International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 05 | May-2016 www.irjet.net p-ISSN: 2395-0072

AN EFFICIENT SOFTWARE DEVELOPMENT SYSTEM USING XP, RUP AND DSDM

[*1] Mrs. Barathi. S., [*2] Mrs. Shoba S. A.,

[*1] M.Phil Research Scholar, PG & Research Department of Computer Science & Information Technology Arcot Sri Mahalakshmi Women's College, Vellore, Tamil Nadu, India.

[*2] Assistant Professor, HOD of PG & Research Department of Computer Science & Information Technology Arcot Sri Mahalakshmi Women's College, Vellore, Tamil Nadu, India.

Abstract- ***A number of Information System development methods is in practice. When any organization is going for Information System development they will be looking for the IS development method that is most complete and/or significant and to fulfill their requirements and give them a complete solution. The purpose of this paper is to compare the three most practiced methods and analyze them on different aspects and to find which of them is the most complete and or significant. New Technologies and concepts are taking place very rapidly and that is why as the time progresses, the methods which were developed earlier seemed to be incomplete and not fulfilling the requirement of the up–to-date technologies and therefore new methods are evolved. Most of them are based on the old method with some amendments or additions, some might be new concepts. So in this paper three of the most popular Information System development methods (XP, RUP, and DSDM) which are in practice nowadays in the market are discussed and compared with respect to different aspects, like process of the methods and practices.***

Key Words: *IS development method,* Project *Management Methodologies (PMM), DSDM, XP, RUP Process*

I. INTRODUCTION

Project Management is a process. Tasks and activities are planned, organized, assigned resources and executed within a given budget and period. There are several software management models that are used in the process of managing a software project. Since there are many techniques and models, some of them are selected and analyzed. This is to help managers in making a decisions to choose a tool to use depending on their needs. The main reason for looking at the software models is the limited information on the usage of the models.

The purpose of this thesis is to research on the most used software project management models. The models that are reviewed are the currently being used. General information on each of the selected models will be discussed with emphasis on their key points, advantages and disadvantages. There are several software development models that will be highlighted. The models that will be discussed to software project management include; Rational Unified Process (RUP), Dynamic Systems Development Method (DSDM) and extreme Programming (XP).

Software Project

A project is a well defined task, which a collection of several operations is done in order to achieve a goal every project may have a unique and distinct goal. Project is not a routine activity or day to day operations. Project comes with a start time and end time. Project ends when its goal is achieved and hence it is a temporary phase in the lifetime of an organization. Project needs adequate resources in terms of time, manpower, finance, and material and knowledge bank.

Project Management

Project management is the application of knowledge, skills, tools and techniques to project activities to meet the project requirements. The effectiveness of project management is critical in assuring the success of any substantial activity. Areas of responsibility for the person handling the project include planning, control and implementation. A project should be initiated with a feasibility study, where a clear definition of the goals and ultimate benefits need to be determined.

Software Project Management

Software project management encompasses the knowledge, techniques, and tools necessary to manage the development of software products. This module discusses material that managers need to create a plan for software development, using effective estimation of size and effort and to execute that plan with attention to productivity and quality. Within this context, topics such as risk management, alternative lifecycle models, development team organization and management of

technical people are also discussed. The software project management should be part of software engineering programs because the technology of developing software is so closely tied to the techniques of management.

Fig 1.1 Software Project Management

II. Related work

The development of large scale software is generally organized in the form of a project. Intensive management facilitates such development. Therefore, the key to the success of a project is to develop technology which strongly supports project management. This paper describes the requirements for a system which can support project management of software development. And the relationship between activities and resources must be established. This paper proposes a process model which meets the requirements. This paper also uses an example to show the behaviour of the model and assesses the model's usefulness.

Process models are the bedrock on which all software development projects are based. Since the first process model was defined in the late 1950s, more contemporary processes have evolved to deal with even more complex projects in even more dynamic problem domains. Although there are many such process models now available for software engineers to follow, they can be classified according to one of five basic types. They differ in the level to which the process might be applied and also in the additional guidelines and philosophies they define. By combining this technique with a Functionality Cost/Benefit graph, it helps to identify the key decision points in the software development process with respect to a software system's functionality.

Across all industrial sectors, project management has become an essential element in the successful delivery of projects. Regardless of the industrial sector or size of project, Project Management Methodologies (PMM) can be applied to improve the probability of meeting the project goals. In an earlier published work, we had classified PMM in five distinct but interdependent levels. Each of the PMM across the three sectors will be compared and discussed against a list of elements to elicit a common set of requirements.

Information systems are increasingly becoming regarded as crucial to an organisation's success. A development methodology for an information system is a framework to organize, program and supervise the process of developing an information system. There many are different methodologies for information systems development. Obviously, no methodology can claim that it can be applied to any organisation. Therefore, organisations should have an evaluation framework for selecting an appropriate and efficient methodology. A comparison framework is proposed and methodologies are compared by this framework. The value of this framework is that, with use of it, organisations can evaluate their development methodology with respect to the key features of it before implementing any methodology as well as expending extra costs.

Due to the spreading of SMS services and appearing of new business models, value added SMS services have been introduced. According to the research results about wide distribution of security incidents on ICT systems worldwide, in spite of known security solutions, there is a necessity for organizational approach to implement security. This paper presents research and development efforts in building process model Secure RUP for security requirements engineering conformed to RUP framework. The model consists of processes, artifacts, activities and according roles for successful elicitation, analysis and specification of recognized security requirements and is validated on presented case study.

III. PREVIOUS IMPLEMENTATIONS

3.1 Rational Unified Process (RUP)

The IBM Rational Unified Process (RUP) is a prescriptive, well defined system development process, often used to develop systems based on object and/or component based technologies. 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 easy to adapt because it is documented using Unified Modelling Language (UML). The UML is an industry standard language that allows us to clearly communicate requirements, architectures and designs. The UML was originally created by Rational Software, and is now maintained by the standards organization Object Management Group (OMG).

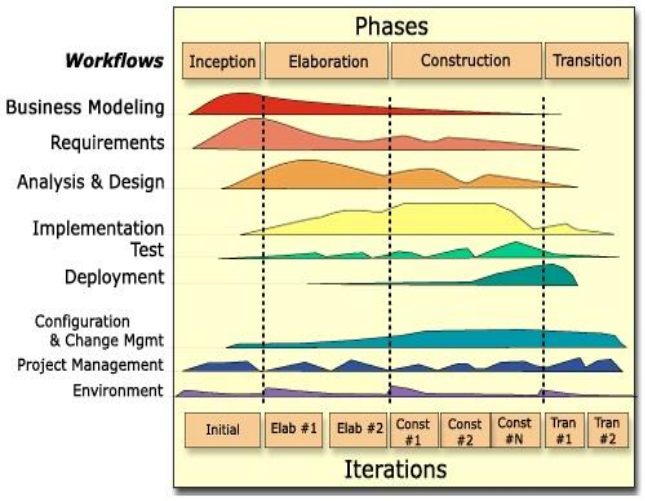

Fig 1.2 Overview of the phases and disciplines of the Rational Unified Process

Iterations

Each phase in the Rational Unified Process can be further broken down into iterations. Iteration is a complete development loop resulting in a release (internal or external) of an executable product, a subset of the final product under development, which grows incrementally from iteration to iteration to become the final system.

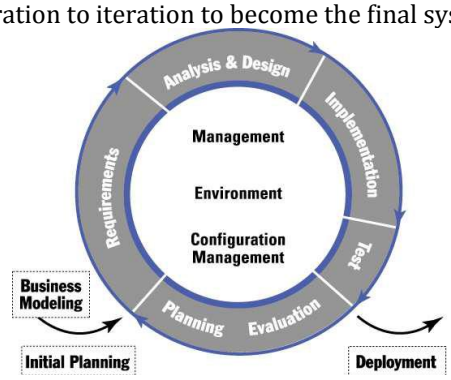

Fig1.3: Iterative approach of RUP

3.2 Dynamic System Development Method (DSDM)

DSDM is skeletal structure for RAD, it originated in 1994 and maintained by an organization called DSDM Consortium, unlike other methods DSDM keep the time and resources fixed and functionality can be variable. Simply system will be delivering in a fixed time. It focuses upon early delivery of real benefits to the business. DSDM Atern is an agile framework for management and delivery of IT- and non-IT-related projects. It can be regarded as a formalized version of RAD (Rapid Application Development).

DSDM is an iterative and incremental methodology that combines the project and product management life cycle into one best process. It was developed to fill in some of the gaps in the RAD method by providing a framework which takes into account the entire development cycle. It is a proven framework for agile project management and delivery, helping to deliver results quickly and effectively as it concentrates on strategic goals and incremental delivery of real business benefits while keeping control of time, cost, risk and quality.

IV. SYSTEM IMPLEMETNATION

Software models are very important in software project management. There are several software management models that are used in the process of managing a software project. Since there are many of the techniques and models, in this paper selects and analyzes some of them. Project managers should select a project model that they are familiar with and is most suitable for the project. The main reason for looking at the software models is the limited information on the usage of the models. The size of the project and the duration for the development of a project should be considered in software development model.

In this paper three software project management models to analyze and compare, which are Rational Unified Process (RUP), Dynamic System Development Method (DSDM) and Extreme Programming (XP). A Hybrid Software Development approach for Small to Medium Scale Projects, which has used the models RUP, DSDM and XP. To comparing these models user represent characteristics, strengths, and weaknesses of RUP, DSDM and XP process models, and a new hybrid software development model, which integrates the strengths of RUP, DSDM and XP while suppressing their weaknesses.

RUP

The life cycle of RUP is divided in to four phases: Inception, Elaboration, Construction, and Transition. And each phase is then further divided into iterations. Each iteration have a purpose to produces an integral part of the software.

Each phase ends with a well defined milestone. At these points, the stakeholders assess the project, including what has been done and the plans for moving forward. A go/no-go decision is made about whether to proceed with the project. Each phase has a specific set of goals, which are addressed within the iterations of the phase, so that the phase milestone may be met.

This methodology emphasizes on accurate documentation It is proactively able to resolve the project risks that are associated with the clients evolving requirements for careful changes and request management. Very less need for integration as the process of integration goes on throughout the development process.

RUP phases

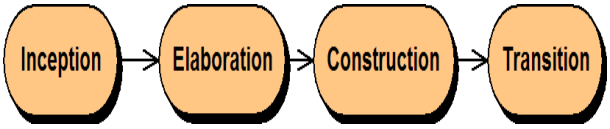

Fig 1.4 The RUP phases

Dynamic System Development Method

DSDM is an iterative and incremental methodology that combines the project and product management life cycle into one best process. DSDM is not suitable for all projects. In particular, systems that are real-time, safety critical, or have well defined requirements are not suited to the use of DSDM. DSDM is especially suited to projects with changing requirements. It is critical that these requirements be capable of being prioritized.

There are 9 underlying principles of DSDM consisting of four foundations and five starting-points for the structure of the method. These principles form the cornerstones of development using DSDM. Many development methods ask for testing as late as the design or implementation phase. DSDM requires testing early in the development process. Even testing interview documents by cross checking them with a control group, or similar techniques.

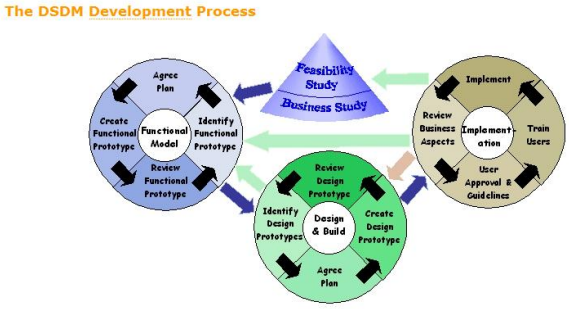

Fig 1.5: DSDM Process

Principles of DSDM

There are 9 underlying principles of DSDM consisting of four foundations and five starting-points for the structure of the method. These principles form the cornerstones of development using DSDM.

1. Active User Involvement is Imperative

The first principle is considered the most important, because user involvement throughout the project effectively reduces errors in terms of user perception, and therefore reduces error costs. Instead of working with a large set of users a DSDM project guidelines recommend working with a small, select set of users continually, rather then in periodic workshops or review sessions. Continuity is required by several other DSDM principles as well, highlighting this single project attribute even more.

2. Teams must be empowered to make Decisions

To proceed as quickly as possible transaction costs, as resulting from friction in communications of project participants and managers, need to be avoided. Requesting authorization of even modest resources, or simple requirement changes will slow down a project significantly. Addressing these inefficiencies users and other DSDM participants should be given limited authority to make decisions related to:

- Requirements in practice W
- Which functionality needs to be in a given increment
- Prioritization of requirements and features
- Fine details of the technical solution

3. Focus on Frequent Delivery

Frequent deliveries of results ensure that errors are detected quickly, are easily reversed and closer at the source of the error. This applies both to program code as well as to documents like requirements or data models.

4. Fitness for Business is Criterion for Accepted Deliverables

As the name of the DSDM framework suggest, its primary endeavour is to deliver software which is good enough to solve the business need and bother with any enhancements in a later iteration. Refactoring, design engineering and feature enhancement, being part of the natural live cycle of any software system, need to be recognized as integral part of the project, rather than being a task which is taking place only after the project is finished. DSDM does not promote to write ad-hoc software, but suggest satisfying the business needs first, and acquire time boxes for refactoring and related activities in a later iteration. Even on a DSDM project it's crucial to identify the key issues, which do required a robust design. Agile methods in fact do require a sound understanding of patterns and architecture; depending much more on patterns then traditional development methods, because it is fare more likely to make a decision which will turn out to prohibit delivering iteration results three or four iterations into the future.

5. Iterative and Incremental Development is Mandatory

In order to keep the complexity of the project manageable, it needs to be decomposed into small feature packages; with each release adding new features until the complete set of business requirements are fulfilled. This principle requires accepting the fact that any software

system is subject to change. This principle can easily be introduced even at the beginning of a project, since specifications and other results can be produced in an iterative manner as well. The smaller the increments, the easier a response to changed requirements is possible.

6. Requirements are Base lined at High-Level

To limit the degree of freedom to which requirements can be altered during the development process, some high-level requirements need to be established. This baseline which is to be interpreted as a requirements "freeze" is agreed upon during the business study phase of the process.

7. Testing is Integrated Throughout the Lifecycle

Many development methods ask for testing as late as the design or implementation phase. DSDM requires testing early in the development process. Even testing interview documents by cross checking them with a control group, or similar techniques.

8. Collaborative and Co-operative Approach

Avoiding separation and encouraging collaboration of technical staff and business staff in a project is mandatory during DSDM projects, because co-operation is crucial to succeed in a DSDM project.

3.3 Extreme Programming

This is a software engineering model that was developed by Kent Beck in 1996. Software Project management can use this model in designing their software project. This model is aimed at reducing risks in the software development and management process. In project management, delays in making decisions can result into huge losses. XP is aimed at reducing the cost as a result of delaying decisions. This model pays attention to the cost and revenues of design decisions. Delaying decisions on whether to implement options in software development is very risky. This is because implementing options later becomes more expensive at a later date.

Exploration Phase:

In this phase the customer write down the stories which are actually the requirements of the system, so customer put these stories on stories cards and in the mean while the development team practice there tools and technology which they are using in this project. So the technologies are tested and system get visualise by building the prototype of the system. Depending on the size of the project this phase may take a few weeks to a few months, depend on the situation of the project and technology (Pekka et al. 2002).

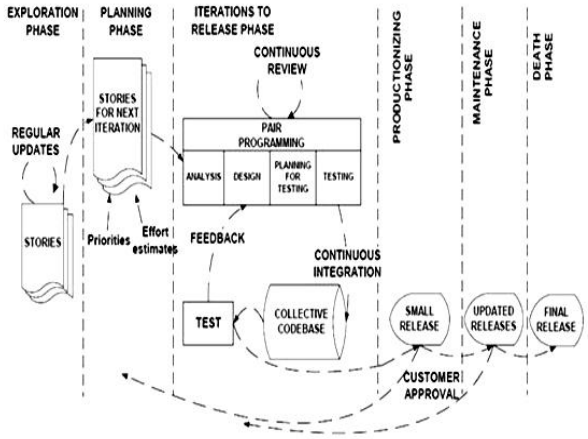

Figure 1.6 Shows the Life Cycle of XP Process

- Extreme programming methodologies emphasis on customer involvement.
- This model helps to establish rational plans and schedules and to get the developers personally committed to their schedules which are surely a big advantage in the XP model. This model is consistent with most modern development methods so, developers are able to produce quality software.

Planning Phase:

As in the exploration phase the user write down the requirement stories these stories are then prioritized according to the requirement of the system and the contents of the initial release are prepared. Then the developer's team estimate the requirement and work for the first release and mange the time and resources schedule. This phase take a couple of days and the first release are coming out in round about two months (Pekka et al. 2002).

Iterations to Release phase:

In this phase the schedule which was planed are divided to multiple iterations. And in the first iteration the architecture of the system is created, and this will be the result of the selected stories that focuses the developers on the structure of the required system. The customer decide which stories to be include in every iteration and the necessary tests which the customer wants are developed and applied on code after each iteration. Each iteration take approximately 3-4 weeks to implement, and the system will be ready to use at the end of last iteration (Pekka et al. 2002).

Productionizing Phase:

As in this phase the system is ready for the first release but before release it is tested for extra performance and functionality, may be some new amendment are also decided if they are the part of the first release. Some new ideas and suggestions may be arisen so they are recorded and might be implemented later, because this phase is the one to be completed in short time from one to three weeks (Pekka et al. 2002).

Maintenance Phase:

In this phase as the system is release to the customer so the XP team will be working on both sides on the customer support side as well as on new iteration, this can reduce the speed of the developers and may be some new people are introduced to the XP team and the team will be restructured.

Death Phase:

In this phase system is completed, customer is satisfied from the performance of the system and customer requirements are finished. All work is documented and design, architecture, and code are finalized. The death phase also occurs if the desired outcomes are not fulfilled or if the system is going to be too expensive to afford.

EVALUATION RESULT

Comparsion of Models

Software models are very important in software project management. There are several software management models that are used in the process of managing a software project. Since there are many of the techniques and models, in this paper selects and analyzes some of them. Project managers should select a project model that they are familiar with and is most suitable for the project. The main reason for looking at the software models is the limited information on the usage of the models. The size of the project and the duration for the development of a project should be considered in software development model. In this paper three software project management models to analyze and compare, which are Rational Unified Process (RUP), Dynamic System Development Method (DSDM) and Extreme Programming (XP).

A Hybrid Software Development approach for Small to Medium Scale Projects, which has used the models RUP, DSDM and XP. To comparing these models user represent characteristics, strengths, and weaknesses of RUP, DSDM and XP process models, and a new hybrid software development model, which integrates the strengths of RUP, DSDM and XP while suppressing their weaknesses.

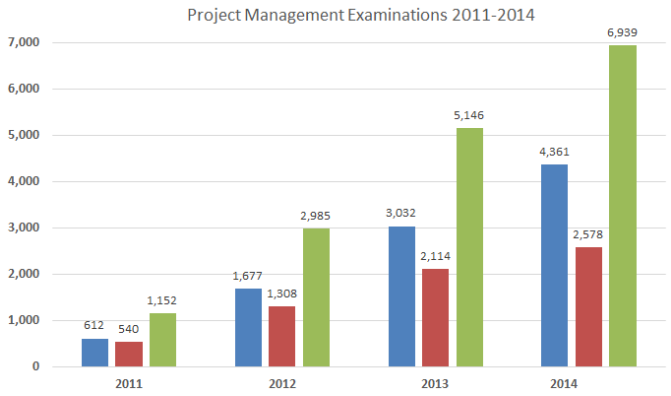

Fig 1.7 Project Management Examinations

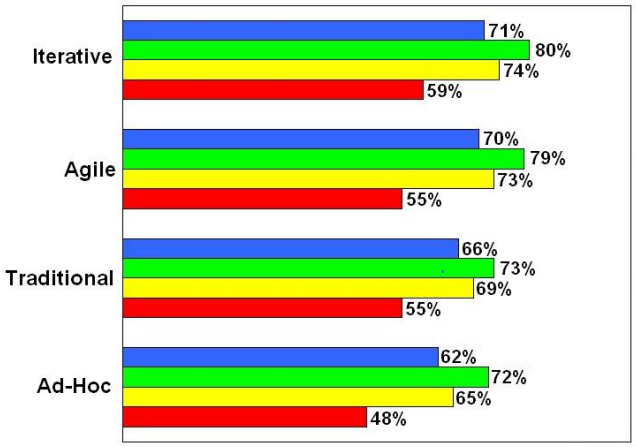

Fig : Comparing paradigm success rates by geographic distribution.

The feedback cycle of common development techniques (for the sake of brevity not all techniques are shown) to the cost curve. Agile techniques, such as Test Driven Design (TDD), pair programming, and Agile Model Driven Development (AMDD) all have very short feedback cycles, often on the order of minutes or hours. Traditional techniques, such as reviews, inspections, and big requirements up front (BRUF) have feedback cycles on the order of weeks or months, making them riskier and on average more expensive.

1) Rights and Responsibilities

I used to talk about the rights and responsibilities of project stakeholders, a concept that I adopted from Karl Wiegers excellent book entitled Software Requirements. Unfortunately, I recently discovered that these rights and responsibilities as written in Agile Modeling weren't as clear as I had originally thought and were potentially even divisive. So, after a fair bit of discussion on the AM mailing list I've decided to reword the rights and responsibilities from the point of view of everyone on the project, not just the stakeholders.

The rights of everyone:

1. To be treated with respect.
2. To produce and receive quality work at all times based on agreed to project standards and principles.
3. To estimate the activities you are actively involved with, and to have those estimates respected by others.
4. To be provided adequate resources (time, money, and so on) to do the job that's been asked of you.
5. To determine how your resources will be invested. For people funding the project how the funds will be spent and for people working on the project (e.g. investing time) what tasks they choose to work on.
6. To be given the opportunity to gain the knowledge pertinent to making the project a success. Business people will likely need to learn about the underlying technologies/techniques and technical staff to learn about the business.
7. To have decisions made in a timely manner.
8. To be provided good-faith information in a timely manner. Sometimes this is just the "best guess" at the time, and that's perfectly all right. This includes but is not limited to business information such as prioritized requirements and detailed domain concepts as well as technical information such as designs and detailed technical concepts.
9. To own your organization's software processes, following and actively improving these processes when needed.

Comparison RUP/ XP/ DSDM

Tasks / Processes:

Requirement Analysis and Business Modelling.

- RUP begins by creation of the vision document which comprises business motivation required system features, pre-conditions and constraints, risks, main use-case analysis.
- XP have the user stories with communication feedback and on-site customer.
- DSDM begins with the creation of feasibility report, feasibility prototypes and outline project plan. Requirements priorized by MOSCOW techniques.

Analysis & Design

- It has general project description and it is more precise definition of min use-cases. It has UML diagrams such as Class diagram, sequence diagram, collaboration diagram and Activity diagram.
- It has to develop with the simple design and system metaphor.
- It has not required any analysis and design but it has used only prototypes to manage

Implementation and Development

- Based on these descriptions, user creates a main use-case and prototypes architecture.
- XP has multiple tasks are required small releases, continual integration, collective ownership, refactoring and par programming.
- Common understandings of technique architecture used and implement with train users, user approval and review business

Project management

- Detailed project schedule should be required and defined the project plan.
- XP stories need to estimate and it has iteration plan.
- Priorized requirement list with outline prototyping plan

Project Size

- It is used for medium to large scale projects.
- It is used for small projects.
- It is used for large or small projects.

Configuration and change management

- Change control strategy is required.
- It does not require.
- Products delivered frequently at very fast rate, products need to be strictly complete.

Pair programming

- It does not required Pair programming.
- It implements the Pair programming to develop the software projects.
- It requires for project controls

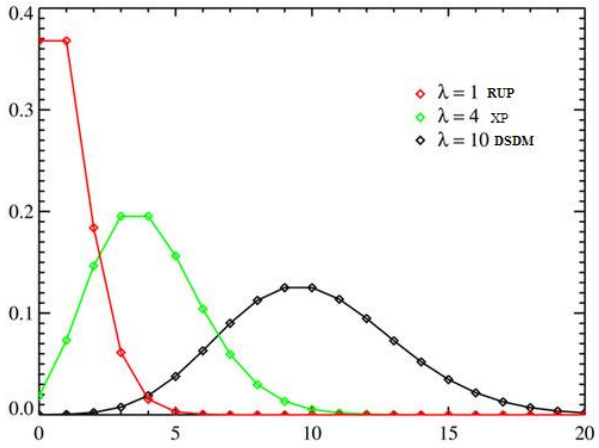

Fig: Comparison of RUP/XP/DSDM

CONCLUSION

Software models are very important in software project management. Project managers should select a project model that they are familiar with and is most suitable for the project. The size of the project and the duration for the development of a project should be considered in software development model. This paper has only looked at three software development models with much emphasis on the management part. There are many other software models that can be selected in a project. It is also possible to combine two or more models in a single project.

In this paper, user has analyzed the similarities and differences between RUP, DSDM and XP methodologies, based on the many keywords and key values. In the above discussion DSDM are the best one. Common values are user/customer involvement, iterations, continuous testing and flexibility. The implementation of these values are however very different. DSDM have less iteration than RUP and XP. DSDM enables us to work with the customers to identify what really benefits them, then deliver a system that targets those benefits. The experience was shown that DSDM delivers in half the elapsed time for half the cost for a third fewer technical errors when compared to traditional project lifecycle approaches. The system delivered less than user asked for, but user expected more than that because they got exactly what they needed. The above points are helped to suggest DSDM is the best one.

REFERENCES:

1. P. Abrahamsson, O. Salo, J. Ronkainen and J. Warsta, "*Agile Software Development Methods-Review and Ana- lysis*," VTT Publications 478, 2002.
2. D. Truex, R. Baskerville and J. Travis, "*A Methodical Systems Development: The Deferred Meaning of Sys- tems Development Methods*," *Accounting Management and Information Technologies*, Vol. 10, No. 1, 2000, pp. 53-79.
3. S. Cronholm, "*Using Agile Methods?—Expected Ef-fects*," *Proceedings of 17th International Conference on Information Systems Development* (*ISD*2008), Paphos, 25-27 August 2008, pp. 913-924.
4. N. Ganesh and S. Thangasamy, "*Issues Identified in the Software Process Due to Barriers found during Eliciting Requirements on Agile Software Projects: Insights from India*," *International Journal of Computer Applications*, Vol. 16, No. 5, 2011, p. 7.
5. B. Boehm, "*Get Ready for the Agile Methods, with Care*," *Computer*, Vol. 35, No. 1, 2002, pp. 64-69.
6. R. F. Roggio, "*Process Driven Software Development: An Approach for the Capstone Sequence*," *Proceedings of Information Systems Education Conference* (*ISECON*), Pittsburgh, 1-4 November 2007, pp. 234-242.
7. J. Newkirk, "*Introducing to Agile Processes and Extreme Programming*," *Proceedings of 24th International Con- ference on Software Engineering*, Orlando, 25 May 2002,
8. P. Abrahamson, "*Extreme Programming: First Results from a Controlled Case Study*," *Proceedings of 29th Eu- romicro Conference* (*EUROMICRO*'03), Antalya, 1-6 September 2003, pp. 259-266.
9. L. Lindstrom and R. Jeffries, "*Extreme Programming and Agile Software Development Methodologies*," *Informa- tion Systems Management*, Vol. 21, No. 3, 2004, pp. 41- 52.