A Comparison Between Two Software Engineering Processes, RUP And Waterfall Models

Mina zaminkar[a], Mohammad R. Reshadinezhad[b]
a Graduate student, , Department of Computer Science Research Branch, Islamic Azad University, Yazd, Iran
b Assistant professor, Department of Computer engineering, University of Isfahan, Isfahan, Isfahan , Iran

Abstract

Two of the leading software engineering processes are Rational Unified Process (RUP) and waterfall models. RUP is a unified model planning form for large business applications that provides a language for describing method content and processes. The waterfall model is a classical model of software engineering. This model is used in governmental projects as well as many major companies associated with software engineering. The main concern in this research is to represent the mentioned models of software development and make comparison between them to show the features and defects of each model.

1. Introduction

Computer has become indispensible in today's life in fields such as industry, commerce, education and etcetera. Location of resources, team structure, corporate culture and even technology used, can all play a key factor in determining which development practices will work in an organization. Now days, companies become more dependent on computer in their works because of computer technology. Computer is considered as a time saving mechanism and its progress helps in executing long, complex, repeated processes at a high speed and short time. Noticeably, the number of companies that produce software programs for the purpose of facilitating works of offices, administrations, banks, etc., has increased recently which results in the difficulty of enumerating such companies. During the last four decades, software has been developed from a tool used for analyzing information or solving a problem to a product in itself. However, the early programming stages have created a number of problems turning software an obstacle to software development particularly those relying on computers. Software consists of instructions and programs that contain a collection that has been established to be a part of software engineering procedures [1]. Moreover, the aim of software engineering is to create a suitable work to construct programs of high quality. Figure 1 shows the software engineering conception that is used in today's life.
While many companies are actively seeking to use agile practices, such as extreme programing (XP) [1] or Scrum [2] to help streamline production with fast, effective development practices that can give their customers what they want in the shortest time possible, elements of some of the traditional software development methods such as the Rational Unified Process (RUP) [3] or Waterfall are often required to bridge the gaps that some of these new practices have. In the end, most of organizations must adopt a blended approach that bests fits their software project, business culture, and development environment.

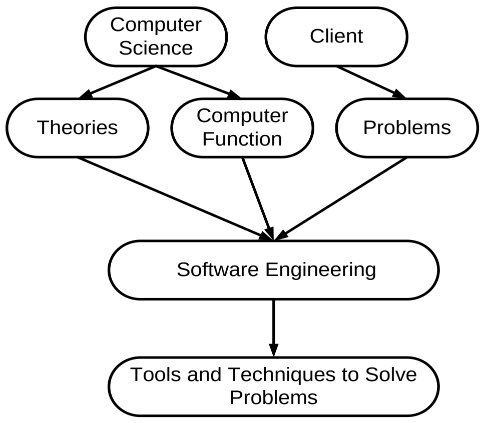

Figure 1. Software engineering conception

2. The RUP and its core concepts

During the 1990's Rational Company aimed to unify the various, then existing methodologies for object oriented analysis and design into a "Unified Method". This project was configured in two steps: in the first step, designing and publishing the "Unified Modeling Language" (UML) as a notation for any kind of software modeling results [4], in second step, complementing UML by a paradigmatic, idealized process description the RUP, which is well documented by [4,5,6], and by further presentations of its authors and other people.

By the RUP approach, its authors claim to "enhance team productivity" and to "give project managers control over schedules and deliverables". Furthermore, the RUP is advertised as being "iterative and incremental, use case-driven and architecture-centric". In general, it is argued that RUP suffers from its oversize and it's over sophistication. A primary goal of the RUP is to support software engineers working with UML. In [7] the author has called UML a "modern dinosaur". The RUP decomposes the software life cycle into phases which may be subject to several iterations, consisting of activities which are interwoven with so called disciplines and which are terminated by milestones. Looking at the central graphical illustration of the RUP (Fig. 2.) observe that the indicated five disciplines have their peak intensity in corresponding phases: "Requirements" in the inception and elaboration phase, "Analysis and design" in the elaboration phase etc.

The Rational Unified Process is a Software Engineering Process (Figure 2). It provides a disciplined approach to assigning tasks and responsibilities within a development organization. Its goal is to ensure the production of high-quality software that meets the needs of its end-users, within a predictable schedule and budget. RUP is “use-case driven, architecture-centric, and incremental and iterative”. Software projects that use RUP divide the project time-line into four consecutive phases.

Inception, where the project’s scope, estimated costs, risks, business case, environment and architecture are identified.

Elaboration, where requirements are specified in detail, architecture is validated, the project environment is further defined and the project team is configured.

Construction, where the software is built, tested and supporting documentation is produced.

Transition, where the software is system tested, user tested, reworked and deployed.

Each phase is concluded with a well-defined milestone a point in time at which certain critical decisions must be made and therefore key goals must have been achieved. Iterations occur in each phase. Activities in iterations are focused on one of the four activities: gathering requirements, analyzing, designing, implementing, and testing. Each of these activities place a more or less important role as the project moves from phase to phase. RUP also defines the roles and activities of team members in-depth and relies at each stage on the production of visual models, which are rich graphical representations of software systems, and specific use cases rather than the large amounts of documentation required for each stage of Waterfall. All team members have access to the same large knowledge base of guidelines, templates, tools, and other items to ensure that they share the same language and perspective on the project [8].

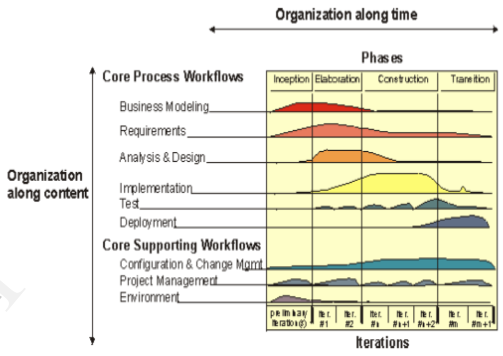

Figure 2. The RUP structure of each workflow during the project's phases

As it is seen in Figure 2, there are nine core process workflows in RUP, which represent a partitioning of all workers and activities into logical groups.

3. The waterfall model

The waterfall model is a popular version of the systems development life cycle model for software engineering. Often considered the classic approach to the systems development life cycle, the waterfall model describes a development method that is linear and sequential. Waterfall development has distinct goals for each phase of development. Imagine a waterfall on the cliff of a steep mountain. Once the water has flowed over the edge of the cliff and has begun its journey down the side of the mountain, it cannot turn back. It is the same with waterfall development. Once a phase of development is completed, the development proceeds to the next phase and there is no turning back. As this model emphasizes planning in early stages, it ensures design flaws before they develop. In addition, its intensive document and planning make it work well for projects in which quality control is a major concern. The pure waterfall lifecycle consists of several non-overlapping stages. The configuration of this model is shown in figures 3 and 4 as it is introduced in [8] and [9] respectively. The model begins with establishing system requirements and software requirements and continues with architectural design, detailed design, coding, testing, and maintenance. The waterfall model serves as a baseline for many other lifecycle models.

The details of each step in waterfall model are as follows:

1. **System requirements:** The components of building the system, including software tools, hardware requirements, and other necessary components are established in this stage.

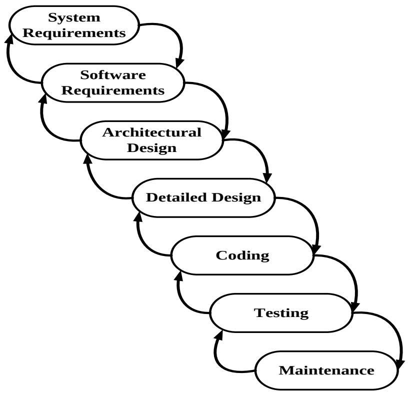

Figure 3. Waterfall model [8]

2. **Software requirements:** The expectation for software functionality is established in here and also identifies which system requirements the software affects. The requirements analysis consists of specifying interaction needed with other applications and databases.
3. **Architecture design:** In this stage it is looked at to see if the framework of the system is meet the specific requirements. In here the external interfaces, devices and tools used in the project can be evaluated by the designer.
4. **Detailed design:** In this step specification for how each component is implemented is produced and also the software components are examined.
5. **Coding:** The detailed design specification is implemented in here.
6. **Testing:** Examines to see whether the software meets the required specifications or not and if there is any mistakes in the written code.
7. **Maintenance:** After the software is released, the problems have to be addressed and requests have to be enhanced. It is the process of modifying a software solution after delivery to enhance the output [10].

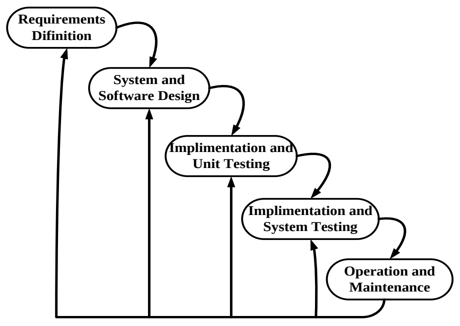
Requirements Difinition
System and Software Design
Implimentation and Unit Testing
Implimentation and System Testing
Operation and Maintenance

Figure 4. Waterfall model [9]

Another waterfall model is introduced by [11]. In there model each phase starts with a design goal and ends with the client (who may be internal) reviewing the progress thus far. This model is called modified waterfall model and is depicted in Figure 5.

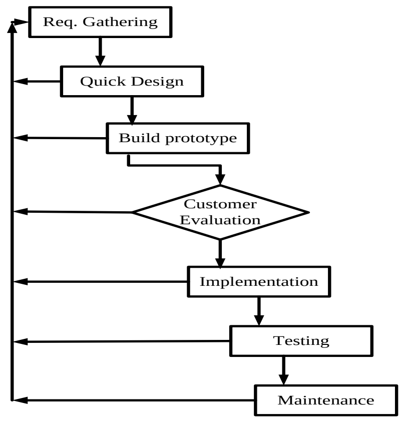

Figure 5. Modified waterfall model [11]

They stated that, the modified waterfall model has some strength such as allow changing the client requirement at any stage second; User can view the system model before development of the actual system and just after the design phase, so that if there is any change in the model then it can be handled in early phase of development. The last strength is that, User can review the ongoing system after each stage, and if there are any changes then it can be incorporated at early stage as soon as it is observed [11].

4. Comparison of RUP and Waterfall models

Here the advantages and dis advantages of each method will be illustrated. First the advantages and dis advantages of RUP is demonstrated.

Advantages of RUP are as follow:

1. This model is a complete methodology in itself with an emphasis on accurate documentation.

2. It is proactively able to resolve the project risks associated with the client's evolving requirements requiring careful change request management.
3. Less time is required for integration as the process of integration goes on throughout the software development life cycle.
4. The development time required is less due to using of components over and over.
5. There is tutorial and online training available for users to use this process.

Disadvantages of RUP are as follow:

1. The team members need to be expert in their field to develop the software under this methodology.
2. The development process is too complex and disorganized.
3. On cutting edge projects which utilise new technology, the reuse of components will not be possible. Hence the time saving one could have made will be impossible to full fill.
4. Integration throughout the process of software development, in theory sounds a good thing. But on particularly big projects with multiple development streams it will only add to the confusion and cause more issues during the stages of testing.

Advantages of waterfall model are as follow:

1. It allows for departmentalization and managerial control.
2. Simple and easy to understand and use.
3. Easy to manage due to the rigidity of the model, each phase has specific deliverables and a review process.
4. Phases are processed and completed one at a time.
5. Works well for smaller projects where requirements are very well understood.
6. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process like a car in a car-wash, and theoretically, be delivered on time.
7. Significant administrative overhead, costly for small teams and projects

Disadvantages of waterfall model are as follow:

1. It does not allow for much reflection or revision.
2. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
3. No working software is produced until late during the life cycle.
4. High amounts of risk and uncertainty.
5. Not a good model for complex and object-oriented projects.
6. Poor model for long and ongoing projects.
7. Not suitable for the projects where requirements are at a moderate to high risk of changing.

5. Conclusion

In this paper the two models of RUP and waterfall for software engineering was compared. Each model was introduced separately and advantages and disadvantages of each model were illustrated completely. As future work, other software development life cycle models such as spiral and incremental are to be considered and also simulated, allowing project managers to select the best software development methodology.

6. References

[1] Ian Sommerville, *Software Engineering*, Addison Wesley, 9th edition, 2010.
[2] Rising L., N. S. Janoff, *The Scrum Software Development Process for Small Teams*, IEEE Software, July/August 2010.
[3] Philippe Kruchten, *The Rational Unified Process, An Introduction***,** Addison Wesley, 3rd Edition, 2003.
[4] Unified Modeling Language (UML) 1.5 Documentation. OMG documentformal Rational Software Corp., Santa Clara, CA 2003. http://www.rational.com/uml/resources/ documentation as of 18th Aug. 2003.
[5] I. Jacobson, G. Booch, J. Rumbaugh: *The Unified Software Development Process*. Addison-Wesley 1999.
[6] Rational Unified Process- Product Overview. http://www.rational.com/products/rup as of 18th Aug. 2003
[7] K.D. Schewe, UML A Modern Dinosaur? A Critical Analysis of the Unified Modelling Language. In: H. Jakkola et al. (eds.) Information Modelling and Knowledge Bases XII. Proc. 10th European-Japanese Conference , pp. 185-202, Vol 67, IOS Press 2001.
[8] National Instruments Corporation, "Lifecycle Models", 2006 , http://zone.ni.com.
[9] CTG. MFA – 003, *A Survey of System Development Process Models*, Models for Action Project: Developing Practical Approaches to Electronic Records Management and Preservation, Center for Technology in Government University at Albany / Suny,1998 .
[10] Andrew Stellman, Jennifer Greene, *Applied Software Project Management*, O'Reilly Media, 2005.
[11] U. A. Patel, N. K. Jain, *New Idea In Waterfall Model For Real Time Software Development*, IJERT, Vol. 2 Issue 4, April, 2013.

Mohammad R. Reshadinezhad He was born in Isfahan, Iran, in 1959.He received his B.S. and M.S. degree from the Electrical Engineering Department of University of Wisconsin, Milwaukee, USA in 1982 and 1985,respectively. He has been in position of lecturer as faculty of computer engineering in University of Isfahan since 1991. He also received the PhD Degree in computer architecture from Shahid Beheshti University, Tehran, Iran, in 2012. He is currently Assistant Professor in Faculty of computer Engineering of Isfahan University. His research interests are digital arithmetic, Nanotechnology concerning CNFET, VLSI implementation, logic circuits design, Cryptography and software engineering.

Mina Zaminkar She received her B.S. in computer engineering (software) from university of Dolat Abad, Iran in 2010 and currently working on M.S. degree in computer engineering at the department of computer and research branch, Islamic Azad University, Yazd, Iran. Her research interests mainly focus on computer software engineering, data mining, data bases and biometric. She is currently engaged in research involving railway interlocking and Quality Management.