner, G., Conrad, S., Schrefl, M. (2001). View Integration of Behavior in Object-Oriented Databases. *Data & Knowledge Engineering 36*(2), 153–183.

Rozinat, A., Van der Aalst, W. (2008). Conformance checking of processes based on monitoring real behavior. *Information Systems 33*(1), 64–95.

Aleem S, Fernando Capretz L, Ahmed F (2016). A Digital Game Maturity Model (DGMM), *Entertainment Computing*, *17*, 55-73.