A recommendation report on Spare Time Product Development

Pujar, H.; Halyal, S.

Award date:
2010

A recommendation report on Spare Time Product Development

Hema Pujar
0729932
Smitha Halyal
0729096

Master's thesis August 19, 2010
Department of Mathematics & Computer Science
Software Engineering & Technology Group
Eindhoven University of Technology

Abstract

It is the usual scene in the IT companies that employees have not enough clients. According to a recent study, at times, 30 percent of the employees in an IT company do not have clients. This results in idling of the skilled employees and they are said to be on bench. Capgemini has come up with an idea called Spare Time Product Development (STPD) as a measure to prevent its employees from facing such situations. This is adopted for effective human resource management, when the organization has not enough clients for its employees to work. The STPD is adopted to manage employees both in terms of employee management and also from business management perspective. It is used in developing projects involving people on bench. The people on bench constitute the Cloud Team of the STPD. It also has a Core Team, who is the permanent members of the development team and a Solution Manager for facilitating the project development. Capgemini develops RUP based and agile based projects. Our research study was to formulate the requirements for task creation and knowledge transfer required in STPD. To study how much of RUP and Agile processes fit into STPD and providing recommendations for developing RUP/Agile projects using STPD and tooling were our other research questions.

The research was carried out by understanding the STPD, its characteristics and defining requirements. Then this was followed by the study of task creation and knowledge transfer and formulating the requirements for both. The RUP/Agile process fitting into STPD model was studied and recommendations related to tooling for both RUP/Agile are provided.

Acknowledgment

This master thesis is written as part of Master's program in Computer Science and Engineering at Department of Software Engineering and Technology, Eindhoven University of Technology. There are many people who helped in conducting this research study and we would like to thank them for their precious time and immense support they provided us.

Before we proceed to thank all, we would like to thank Manipal University and Eindhoven University of Technology for providing us the opportunity for pursuing dual master degree program.

We would like to thank our supervisor Prof.dr. Mark van dan Brand, Professor Software Engineering and Technology, Department of Mathematics and Computer Science. Without his guidance, support and constant encouragement it might have been difficult to conduct this research study. His helping ideas always guided us the way to proceed in all stages of the research. With deep respect we convey our sincere thanks to him.

We would also like to thank our supervisor from Manipal University, Dr.(Prof).Manohara Pai.M.M for his guidance.

We would also like to thank our supervisors from Capgemini Utrecht, Marijn Ruster, Principal Consultant and Raymond Honnings, Principal Consultant. We are grateful to them for providing an opportunity to carry out our internship at Capgemini and also for providing us with excellent facilities and great cooperation. We are thankful for their guidance throughout our research study. We would also like to thank all the respondents in Capgemini for their valuable time and encouragement.

We would also like to thank Prof dr.Mark van dan Brand, dr.Alexander Serebrenik, dr. Michael.A.Westernberg, Raymond Honings, Capgemini and Prof.Balachandra, MU for being the members of our assessment committee.

Last but not least we would also like to express our sincerely gratitude to all our dear friends for their support.

Contents

1. Introduction

In this chapter section 1.1 gives a brief introduction to Capgemini. Section 1.2 provides an overview of Spare Time Product Development. Section 1.3 explains STPD in analogous to jigsaw puzzle and lists the issues that needs to be addressed in Spare Time Product Development. Section 1.4 contains our research questions. Section 1.5 describes the approach we used. Section 1.6 gives the document overview.

1.1 Introduction to Capgemini

At times IT companies do not have enough clients. This leads to people idle in bench. Capgemini wants to effectively utilize the bench time. Before describing how Capgemini achieves this, a brief introduction to Capgemini is given.

Capgemini is one of the world's largest information technology, management consulting, out sourcing and local professional services companies headquartered in Paris, France and operating in many countries.

Capgemini employees are grouped into four major disciplines, each of which is governed by its specific economic rules and managed with its own profit. It helps clients deal with changing business and technology issues. The company's relationship with its clients is a partnership, bringing the company's experience, best practices and tools to apply to client's unique requirements. It provides wide range of solutions within four professional disciplines. They are

- Capgemini Consulting: It helps clients in identifying, structuring the transformation strategies.
- Technology Services: This is the service provided in designing, developing and implementing a variety of technical projects.
- Outsourcing Services: To meet client's operational and strategic objectives, Capgemini offers a range of flexible outsourcing solutions to optimize service levels, improve cost structures. Capgemini's one of the outsourcing branches is in Mumbai, India.
- Professional Services: It is about Capgemini offering services in terms of infrastructure, applications and engineering [1].

As a service provider it has seen changes evolving in the software industries and has commendably adapted itself to the changing market.

1.2 Concept of Spare Time Product Development

The idea Spare Time Product Development (STPD) is initiated by Capgemini. To describe in simple terms, it is the development of a product or a part of a product by the people on bench. This initiative by Capgemini, gives an opportunity to its employees to enhance their skills and also it serves as means for employees to educate themselves in various phases of product development. The business model of this concept is shown in Figure 1.

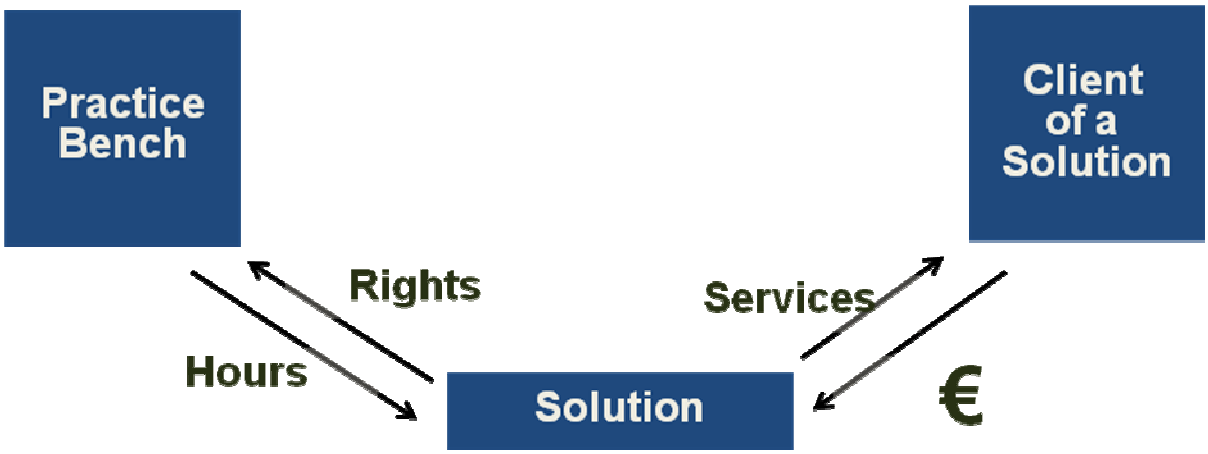

Figure 1: Business model

In the Figure 1, the practice bench is the people on bench and they work on a solution/product of a future client. Here the solution (in the Figure 1) is the application which is provided as a service to the future clients. The employees who work on developing the solution/product receive rights at a later point of time, based on the number of hours they have worked. A discussion was held between us and our guides at Capgemini, to identify the roles in this model. Primarily three high level roles have been identified for this model, they are

- Project manager: The project manager coordinates and manages all the tasks of product development. The project manager is part of the core team but is mentioned separately since this role has specific responsibilities compared to developers.
- Core team: The core team members are the permanent members in STPD.
- Cloud team: Cloud team constitutes people on bench. They are non permanent members.

1.3 Context

To get a better understanding of STPD the following analogy is presented. STPD is analogous to a jigsaw puzzle. In the STPD model, the product to be developed is the jigsaw puzzle. The jigsaw puzzle game involves the assembling of numerous small, different shaped pieces.

The amount of time a cloud team (refer to 1.2) member can spend on the project may be restricted. This necessitates the product developed by STPD to be broken down in a number of tasks. The tasks to be done to complete the product resemble the pieces of the jigsaw puzzle.

Consider a jigsaw puzzle, in which different parts of the puzzles are completed by different players, at different times. In STPD the people moving in and out of the project resemble to players as mentioned in jigsaw puzzle.

The cloud team is not fixed and all the members may or may not be available till the end of the project. It is important and necessary to manage the knowledge (information about the task) in STPD environment. In such an environment, knowledge about the tasks must be available to new members of the cloud team, which can be termed as knowledge transfer.

The environment must also posses the features for a task to be developed by the people moving in and out of the project team. This requires a study of task creation. To develop a product with STPD a proper process and good tooling support is required.

We could not confine our work to a particular product type, since STPD needs to be applied to every type of software to be developed. For our master project, we were given the business model. We tried to explore the anticipated STPD environment and identified the issues which would arise in the projects developed by STPD.

1. What kinds of people are on bench?
2. How long are people available on bench?
3. What kind of product is developed by the STPD?
4. How many people are available on bench?
5. Is there any coding standard or architectural style to be followed in the STPD model?
6. Which process to be used for the STPD?
7. What is the tooling support for STPD?
8. What are STPD model elements?
9. Is the core team required?
10. What should be the size of core team?
11. Does the STPD require Project Manager?
12. When to use the core team?
13. When to use the cloud team?
14. What will be the work distribution between the core team and the cloud team?
15. What is the purpose of STPD?
16. What happens to the product under development, if there are no people on bench?
17. Who assigns the work?
18. Who is responsible for defining the problem?
19. Are there customers with problems?
20. Whether the tasks defined can be done in the spare of time of an employee?
21. Whether tasks defined requires the understanding of the system being built?
22. If the size of the core team is more than one, what kinds of people constitute the core team?
23. What happens to the task, chosen by a cloud team member, leaves it without completing?
24. How the information about the status of the project and the tasks are conveyed to the cloud team members?

25. Which is the medium used for communication in a team?

Of all the above mentioned issues, the answers to questions 1and 20 are the assumptions made to support. The questions 6, 7, 17, 21, 23, 24 and 25 helped to formulate our research questions. The questions 1, 6, 11, 12, 13, 15, 17, 20, 21, 23, 24 and 25 helped to formulate the characteristics and requirements of the STPD. Questions 2, 3, 4, 5, 10, 14, 16, 18, 19 and 22 were beyond the scope of this thesis.

1.4 Research question

In our master project we have addressed the following questions

1. What are the requirements for STPD?
2. What are the requirements for task creation?
3. What are the requirements for knowledge transfer?
4. How can Agile software development method process be adapted to work with STPD?
5. How can RUP be adapted to work with STPD?
6. What tools support the above adapted processes?

The first research question is addressed by Hema and Smitha. The second, fifth and sixth research questions is addressed by Hema. The third, fourth and sixth research questions is addressed by Smitha.

1.5 Approach

We addressed our research questions in the following way.

- Our first step was to understand the model, its characteristics and define requirements and assumptions.
- In our second step, we formulated the requirements for task creation and knowledge transfer.
- Our next step was to adapt the processes (Agile and RUP) to work with the STPD.
- Last step was the study of tools, which support the adapted process.

The result of our work is a set of recommendations to facilitate STPD.

1.6 Document overview

Chapter 2 describes the Spare Time Product Development model (STPD). Chapter 3 provides an overview on existing approaches. Chapter 4 describes task creation and knowledge transfer in STPD. Chapter 5 describes RUP process adaption and supporting tools. Chapter 6 describes Agile adaption and supporting tools. Chapter 7 concludes the research work. Work related to RUP process, task creation and tooling is done by Hema and work related to Agile software development process, knowledge transfer and tooling is done by Smitha. Rest is a combined effort.

2 Spare Time Product Development model (STPD)

2.1 Purpose of STPD

The idea is adopted by Capgemini for effective human resource management, when the organization has not enough clients for its employees to work for. If the employees have no clients then they have no projects in hand to work. They have spare time and are said to be on bench. According to recent studies, technology companies have an average of 30 percent of their employees on bench [33].

Capgemini provides its employees an opportunity to work on projects during their spare time. The projects being carried out using STPD are incorporating new features in the technologies like Microsoft Visual Studio .Net. The main idea behind STPD is to use it in developing solutions for future clients. The people from bench are involved in developing such solutions and are awarded rights or incentives in future. This totally benefits the people on bench during their spare time and the organization is also profited.

The primary objective of the STPD is to prevent highly skilled employees from idling. The organization takes measures to enhance technical skills, managerial skills and also in personal development of the employee for the upcoming projects in their spare time.

2.2 STPD

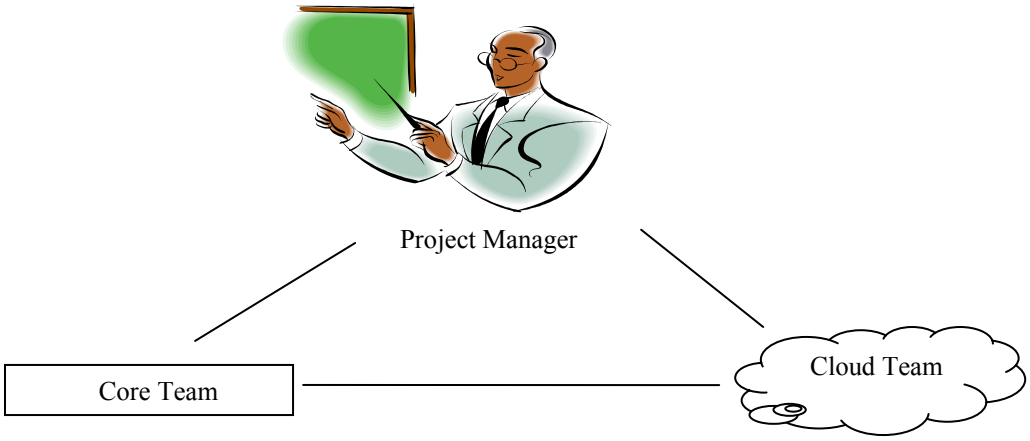

Figure 2: STPD model

The STPD model Fig.2 shows the people involved in it. As described in section 1.2, there are three elements. They are: a Project Manager, Core team and Cloud team.

Capgemini uses this model in developing solutions which are frequently demanded by the clients. These solutions are built using STPD model and they are modified or extended with features required by the future client. These modified or extended solutions are then delivered to the future clients. The clients pay for the solutions delivered and the people who contributed in developing the solutions are awarded rights. STPD team participates in building the solutions.

Project Manager of STPD is the leading person in developing projects using this model. He is responsible for building products, guiding both the core team and the cloud team in the development of the product till the delivery of the product.

The Core Team

The Core team has permanent members. They represent the development team of a project. They know about the product under development in depth and they develop mainly the main functionalities of the product. The size of the STPD model core team is at least one. Project Leader plays this role in the STPD when its core team size is one. This Project Leader in the team assists the Project Manager in all stages of the product development. Project Manager is one permanent member of the STPD who has the complete knowledge of the project under development.

The responsibilities of a Project Leader are:

- He is responsible for defining the tasks.
- He creates the task list.
- He might assign tasks to the core team members.
- He guides the core team and the cloud team in developing the project.
- He provides assistance to the core team and cloud team.
- He arranges for meeting, if necessary.

The Cloud Team

The Cloud team is made up of employees from the bench. The members are not permanent and their number in the team is not fixed. They are associated with the development of the project for a specified period of time. This period of time is the time during which they have no clients to work for, is the bench time. The members of this team move in and out of the team and do not have knowledge about the product under development. The tasks of the project are defined such that they do not require the knowledge of the product under development. They perform low level implementations of these tasks.

2.3 Characteristics of the STPD

In the section 1.3 questions are mentioned. The answers for the questions 1, 6, 11, 12, 13, 15, 17, 20, 21, 23, 24 and 25 in section 1.3 helped to define the characteristics of the STPD. The characteristics of the STPD are listed out below.

The characteristics of the STPD model are:

- The bench is not fixed.
- There can be a Software Architect or a Designer or a Requirement Specifier or a Software Developer or a Tester on bench.
- The time people spend on the bench is not fixed but requires a minimum amount of time to be available.
- Core team size is at least one, it is the Project Leader.

2.4 High level requirements/Assumptions for the STPD

As earlier said in section 1.3, many issues were identified in developing projects using STPD and are listed as questions. The answers to questions 1, 6, 11, 12, 13, 15, 17, 20, 21, 23, 24 and 25 were defined in such a way that they became requirements to develop projects using STPD. The requirements are listed out.

Requirements of the Spare Time Product Development model are:

- An employee must be available for certain fixed amount of time.
- Tasks of a project must be defined such that they are implementable without requiring the knowledge of the product being developed.
- Information about the project must be managed and transferred till the end of the project development.

There was a need to make some assumptions about STPD for us to proceed. There can be any expertise on bench. Expertises cannot be of same kind on bench. As we were asked to study about developing RUP/Agile using STPD in the later stage of research, the second assumption was made to limit the research and to complete the thesis in time. The cloud team members move in and out during project development. So an assumption was also made related to this. The assumptions made are:

Assumptions for the STPD:

- All the tasks defined can be completed within the spare time of the employee.
- There are only software developers on the bench.

The above listed requirements and assumptions for STPD answers our first research question mentioned in Section 1.4.

3 Existing approaches

This chapter describes some of the projects carried out using STPD model at Capgemini in existing approaches in Section 3.1. The Section 3.2 of this chapter describes the open source model and how it is similar and dissimilar to the STPD model.

3.1 Within Capgemini

This section briefly describes the projects in Capgemini which are developed based on the STPD model. Agile Dashboard and VSTS2010@ADC Capgemini CSD are two such projects which are being carried out based on the STPD model. The process and the tools used are similar to the open source environment. As in open source, the tools help the people on bench to choose their tasks and do their work. Their work is committed in the repositories maintained. The two project processes are described in the next section of this chapter.

3.1.1 Agile Dashboard

Agile dashboard is Capgemini's smart agile software development platform. It captures and combines the best practices mainly in .Net increasing the productivity and quality of the software development. This project uses a dashboard which is publicly available. It helps to track projects online, facilitates simple charting such as burn-down charts. It does not have user authorization and does not involve extensive reporting.

The objective of the project is to make the Capgemini agile ready/ADP (Accelerated Delivery Platform) and to upgrade the Agile Dashboard to display the knowledge, experience of Capgemini in the enterprise agile and the enterprise .Net spaces.

The project team has a Project Manager who facilitates the project development. There is a Project Leader who is the permanent member of the team guiding the project development and there is a developing team which constitutes people from bench. The project follows the Smart agile process and has a team which is staffed with the available resources. The team is stable during two week iterations and the documents are blogged.

The Project Manager of this team was interviewed to know how the communication takes place between the team members and how tasks are created in such environment.

In this team as earlier mentioned, documentation is blogged. The team members blog about their work. The dashboard has colored sticky notes. These sticky notes with text represent tasks with their descriptions. The color of a sticky note represents the status of the task. They are stuck in three different columns namely done, to be done and in progress columns. The color and the position of the sticky notes convey the same information. This dashboard with sticky notes in different columns provides the team members the visual representation of the status of the project and its tasks. This is how the knowledge about the project/tasks is conveyed to the members in the team.

This project involves people on the bench in extending the tools based on .Net technology, with new features. These people move in and out of a team. They are allowed to choose their feature to enhance the tool. These features are the tasks. These tasks are defined in such a way that, to work on them does not require the knowledge of the entire application. If a task is big for a person to complete during two weeks iteration, then he asks the Project Manager that the task has to be broken into smaller subtasks to work. These tasks must be breakable. Another feature of this way

project development is that if a person leaves a task without completing it, then the person coming in can work on this uncompleted task by reading the blogs of this task. The uncompleted task is saved in their repositories maintained. The project is about extending the .Net tools with new features with the people on bench as the team members.

3.1.2 Conclusion about Agile Dashboard

This project involves people from bench in incorporating features in new technologies. This is based on STPD but if this way of developing projects is used for developing product then it would be possible. The tools used for incorporating features possess the required features for developing a product STPD. There are no proper knowledge transfer features which are required for communicating between the team members. It requires team members to be in one room. They can be extended with some features for knowledge transfer techniques for better performance. This is an example project for our research study.

3.1.3 **VSTS2010@ADC Capgemini CSD**

VSTS2010@ADC is similar to Agile Dashboard project running in Capgemini. The aim of this project is to study the new features Microsoft Visual Studio. The study of features is done to understand how the features can be used. The Project Manager of this team was also interviewed to know how the members of the team communicate and how tasks are created.

This project hires people on bench to analyze the features of Microsoft Visual Studio 2010. People on bench with specific skills are asked for working in the project. The live meeting with the team members is the medium of communication between the team members of the project. Project Manager of the team lists out the tasks or the features of .Net technologies in an excel sheet with name of the task, status of the task and the date on which the task is taken up a person as the columns. The team members select the features from this task list and work on them. One person can work on one task/feature at a time.

The features analyzed by the team members are documented and are saved in the repositories maintained. If a person working on any feature gets assigned to some project, then he commits his work. Another person coming in can take up the same task read the work done and can work upon it. The work done is saved for the cloud team members.

3.1.4 Conclusion about VSTS2010@ADC Capgemini CSD

This project is described in the existing approaches because it hires people on bench and the assigned members can leave this project any time they get new billable assignment. This served as the basis for our research. The process and tool used in this approach was to study new features. STPD is different, in the sense, it is used to develop products. The product is developed by core and cloud team. This requires proper process and strong tooling support.

3.2 Open source

The open source is a way of developing software which relies on the contribution of developers who are geographically located at different places. These developers contribute using internet. This way of developing projects is a classic example for collaborative way of developing projects. It is a way to develop software involving people from anywhere.

The open source project development model has roles playing in developing projects. The roles in the open source model are the owner, contributors and core developers. The roles of the open source model are described.

The owner of the project is one who has problem and wants to find solution by developing product. He develops the first solution of his problem. He then makes it publicly available and announces this in various open source platforms to attract interested people to contribute to his product. As an owner, he attracts the contributors.

The persons who all are actively progressing the open source project are the contributors. The developers who are contributing to the improvement of the product or are developing its lacking features are contributors. They contribute to such project if they are in need of it or the project is interesting to them and they might code for it. They find the project interesting and contribute but are not asked by anyone. They are self motivated.

Core developers are the frequent contributors to the project. They develop most of the main functionalities of the project and maintain them. The owner of the project (an individual or a team) then becomes more of a co-coordinator and a project leader than a code writer. He decides what modifications have to be considered for the new versions of the project.

Why study Open Source project development?

The main reason for describing the open source project development model is that there are similarities between the STPD and the open source project development model. The STPD model has similar roles as in the open source project development model. The core developers of the open source model represent the Core Team of the STPD as both develop the main functionalities of the system being built. They are permanent members in STPD and develop the main functionalities of the project. The contributors of the open source model are not permanent and they move in and move out of a team at any time during the project development. These are similar to the Cloud Team of the STPD. The owner role of an open source project is similar to the Project Manager role of the STPD, who coordinates and manages the project development.

Besides the similarities there are also dissimilarities between the open source project development model and STPD. In the open source model, members of the team have the complete privilege to choose their task of interest while in STPD the Core Team members might be assigned by the Solution Manager. The open source project development is not planned. The requirements keep changing and they have many releases. In STPD, what the end product has to be built is known.

The main dissimilarity between the two working ecologies is that the contributors of the open source model code because of their personal interest. They get motivated by themselves and contribute. They are not time bounded and are not awarded for their work. But the Cloud team members of the STPD which are similar to the STPD are awarded for them to keep motivated all the time. They work on projects during their spare time for which they are rewarded. They contribute and are rewarded with incentive for their contribution. They are time bounded for completing the tasks. This is the main reason for not adopting the open source working culture and which differentiates both open source model and the STPD.

Though there are dissimilarities, open source project development study helped in understanding how people moving in and out of a project development called Cloud Team can contribute to project development. Open source study helped in understanding the participation of the Cloud Team in a project development.

4 Task Creation & knowledge Transfers

This chapter describes about the study of task creation and knowledge transfer in STPD. The various features for task creation and knowledge transfer are studied and the features suitable for developing projects in STPD are listed out as requirements for STPD tooling.

As our objective of research was to study about task creation and knowledge transfer in STPD, these are in this chapter. Later the objective changed to RUP/Agile projects developing using STPD. So both these objectives are linked to give a flow to our research study and to give a proper conclusion to STPD. So the requirements for both task creation and knowledge transfer are formulated first and then recommendations for RUP/Agile are made later.

4.1 Task creation

This section describes the various features for task creation in a project development environment. The features for creating tasks of a project in various project management tools are studied. The main features required for creating tasks of a project to develop using STPD model are listed out as the requirements for task creation.

A project is broken into its tasks. These tasks are implemented and integrated in its development for the complete project as end product. If the project is a big project with hundreds of tasks, then it requires a project management tool for developing and maintaining the tasks of the project and the project itself. The project management tools provide features for uploading the defined tasks of a project by a Project Manager, features for implementing the tasks of the project by developers without providing any opportunity for rise of issue between developers of these tasks. Such features for task creation are studied in order to list out the necessary ones for creating tasks of the projects developing using STPD model.

Task creation features are usually for enhancing collaboration capabilities in a project development tool. These features are provided to aid the Project Manager or the administrator of the project who uses the tool, in developing of project tasks. Such tools are mainly used where team members are located in geographically different places and are involved in the same project development. They have same objectives to be achieved. The tools must provide the features required for the efficient development of such project though there are various hindrances as the team is not under one roof.

The section 4.1.1 describes the various tools studied to learn the features for the task creation, 4.1.2 of this chapter describes about the features for task creation and section 4.1.3 briefly describes about a project in Capgemini similar to RUP based, if this project has to be developed using STPD model.

4.1.1 Project management tools studied

This section briefly describes the various Project management tools studied to learn the different features associated with creation of tasks. The features described in section 1.2 are the union of features found in the tools mentioned below.

Project Management Tools:

1. Gantt Projects[17]
2. HotProject [18]
3. Teamwork Project Manager[19]

4. @task[20]
5. activCollab [21]
6. CentralDesktop [22]
7. WorkBench[23]
8. Planzone[24]
9. CoMindWork[25]
10. Zoho Projects[26].

The above tools and their features were studied in order to learn the various task creation features. The features for creating tasks from all the tools are categorized and summarized and listed out in the next section.

4.1.2 Features for task creation

The various features for creating tasks, communication medium while creating tasks, notifications, version control for the tasks studied in various project management tools are listed out below.

There must be a feature for uploading of tasks of a project. Visual representation in a project management tool involves representing the tasks in a project which gives the information about the status of the tasks and the entire project itself. The status of the task involves representing whether a task is completed/in progress/incomplete. This is another feature.
Example: A dashboard representing with colored sticky notes on it as in an agile project development tool. The sticky notes represent the tasks and the color of the sticky notes represents the status of the task. Green colored sticky note represents completed tasks, saffron colored sticky notes represents in progress tasks and red colored sticky notes represents the incomplete tasks. The dashboard with such sticky notes represents the complete status of the project.

The feature for letting user to choose their task gives a user complete freedom to choose his task to work upon it. They choose and make it is as assigned to themselves by selecting their name from the team members list. If a task is assigned to a person then the name of the person assigned to and the details of the tasks are seen. A task assigned once must not be available for choosing by others. As described earlier the tasks are visually represented. Example- a task can be represented as a sticky note with its description. The color of the sticky note represents the status of the task. This is also shown before a task's name whether a task is completed or in progress or uncompleted.

The tasks are set with priorities like high, medium and low. If the task is of high priority, it has to be completed first. The priority of a task can be used to filter tasks. There should be a feature to show dependencies between tasks. The project management tools provide linking of tasks to show dependencies between them. The tasks which are not linked by any other tasks are chosen for development or all the tasks on which a task is dependent on are implemented, is chosen for development.

A task can be made publicly available as open source file or can be made private to the team members only. The tasks are organized. One way of organizing tasks is maintaining the tasks in folders. The tasks are also shown in the hierarchy in a tree structure. The tasks and the subtasks are

shown in the tree showing the organization of the project. This feature helps to filter tasks based on status, priority, owner, project, date of creation of task. Tasks can also be ordered alphabetically.

A feature allows adding notes to task. It allows updating the status of a task. This feature is important when a task is submitted without completely finishing. The added notes on the incomplete task helps the person choosing this task, to understand its status.

There is a feature allowing attaching of multiple files to a task. Some tools have restrictions on the type of file attached otherwise any type of file can be attached.

Each task has certain time allocated within which the task must be completed. As and when a task is taken up, its start date and the due date are set. The tasks are time tracked and they can be billed for the number of hours worked on it. There are options to track billable hours and non-billable hours. Notifications are also sent to the responsible persons when the deadlines of tasks are nearing.

The feature for locking of task is to prevent multiple persons working on the same task. If a task is locked by a person, then only he can work on it. In many tools, the tasks when taken up, the details are filled like the name of the person taking up the task. In the task list, the task already taken up is shown with the person's detail who has taken up that particular task. This prevents many people working on the same task.

There are certain features of task creation which require communicating between the team members. The features which require the communication medium are described. A communication medium is required to provide the description of the tasks and assigning of tasks. The task description is sent via mail for the assignee or an option is provided in the tools for assigning of tasks to the members along with the description of the tasks. Some tools have task description document uploaded. Usually the task list is prepared in an excel sheet. The tasks are then assigned and the description is provided via mail. There are discussion forums to ask for online help if required in developing of task, chat rooms for instant help, blogs are another means. This kind of media helps in team collaboration.

Media like blogs, forums, online chat, message forums provides the team to know the status of the project and status of the work other team members is working on. RSS feeds help the team members to know the status of other team member's work.

Social networking like twitter helps in easy communication about tasks or to ask any help in developing any task. This kind of communication provides collaborative way of working environment and rapid assistance when needed. The tools provide an option for integration of such social networks. If there is any change is seen in the status of the project or project tasks, e-mail notification is sent to the team members notifying them about the change. It usually happens in the tools that when a task is assigned or taken up by someone, then notifications is sent to him/team members. Notifications are also sent to the team members when any new task is uploaded or a task is committed by any team member.

Version control helps in maintaining the different versions of files, documents, resources. These different versions are due to the changes made to them. For people coming in, current versions of the files are available. This feature helps in tracking changes made and there by helps in finding the errors when encountered.

The above described were the features for task creation seen in the project management tools. In the next section a project based on RUP is interviewed to learn the task creation features required in tools used in developing using STPD.

4.1.3 Interview with a project team to formulate the requirements for task creation

To list out the important features for creating tasks of a project to be developed using STPD model, a project similar to RUP based team was interviewed. The project is Arbo suite of Capgemini.

Arbo suite is a web based solution which provides organizations and the employers with the right information to manage the whole process from the first reported absenteeism until the employees get back to work. It is a long term (2-3 years) project which uses RUP like process for project development. They have proprietary tool for working culture and communication. The Project Manager of this team was interviewed in order to ask whether such a project can be developed using STPD model.

In this project, Project Leader assigns tasks to the team members. It is required to understand the application under development for a team member to work on any task of this project. There are dependencies among the tasks of the project. This takes long time (1-2 months) for understanding of the application being built and developing its task.

If the tasks of the project are defined in such a way that they do not require the knowledge of the application being built then the project can be developed using STPD. This is because cloud team members, who move in and out of a team at any time, develop the tasks. They should develop tasks in their bench time and according to the Project Manager of Arbo suite, this amount of time (bench time) is not enough to understand the application. It takes great time (when asked how much time- may be 1 year) for the Project Manager to define task such that their implementation do not require application knowledge was the opinion of the Project Manager of Arbo suite. To develop such projects using STPD depends on the Project Manager of such project teams.

4.1.4 Requirements for task creation in STPD

From the study of previous section, the requirements for task creation for RUP based as well as the Agile based project developing using STPD requires some of the task creation features. These are formulated as:

1. There must be an option to upload the defined tasks.
2. A task must only be taken up by a single developer. This requires locking of the task or providing details of the developer who chooses the task.
3. The tasks are dependent. There must be a feature for showing dependencies between tasks.
4. Version Control facility must be provided to keep track of changes.
5. Notifying the team members is not mandatory but good if this feature is provided.

These requirements for task creation are looked in tools in the next chapter for them to be selected to use for RUP based project developing using STPD.

4.2 Knowledge transfer

This section deals with knowledge transfer in STPD. The aim of STPD is effectively utilize the bench time. The people on bench form the cloud team in STPD. One of the characteristics of cloud team is, it is not fixed. The amount time a cloud team member can spend in STPD is not known. The cloud team members will leave STPD any time they get a billable assignment (project). Hence the knowledge pertaining to the project must be preserved and passed on till the end of the project.

Knowledge in reference STPD is, any piece of information related to project. Consider two scenarios

- Scenario 1: Consider, a cloud team working on a project in STPD. The project is in the initial phase. In this scenario, the aim of knowledge transfer is to make the cloud team understand the project quickly. This can be achieved by knowledge transfer means.
- Scenario 2: Consider, a cloud team is working on a project in STPD. If, for example, a cloud team member say 'A' leaves STPD without completing the task (task is the part of the project, he was working on) . Suppose a new member say 'B' enters STPD and takes 'A's place. In this scenario the aim of knowledge transfer is, 'B' should understand the work done by 'A' without actually contacting 'A'. This avoids the time spent in tracking down the people, talking to them and understanding their work.

In STPD people move in and out of the project. Hence the information about the part of the project they worked on has to be made known to the new person coming into the project.

The requirements on knowledge transfer are

- It must be possible to store the knowledge about the tasks.
- The knowledge must be easily accessible to new coming member of cloud team.

Knowledge represents the information. The list below represents the knowledge in the software development. The list presented below is not complete. The list contains only those knowledge (piece of information), which are applicable to all kinds of projects.

- Project Plan
  A project plan is a document, which guides the project execution and project control. The project plan at minimum contains project goals, project deliverables, project schedule, human resource plan, communication plan and risk management plan. Human resource plan includes the roles and responsibilities and the number of people required to carry out the project. Communication plan includes keeping people informed about the project, this can be done by progress report.
- Task description
  The tasks are the parts of project to be carried out, to complete the project. The tasks are of two types, dependent and independent tasks. The task description includes statements, which describe what has to be implemented. Apart from statements, it may include certain other details such as, dependency, priority etc.
- Requirements specification
  The requirements specifications include software product features, software system attributes, external interface requirements, and database requirements. Software system attributes include reliability, security, maintainability, portability, performance, availability and maintainability. External interface requirements include user interface, software

interface, hardware interface, assumptions and dependency. The requirement specification also includes product perspective, constraints product functions etc.

- Estimates
  There are majorly two kinds of estimates cost estimate and effort estimate. The cost estimates depend on many factors, some of them are man hours and tools.
  The effort estimate is amount of time required to complete the project in terms of hours. The effort estimate is also calculated during the project, it is for the amount of work remaining. The effort estimate largely depends on the resources available.
- Status information
  The status represents either the part of work completed or not completed. The status information can be used to represent the status of the task as well as the status of the project.
- Hours worked in order to bill in future
  In every project the hours worked is maintained for billing. This feature is required in STPD, as the business model gives certain rights to the members in future. It is necessary to keep track of numbers of hours worked by STPD members.
- Priority
  The priority here is in reference to the priority of tasks. The priority to tasks is given for various reasons, some of them
  - Certain tasks or requirements or features have more business value than others.
  - Because of the dependency between the tasks.
  - Core requirements or tasks have more value over other requirements or tasks.
- Other information
  This includes the architecture document, design document, marketing document, code document. These documents are required to complete the project.

There are several means of knowledge transfer, they are listed below

- Reports
  Reports come in various form from charts to documents. The charts provide the information about the current progress in terms of amount of work completed and amount of work remaining. The report in the form of document gives more detailed information about the tasks.
- Issue reporting
  Issue reporting must facilitate, reporting of bugs and reporting of impediments. Several tools provide this facility.
- Ability to add wiki pages
  Wikis are the simple web pages that contain information. It allows easy creating and editing.
- Progress bar
  The progress bar is used to show the progress of the task in the graphical form.
- Discussion list
  This is a type of mailing list used to send message to the subscribers of the list. In a similar way any subscriber may answer to the mailing list.
- Email
  It is a method of exchanging digital messages across internet.

- Links
  A link or a hyperlink is a reference to the document.
- Dashboard
  The Dashboard shows all the tasks in the project. In the dashboard view the tasks are usually grouped ex: completed tasks, incomplete tasks, blocked tasks etc. The other forms of dashboard include bug dashboard. Some tools provide dashboard for individual user, which shows up all the tasks, of the user along with the status information.
- Notification system
  A notification system informs the user of an event. The notification will be in the form of email or pop up box.
- Shared calendar
  Shared calendar is used to share the schedule, to set remainders and to invite other people to events on calendar.
- Social networking like twitter
  The social networking applications like twitter can also serve a means of knowledge transfer.

The different types of knowledge and knowledge transfer means is presented above. The following text illustrates how the knowledge can be transferred using the knowledge transfer means. The following suggestion is made with respect to cloud team.

- Project Plan
  It is not important for cloud team to know the entire project plan, because of two major reasons
  - The cloud team members are not available throughout the project.
  - The cloud team members will be working on specific tasks.

  Hence, any means can be used to express the project plan. Documents and Gantt charts are used to describe the project plan.
- Task description
  This information is important for cloud team members. The task description should contain few statements, which explains the work to be done. In the STPD environment, the task description must include the technical skills required to complete the task. The task description also shows the dependency with other tasks. Finally the task description must show the effort estimate. The task description can be linked to stories.
  Dashboards can be used for task description.
- Requirement specification
  Again the cloud team member is not required to know the complete requirement specification.
- Estimate
  The cloud team member needs to know the effort estimate and the remaining effort estimate of the task. This information is available in the task description. The cloud team member should update this information as the work progresses.
- Status
  The status information can be conveyed by progress bar or by using the social networking like twitter. The status message can be used to convey the progress.
- Hours worked in order to bill in future
  Many tools have this feature inbuilt.

- Priority
  The tasks can be assigned priority by means of ranking, tagging, use keywords like (high, low, medium).

  Every team member needs to self responsible to document the part of work carried out. This document must be attached to tasks. The task description can also contain useful links. A shared calendar like Google shared calendar can be used with team members.

5 RUP & model

This chapter gives a brief introduction to Rational Unified Process in section 5.1. Section 5.2.1 describes about similarities and dissimilarities between RUP and STPD. Section 5.2.2 describes RUP fitting into STPD and section 5.2.3 describes about the recommendations given for developing RUP based projects using STPD. The last section 5.3 describes about the open source tools studied in order to make a selection for RUP projects to be developed using STPD.

5.1 Introduction to Rational Unified Process

Rational unified process is a software engineering process. It is a disciplined approach to assigning and managing tasks and responsibilities in a development organization. The main objective of this process is to produce high quality software in estimated time and budget.

It is a framework for software development processes, which can be tailored according to the needs of the process used for a software development. It ensures that all the team members share a common language, process and view of how to develop software. It provides guidelines, templates and tool mentors for all the software lifecycle activities to the team members and delivers many best practices. Object oriented techniques are embedded and UML notations are used for the models built during the development.

The RUP process can be described in two dimensions or along two axes:

1. Horizontal axis – This axis represents time and shows the dynamic structure of the RUP – cycles, phases, iterations and milestones.

2. Vertical axis – This axis represents the static structure of the RUP process involving activities, artifacts, roles and workflows.

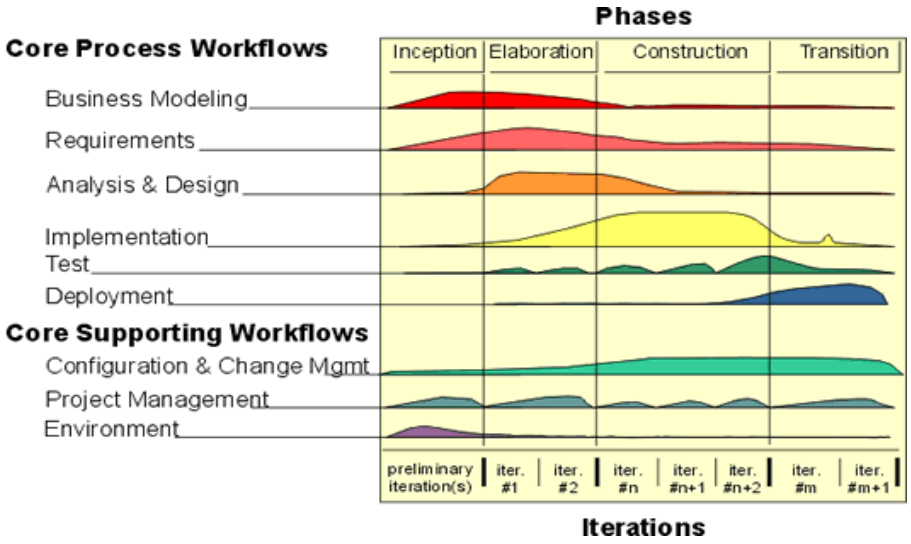

Figure: 3 The architecture of the RUP [27].

Dynamic structure of the RUP

The RUP process works in cycles, where each cycle involves work resulting in an improved version of the product. A cycle has four consecutive phases. They are

- Inception phase
- Elaboration phase
- Construction phase
- Transition phase.

Each phase has objectives called milestones which are set before a phase begins. A phase is carried out in much iteration until the milestones set are achieved. It concludes with milestones which involves making critical decisions and achieving the objectives of the phase.

Inception phase

The goal of the inception phase is to make common understanding among all the stakeholders on the objectives of the project to be developed. The objectives of this phase are:

- Establishing the project's software scope and boundary conditions.
- Identifying the primary scenarios of behavior
- Estimating overall cost and schedule
- Estimating potential risks

Elaboration phase

The goal of the elaboration phase is to stabilize the architecture of the system. This can be done by considering the significant requirements. The stabilized architecture provides a stable basis for the implementation carried out in the next phase, Construction phase. This phase is to stabilize the necessary things like vision, architecture, plan required for the next phase.

Construction Phase

In the construction phase all the remaining requirements are clarified and the development of the system is completed based on the stabilized architecture. The primary objectives of the phase are:

- to minimize the development costs
- To achieve adequate quality as rapidly as practical.
- To achieve useful versions as rapidly as practical [27].

The outcome of the construction phase is a product ready to put in hands of the users. It consists of the following:

- A software product
- User manuals
- A description of the current release [27].

Transition phase

The transition phase is that phase where the product, outcome of the construction phase is rolled out to the customers.

The transition phase includes the following

- beta testing of the new system
- training of users and maintainers
- roll-out to marketing, distribution and sales team
- refining by bug fixing, enhancing for performance and usability [27]

Static Structure of the RUP

The RUP process describes who is doing what, how and when by using modeling elements like

- Roles – the who
- Activities –the how
- Artifacts – the what
- Workflows – the when

Roles

A role defines the behavior or responsibility an individual or a group of individuals when assigned to them have to carry. It mainly describes the activities to be carried out by a person or the team playing this role. A role name itself describes the actual work to be carried. Example – Design Reviewer is the role whose responsibility is to review the design.

Activities

An activity is a specific role that an individual or a team is asked to perform. Each activity has clear purpose of performing it and is assigned to specific roles. An activity is associated with a role and affects few artifacts. It is used as an element of planning and progress. If the activity is large it is carried out in parts.

Artifacts

It is the information that is produced, modified or used by a process. These are produced during development of a product and are used to obtain the final product. These are used by the roles to perform activities and are the outputs of such activities.

Workflows

Workflows show a meaningful sequence of activities that result in the product and also the interactions of the role. They are required to understand the flow of the work to be carried out. In UML terms, a workflow can be expressed as a sequence diagram, a collaboration diagram or an activity diagram. There are nine essential workflows in the RUP process. They are

1. Business modeling
2. Requirements
3. Analysis and Design
4. Implementation
5. Test
6. Deployment
7. Project Management
8. Configuration and change management

9. Environment

1. Business modeling

The business modeling goals are:

- to understand the structure and dynamics of the target organization in which the system is to be deployed
- To understand the current situation of the target organization, problems and also to identify the room for improvement in the organization.
- To make sure that the customers, end users and developers have a common understanding of the target organization.
- To derive the system requirements needed to support the target organization [27].

2. Requirements

This workflow is to describe what the system to be built, should do. It describes the purpose of the system being built and can be used to make the customers and the developers agree upon the same requirements. This will also help in planning the iterations and estimating the cost and time needed to build the system.

3. Analysis and Design

The purposes of the analysis and design workflow are:

- to transform the requirements into a design of the system
- to evolve a robust architecture for the system
- To adapt the design to match the implementation environment, designing it for performance [27].

4. Implementation

The purpose of implementation workflow is:

- To define the organization of the code, in terms of implementation subsystems.
- To implement classes and objects in terms of components.
- To test the developed components as units.
- To integrate the results produced by individual implementers or team, into an executable system [27].

5. Test

The purpose of test workflow is:

- To verify the interactions between objects.
- To verify the proper integration of all components of the system.
- To verify that all requirements have been correctly implemented.
- To identify and ensure defects are addressed prior to the deployment of the software [27].

6. Deployment

The workflow is to ensure that the software product is available for its users. This workflow involves packaging, distributing to installing of the software. It also describes about providing assistance to users and involves planning and conduct of beta tests.

7. Project Management

The purpose of the project management workflow is:

- To provide practical guidelines for planning, staffing, executing and monitoring projects.
- To provide framework for managing risks [27].

8. Configuration and change management

This involves:

- identifying configuration items
- restricting changes to those items
- auditing changes made to those items
- Defining and managing configurations for those items [27].

9. Environment

This workflow is to provide software organization with both process and tools needed to support the development of the product. The purpose of this workflow is to provide the software development environment to the organization.

Best Practices

RUP process deploys the best approaches used in software development by many industries called the “Best Practices”. There are 6 such best practices. They are:

- Develop software iteratively
- Manage requirements
- Use component architecture
- Visually model software
- Verify software quality
- Control changes to software

Develop software iteratively

Software developing needs understanding the problem and requirements. These are not achieved at once. There is iterative approach carried out for increasing understanding of the problem through successive refinements. Over multiple iterations, there is incremental understanding of the software to be built. Rup supports the iterative incremental approach and addresses the risks at every stage of the lifecycle.

Manage requirements

The RUP describes how to elicit, organize and document the required functionalities and constraints. The use case and scenario approach is an efficient for capturing the functional requirements and there by these are important for the design, implementation and testing of the system.

Use component architecture

RUP process supports component-based software development. Components are modules, subsystems that represent a function. This process focuses on stabilizing the architecture before the allocation of resources and provides defining the architecture such that the components are reused.

Visually model software

RUP provides modeling. These help in visual representation of the software and capture the structure and behavior of the architecture components. Visual representations aid to communicate the different aspects of the software. They show how the different elements of the system fit together.

Verify software quality

Verifying the quality of the software built is an essential part of any software development. It is to be confirmed that the required software is built at the end. The quality of the software is reviewed with respect to the requirements based on the reliability, functionality, application performance and system performance. The RUP process assists in the planning, design, implementation, execution and evaluation of these tests types.

Control changes to software

This workflow description helps in keeping track and monitors the changes made to enable the successful iterative development. The changes to be made are also controlled. It brings a team together in automating integration and management [27].

This is a brief introduction to the Rational Unified Process. Rational Unified process is an IBM product. It is a framework which can be tailored according to the needs of the project development.

5.2 RUP fitting into STPD

In the previous chapters we studied about the STPD model and the RUP process. In this chapter, we describe in section 5.2.1 about similarities and mismatches between RUP and STPD model, in section 5.2.1 RUP fitting into the STPD model and in last section 5.2.3 certain recommendations for developing the RUP based projects using STPD are described. This chapter mainly describes fitting of RUP in STPD model.

5.2.1 Similarities and mismatches between RUP process and Spare Time Product Development (STPD)

This subsection describes the similarities and mismatches between the RUP and the STPD model. RUP process is more of a traditional way of developing software's while developing software's using STPD model is agile type of developing software.

The following are the similarities between the RUP and the STPD model.

1. The RUP process and the STPD model processes of software development are iterative process.

2. The team members do not know what the system is being built when the first iteration of inception phase of RUP begins. Cloud team of the STPD at any iteration may not know what the system is being built.
3. In the RUP, Project Manager takes the entire responsibility of developing the project like the Project Manager of STPD.
4. The Project Manager in the RUP process assigns the tasks to the team members while in the STPD; the cloud team members choose their tasks.
5. There is a project manager, architect, designer, integrator, tester, many developers, reviewer, review co-ordinator, manager etc in the RUP while there is one Project Manager, a Core Team and a Cloud Team. There are only developers in the cloud team of STPD.

The following are the mismatches between the RUP and the STPD model.

1. RUP process has a proper structure which is followed for the software development unlike the STPD where the process of carrying out software development has no definite structure.
2. In the RUP process, the work is assigned to the members. In STPD, members are not assigned; they choose their work based on their knowledge.
3. RUP is an exclusively planned and managed software development process unlike the STPD.
4. In RUP, information is shared on a need-to-know basis but in case of STPD, information is openly shared between team members.
5. The development steps involved in the RUP are study, definition, configuration, design, construction and delivery. The steps in the STPD model process are solution identification, code development and testing, code change review, code commit and documentation, release management.
6. Requirements formulating, understanding the product to be built are done during the development of the product. In the STPD, requirements formulating and understanding of the system are done before starting the development of the system by the core team.

5.3 Rational Unified Process (RUP) fitting into Spare Time Product Development model (STPD)

In the previous chapter, we studied about the Rational Unified Process. This chapter discusses about the Rational Unified Process fitting into the STPD model.

This Rational Unified Process fitting into the STPD model is examining to see how employees having no clients can be made to involve in the project development which are based on RUP. This is done to involve the people on bench in developing projects based on RUP/Agile process. This is one of the research questions, dealt in this chapter.

The RUP process is concrete and it is a framework for many software development processes. It cannot be modified for it to fit into the STPD model. As discussed in the previous chapter RUP has four phases and each phase has several roles playing. RUP process is kept intact and the roles of it are tried to map to the STPD model. It is examined, which part of RUP fits into the STPD model and how much of it fits.

The STPD has no phases. It has three elements while the RUP process has 40 roles playing in it. Each role in the RUP has various responsibilities to play. In the STPD model the Project Manager, Cloud Team and Core Team perform different responsibilities. If the responsibilities match then the role of

the RUP process is mapped to the matching team of the STPD. There are many roles in each phase of RUP, but only few of them are necessary to achieve the objectives of the phase and if all the roles of the RUP in the phases are mapped to the Project Manager, Core Team and the Cloud Team of the STPD then they are overloaded with the responsibilities. As a measure to this, only the roles required to achieve the objectives of the phases are considered in each phase.

The mapping of roles of RUP to those playing in the model is seen in the figure Fig.4. It can be observed from the figure Fig.4 that the roles in the inception, elaboration and transition phases are mapped to only Project Manager and the Core team of the STPD model but all the roles of the construction phase are mapped to all playing in the STPD model. From this it can be derived that only Construction phase completely fits into the STPD model.

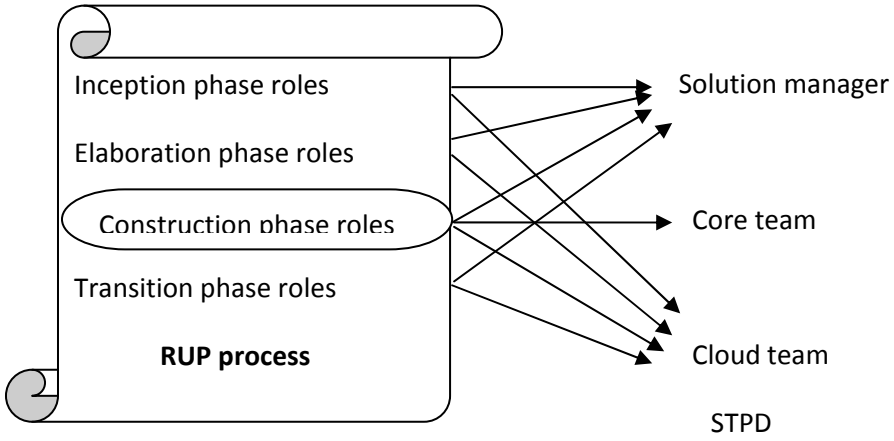

Figure.4 Mapping of roles of RUP to STPDs

As discussed in the previous Section, Construction phase involves development of the system. People constituting the cloud team can join hands in developing the system. They have no role in other phases as they require the knowledge of the system being built. Cloud team people move in and out of the project development any time, they develop the software parts which do not require the knowledge of the system. So this works only in the construction phase of the RUP and only the roles required achieving the objectives of the inception, elaboration and transition phases are mapped to the STPD model but all the roles involved in the construction phase are mapped to the STPD.

This chapter describes the roles in each phase of the RUP and mapping to the STPD model.

5.3.1 Roles in an iteration of Inception Phase of RUP

The goal of the inception is to make all customers; stakeholders come to an agreement on the objectives of the project. It is the phase where business case of the system is established. The scope of the project is studied and focuses on ensuring that the project being developed is valuable and feasible [27].

Inception phase iteration involves the following roles below

1. Business Architect/Business Designer/Business-Process Analyst

The business architect has the entire responsibility for the business architecture. This includes identifying and documenting the architecturally significant aspects of the business system that falls under the scope of the business modeling exercise.

A person must have good knowledge of the business domain and needs to be familiar with the tools used to capture the business models.

Business Designer is a person responsible for specifying the workflows of business use cases in terms of business entities and **Business-Process Analyst** leads and coordinates business requirements elicitation and delimiting the organization being modeled.

All the three roles require the same skills and they interact a lot. So it is more efficient to have a single person playing these roles [27].

2. Requirements Specifier

A person playing this role specifies and details the system requirements. He should have good communication skills, both in terms of expressing himself verbally and in writing.
A person who has good domain knowledge can play this role. He can specify the system requirements efficiently playing the role. Such a person is assigned for this role.

3. Process Engineer

Process Engineer plays an important part of management team of a software project. He is responsible for ensuring that the team members are doing their jobs. He supports Project Manager and responsible for all the process related aspects of the project, such as:

- Tailoring the process to match the specific needs of the project.
- Educating and mentoring project members on process related issues.
- Assisting the Project Manager in planning the project.

The Process Engineer role can be assigned to the person filling the Project Manager, in small teams. If the company is new to RUP based projects, this role is contracted to an expert outside the company [27].

4. Project Manager

This role plans, manages and allocates resources, shapes priorities, coordinates interactions with customers and users, and keeps the project team focused. The following skills are recommended to fulfill the Project Manager role:

- He should have experience in the software development processes and the domain of the application.
- He should be skilled in estimating the scope, planning of the project and resource planning, time and resource management, scheduling, estimating cost and managing budgets of the project.
- He should be skilled in analyzing risks and dependencies and also in making decisions.
- He is the leader in developing the project. So he should possess leadership qualities.

The project manager role can usually be combined successfully with other management-type roles, such as Change Control Manager, Deployment Manager, and Process Engineer. He can also take the development roles like Software Architect [27].

5. Software Architect

This role leads the development of the system's software architecture. The software architect has overall responsibility for making the major technical decisions, expressed as the software architecture. This typically includes identifying and documenting the architecturally significant aspects of the system, including requirements, design, implementation, and deployment views of the system.

If the project is large, a team constituting people expertise in different fields should perform this role. For smaller projects, a single person may act as both project manager and software architect. It is better to have these roles performed by separate people, in order to ensure that time pressure on one role doesn't cause the other role to be neglected [27].

6. System Analyst

This role leads and coordinates requirements elicitation. He should be an expert in identifying and understanding problems and opportunities and should have management and leadership skills. The role can be assigned to one or more staff members to perform this role in case of a large team while in a small team one member is assigned to perform this role and the role Test manager or role Deployment Manager [27].

7. Configuration Manager

This role function supports the product development activity managing the environment for the product development team. He ensures that the environment facilitates product review and also change and defect tracking tasks. He is also responsible for writing the plan and reporting. He is also responsible for writing the CM plan and reporting [27].

8. Integrator

This role is responsible for planning performing integration of elements. The elements implemented are tested and are delivered in the integration workspace by the Implementers. The Integrators then combine the tested implemented elements to produce a part of the system. This role requires

- Knowledge of the system or part of the system to be integrated and the interdependencies between the implementation elements
- Familiarity with integration tools [27].

9. Tool Specialist

This role supports the tools used by the project. This includes selecting and acquiring tools, configuring and setting them up, and verifying that they work.

Skills required are:

- An understanding of the underlying processes used by the project.

- General systematic analysis skills for comparing and selecting tools for the project.
- Needs communication skills as he/she is likely to be support contact point for the project members, on installation and other tool troubleshooting issues.

The Tool Specialist role can be assigned to one or more staff members are assigned to perform both this role and the Implementer roles in case of small teams. In large teams, a person can be dedicated to this role [27].

10. Review Coordinator/Reviewer/Technical Reviewer/Management Reviewer

The Review Coordinator role is responsible for managing the review process. The appropriate skills and knowledge required for one to play reviewer are:

- Planning and Organizational skills
- dispute resolution skills

Reviews can be generalized into two main categories: management reviews of project progress and technical reviews of project work products. A Reviewer provides timely feedback to project team members on the work products submitted.

Depending on the type of review, management or technical, a different skill set will be required:

- Technical: domain knowledge or subjective matter expertise appropriate to the work product being reviewed.
- Management: many years of business, technical, and software project management experience, excellent understanding of risk management principles and very good estimation skills.

Technical Reviewer provides timely and appropriate feedback on work products and Management Reviewer evaluates project planning and project assessment work products at major review points in project's lifecycle. A Reviewer either Management Reviewer or Technical Reviewer can play the Reviewer Coordinator role. He can inspect the review process. Review Coordinator role can also be taken by the Project Manager in small teams [27].

11. Technical Writer

This role produces end-user support material such as user guides, help texts, release notes, and so on. Playing the Technical Writer requires experience and/or training in technical writing. The Technical Writer role can be assigned in the following ways:

- One or more staff members are assigned to perform both the Technical Writer and Course Developer roles.
- One or more staff members are assigned to perform both the Technical Writer and System Analyst roles. This is because of the knowledge he has gained during inception and elaboration phases.
- This role is contracted out to a specialized Technical Writers. This is a common approach taken to deal with these responsibilities [27].

12. Test Analyst

This role identifies and defines the required tests, monitors detailed testing progress and results in each test cycle and evaluates the overall quality. The role also represents stakeholders who do not have direct or regular representation on the project.

Roles organize the responsibility for performing tasks and developing work products into logical groups. Each role can be assigned to one or more people, and each person can fill one or more roles.

The appropriate skills and knowledge for the Test Analyst role include:

- good analytical skills
- a challenging and inquiring mind
- understanding of common software failures and faults
- knowledge of the domain (highly desirable)
- knowledge of the system or application-under-test (highly desirable)
- experience in a variety of testing efforts (desirable)

The Test Analyst role can be assigned in the following ways:

- One or more test staff members are assigned to perform both the Test Analyst and Tester roles. This is suitable for small teams.
- One or more test staff members are assigned to perform the Test Analyst role only. This works well in large teams.
- One staff member is assigned to perform both the Test Analyst and Test Manager roles. This strategy is another option for small to mid-sized test teams.
- One or more staff members are assigned to perform both the Test Analyst and Requirements Specifier roles. This strategy is another option for small to mid-sized test teams [27].

13. Test Designer

This role leads defining the test approach and ensuring its successful implementation. This includes identifying the appropriate techniques, tools and guidelines to implement the required tests, and to provide guidance to the test effort on corresponding resources requirements. Roles organize the responsibility for performing tasks and developing work products into logical groups. Each role can be assigned to one or more people, and each person can fill one or more roles.

The appropriate skills and knowledge for the Test Designer role include:

- experience in a variety of testing efforts
- diagnostic and problem solving skills
- broad knowledge of hardware and software installation and setup
- experience and success with the use of test automation tools
- programming skills (preferable)
- programming team lead and software design skills (highly desirable)
- in-depth knowledge of the system or application-under-test (desirable)

The Test Designer role can be assigned in the following ways:

- One staff member is assigned to perform the Test Designer role only. This is a commonly adopted approach and is particularly suitable for large to mid-sized teams.
- One staff member is assigned to perform both the Test Designer and Test Manager roles. This strategy is a good option for small test teams.
- One staff member is assigned to perform both the Test Designer and Software Architect roles. This strategy is also an option for small test teams.
- One staff member is assigned to perform both the Test Designer and Test Analyst roles. This strategy is another option for small to mid-sized test teams [27].

14. Test Manager

This role leads the overall test effort. This includes quality and test advocacy, resource planning and management, and resolution of issues that impede the test effort.
Skills required are:

- general knowledge of all aspects of the software engineering process
- experience in a wide variety of testing efforts, techniques and tools
- people skills, especially diplomacy and advocacy skills
- planning and management skills
- knowledge of the domain, system or application-under-test (desirable)
- Experience programming or managing programming teams (desirable) [27].

5.3.1.1 Inception Phase fitting in the STPD

The inception phase of the RUP process involves many roles. All are not important to achieve the objectives of the inception phase. The teams involved in the STPD model are only a Project Manager, Core Team and Cloud team (people from bench), who are asked to work on the project.

The Project Manager of the STPD model is responsible for the development of the project. The Project Leader, one of the permanent members constituting the core team guides the entire team in developing the product and the people from bench who constitutes the cloud team, help in developing the product.

In the inception phase of RUP, requirements are formulated at the end of the iterations. The system to be built is understood. This phase does not involve any developing of the product. The people from bench forming the cloud team are the developers. The Cloud Team has no role to play in this phase.

The roles important to achieve inception phase objectives are only mapped to the STPD model because

1. The teams of the STPD model are overloaded with the work if they are assigned to multiple roles.

2. The roles like Management Reviewer, Review Coordinator, Reviewer, Technical Reviewer, Technical Writer, Test Analyst, Test Designer, Test Manager, Tool Specialist, Configuration Manager and Integrator are not considered for mapping as these roles play only in the development phase of the product. The roles are described in the previous section to learn which different roles participating in the inception phase. Among them which help in achieving the objectives of the phase are considered.

5.3.1.2 Mapping of roles of RUP in the Inception phase to the roles of the STPD

1. Business Architect role played by the Project Manager of the STPD

Before starting the development of any product, its business scope is studied. If business scope of the product has to be estimated then Project Manager of the STPD model formulates the business

scope of the product. If the business architecture has to be developed; Project Manager of the STPD can be assigned for developing the business architecture. In RUP framework these are responsibilities of the Business Architect. The skills required by Business Architect of the RUP process matches with the skills of the Project Manager of the STPD model. If required for this role, then Project Manager can play this role.

2. Project Manager role played by Project Manager of the STPD

It is important to plan the development of a product. The Project Manager of the RUP framework as described in the inception phase, plans, manages and allocates resources, keep the project team focused. These are the tasks done by the Project Manager in the Bench model. The skills required by a Project Manager of RUP match the skills of a Project Manager of STPD. In all phases Project Manager of the STPD model plays the Project Manager role of RUP. As the Project Manager of STPD is assigned to Business Architect and Project Manager role might be overloaded with work. So a person can be asked to assist him in playing his roles. If it is a small project of 5-10 people in a team then, there is no need for such supporting role.

3. Process Engineer role played by the Project Leader of the STPD

Process Engineer role is a support role as Project Leader in the STPD model. The Process Engineer makes sure that the team members are not hindered in doing their job. The Project Leader in the Core Team of the STPD model also guides the team throughout the development of the product. The Process Engineer is responsible for all the process related aspects like tailoring the process, educating the team members about the process, assisting the Project Manager in planning the project. Project Leader is the member of STPD core team who has the knowledge about the whole process being carried out and tasks carried out by him are similar to those carried out by the Project Engineer. Project Leader of STPD also mentors about tools assisting the Project Manager of STPD. He is appropriate to play Process Engineer role.

4. Requirement Specifier role played by a Core Team member of the STPD

In the STPD model, solutions which are frequently demanded by clients are built. If there is need for the Requirement Specifier then only this role is assigned to one of the members of the Core Team. One of the staff members is assigned as Requirement Specifier in the RUP. The Core Team of the STPD model has permanent members. If the size of this team is one then the developing team is small, Project Leader can play this role in that case. If the size is more than one, then a member of this team can play this role. The Project Leader and the members of the core team have in-depth knowledge of the system building. If the size is one, then only Project Leader can play this role, if needed.

5. System Analyst role played by Project Leader of the STPD

The System Analyst role is required only when there is a large project team i.e. if more than 10 people, to guide requirement elicitation. If the project team is big, then the Core Team size is also big. One member from the Core Team who has domain knowledge and through knowledge of the system is assigned for this role.

These are the required roles in the inception phase of RUP which are mapped to the roles of STPD. The roles like Software Architect, Tool Specialist, Reviewer, Review Co-coordinator, Technical Reviewer, Management Reviewer, Technical Writer, Tester, Test Manager, Test Designer, Configuration Manager, Integrator are not needed. So they are not mapped to any role in this phase.

5.3.2 Roles in an iteration of Elaboration Phase of RUP

The goal of this phase is to establish the software architecture that provides a stable foundation for design and implementation that is performed during construction. The following are the roles in an iteration of the Elaboration Phase.

1. Requirements Specifier

Refer Inception phase Role 2.

2. Software Architect

Refer Inception phase Role 5.

3. Capsule Designer

A capsule represents a specific pattern of class structure and composition which has proven useful in modeling and designing systems which have a high degree of concurrency.

This role designs capsules, ensuring that the system can respond to events in a timely manner, in accordance with concurrency requirements. The skill set required for the capsule designer role is similar to that of the role Designer; however, the capsule designer role requires more experience in handling concurrency issues [27].

4. Designer

This role leads the design of a part of the system, within the constraints of the requirements, architecture, and development process for the project. The designer must have a solid working knowledge of:

- system requirements
- the architecture of the system
- software design techniques, including object-oriented analysis and design techniques, and the Unified Modeling Language
- technologies with which the system will be implemented
- Project guidelines on how the design relates to the implementation, including the level of detail expected in the design before implementation should proceed [27].

5. Graphic Artist

This role creates product artwork that is included as part of the product packaging. To fulfill this role, there must be an expertise in the creative design field.

The Graphic Artist role requires experience or, at a minimum, training-in graphic artistry. A person playing this role needs to possess creativity, an understanding of marketing principles and an awareness of how to attract consumer interest. The Graphic Artist role can be assigned to staff members to perform the Graphic Artist role [27].

6. Database Designer

The database designer is responsible for defining the detailed database design, including tables, indexes, views, constraints, triggers, stored procedures, and other database-specific constructs needed to store, retrieve, and delete persistent objects. This information is maintained in the Data Model.

The database designer must have a solid working knowledge of the following:

- Data Modeling, Database design
- Object-Oriented Analysis and Design techniques
- System Architecture, including Database and System performance tuning, as well as hardware and network workload balancing
- Database Administration
- An understanding of the implementation language and environment [27].

7. Process Engineer

Refer Inception phase Role 3.

8. Project Manager

Refer Inception phase Role 4.

9. System Analyst

Refer Inception phase Role 6.

10. Tool Specialist

Refer Inception phase Role 9.

11. User-Interface Designer

This role coordinates the design of the user interface. This includes gathering usability requirements and prototyping candidate user-interface designs to meet those requirements.

The user-interface designer role is not responsible for implementing the user interface. Instead, this role focuses on the design and the "visual shaping" of the user interface [27].

12. Course Developer

This role develops training materials for training users on how to use the product. The Course Developer role requires experience-or at a minimum training-in course development. A person playing this role requires a good understanding of the product for which the training material is to be created, and preferably a good understanding of that product from the perspective of the target users' needs [27].

13. Implementer

This role develops software components and performs developer testing for integration into larger subsystems, in accordance with the project's adopted standards. The appropriate skills and knowledge for the implementer include:

- knowledge of the system or application under test
- familiarity with testing and test automation tools
- programming skills

It is possible for two persons to act as the implementer for a single part of the system, either by dividing responsibilities between themselves or by performing tasks together, as in a pair-programming approach [27].

14. Integrator

Refer Inception phase Role 8.

15. Change Control Manager

This role defines and oversees the change control process. They should be skilled in estimating cost and schedule impacts of change requests. They should be able to communicate effectively in order to negotiate scope changes and in order to determine how each change request should be handled and by whom [27].

16. Configuration Manager

Refer Inception phase Role 7.

17. Review Coordinator/Reviewer/Management Reviewer/Technical Reviewer

Refer Inception phase Role 10.

18. System Administrator

This role maintains the development infrastructure, both hardware and software. This includes installation, configuration, backup, etc. An individual taking on the role of System Administrator needs a good understanding of the specific hardware and software components used on a project, and the possible dependencies between these components [27].

The System Administrator role can be assigned in the following ways:

- One or more staff members are assigned to perform the System Administrator role exclusively. This is in case of large teams.
- One staff member is assigned to perform the System Administrator role in conjunction with another technical role such as the Implementer or Integrator role. This approach is suitable for small to medium sized teams.
- Each team member in the development team is assigned of responsibility for their own administration tasks [27].

19. Technical Writer

Refer Inception phase Role 11.

20. Test Analyst

Refer Inception phase Role 12.

21. Test Designer

Refer Inception phase Role 13.

22. Test Manager

Refer Inception phase Role 14.

23. Tester

This role conducts tests and logs the outcomes of his/her testing. Roles organize the responsibility for performing tasks and developing work products into logical groups. Each role can be assigned to one or more people, and each person can fill one or more roles [27].

24. Deployment Manager

This role leads planning the product's transition to the user community, ensuring those plans are enacted appropriately, managing issues and monitoring progress. A person playing the Deployment Manager role requires the following skills:

- Experience in deploying systems.
- Communication/Coordination in order to stay current with the status of the product development and communicate the needs of the deployment tasks to the rest of the organization [27].

5.3.2.1 Elaboration Phase of RUP fitting into the STPD

The elaboration phase fitting in the STPD model is same as the inception phase. The cloud team has no role to play; only the Project Manager and the core team have roles to play. The roles which are important to achieve the elaboration goals are mapped to the STPD model. All the roles of the elaboration phase of the RUP framework are not considered to avoid the workload on the people playing in the STPD model.

5.3.2.2 Mapping of roles of RUP in the Elaboration phase to the STPD

1. **Designer role played by the Project Leader or Core Team members of the STPD**

The Designer in the elaboration phase of RUP identifies and defines the design elements. The Project Leader has the profound knowledge about the system requirements, architecture of the system. Since he is the only permanent member of the team, he has the knowledge of the technologies using which the system will be implemented.

The skills required by a Designer in the elaboration phase matches the skills of a Project Leader. He can play the Designer role. If there are members in the Core Team who are equally capable as the Project Leader, then this role can be assigned to one of those members.

2. **Process Engineer role played by the Project Leader of the STPD**

In the elaboration phase too the Project Leader plays the role of Process Engineer, see inception phase fitting in the STPD.

3. **Project Manager role played by the Project Manager**

In this elaboration phase, the same skills are required for a Project Manager as in the inception phase. They are matched with the skills of the Project Manager of the STPD model. So Project Manager of STPD continues to play as Project Manager.

4. **Software Architect role played by the one of the members of the Core Team**

This role leads the development of the system's architecture. Software architecture is a full-time function, with staff permanently dedicated to it. For small projects, a single person may act as software architect like the Project Leader. But if the project team is big, say the Core Team size is large and then one of the members of the Core Team can play this role throughout the whole development. This person can be made permanently play this role.

5. **System Administrator role is played by the one of the members of the Core Team of the STPD**

The System Administrator takes care to install the software's and also ensures that the necessary tools are available. One of the members of the Core Team who has the knowledge of the required software's and the necessary tools required for developing of the project can be assigned for this role.

6. **Tool Specialist role played by the One of the members in the Core Team**

In STPD model, the Project Leader is responsible for most of the tasks in the development of the project. He is responsible to support tools used in the development of the project as like a Tool Specialist of the RUP process, if the project team is small. He selects required tools, configure and set them up for the project development. He also verifies that whether they work before the actual development of the project begins.

If the Core Team of the STPD model has members other than the Project Leader, then one of the members from this team who is skilled in analyzing and comparing tools in selecting the required tools can be assigned to play this role.

5.3.3 Roles in Construction Phase of RUP and mapping to the STPD

The main objective of the construction phase is to develop the system. This phase is responsible for the implementation of the components (modules) of the systems. At the end result of this phase is a deployable product, which can be rolled out for user feedback in the next phase i.e. the transition phase.

The roles involved in an iteration of the construction phase are below

1. Capsule Designer

Refer Elaboration phase Role 3.

Capsule Designer role played by the person playing Designer role in the elaboration phase

The person playing this role should have the knowledge of technical details in-depth and he also designs classes, subsystems. Capsule Designer role is similar to the Designer role but needs experience in handling concurrence issues. A person playing Designer role in the elaboration phase can play only if acquire knowledge in concurrence handling.

2. Designer

Refer Elaboration Phase Role 4.

Designer role played by the same person playing in the elaboration phase.

3. **Database Designer**

Refer Elaboration phase Role 6.

Database Designer role played by Database Specialists

The Database Designer role can be contracted to a team of Database specialists. This role requires experts in database. The tasks performed by the Database Manager vary depending on the product developed. It requires persons having solid knowledge in databases. It can be played efficiently by a database expert. In case of small team, it can be contracted to a specialist and if database management and administrating is needed in big projects, then can be contracted to a team of database specialist.

If there is no such requirement for specialist but person with database knowledge for creating databases, updating databases then one of the members of the Core Team who has knowledge can play this role instead of contracting to specialists. Even if there is no member in the Core Team with the knowledge of database; then one can be trained and assigned to this role, if the database designer role is not a big role in the development of the project.

4. Graphic Artist

Refer Elaboration Phase Role 6.

Graphic Artist role mapping

This ultimate intension of having this role is for the business of the product to be developed. But for one to be Graphic Artist, he should be creative enough. If any person either the Project Manager or any member from the Core Team is creative enough then he can play the Graphic Artist role. But this role requires one who has experience in graphic artistry. In that case, this role must be contracted out to a person expert in graphic artistry.

5. Tool Specialist

Refer Inception phase Role 9.

Tool Specialist role is continued by the same person who plays in the elaboration phase.

6. Deployment Manager

Refer Elaboration phase Role 24.

Deployment Manager role played by the Project Leader/One of the Core Team members assisting the Project Manager of STPD, if any.

Project Leader of the STPD model guides the development of the project and possesses the ability to plan ensuring of deployment on schedule. He has the communicating skill required to communicate with the development team. This skill is necessary in order to stay current with the status of the product development and to communicate the needs of the deployment tasks to the rest of the organization. He can play this role.

If at all it is possible and in order to avoid the burden of playing many roles on the Project Leader, if there is any team member in the Core Team who is assisting or equally capable as the Project Leader can take up this role.

7. **Change Control Manager**

Role Elaboration phase Role 15.

Change Control Manager role played by the Project Manager of STPD

The Project Manager of the STPD model has the knowledge of the entire project plan. He can estimate its impact and the cost involved in incorporating any change requests to the plan or scope of the project.

8. **Configuration Manager**

Refer Inception phase Role 7.

Configuration Manager role played by the member of the Core Team playing Tool Specialist role.

The person playing this role is responsible for installing, configuring and setting up the tools. In the STPD model, for small teams Project Leader performs these tasks, if he is the only member in the Core Team. This role can be assigned to the person playing the Tool Specialist role in the elaboration phase as both the roles are similar in setting of tools.

9. Implementer

Refer Elaboration Phase Role 13.

Implementer role played by the Cloud Team.

The Project Leader of the STPD model plays the detailed designer role. If the project team is small, Project Leader can also play the implementer role along with the cloud team members. One of the skills of an implementer is that knowledge of the system or application under test. Since the cloud team moves in and out of the project development team in the construction phase and they are not aware of the system being built. So it is the responsibility of the detailed designer, to design the classes and subsystems in such a way that they do not require the knowledge of the system under development. The cloud team members can implement such when descriptions of the tasks are provided.

10. Integrator

Refer Inception phase Role 8.

Integrator role is played by a number of dedicated members from Core Team. A number of persons from the Core Team can be dedicated for this role. This is because, these Core Team members are permanent and they have the in-depth knowledge of the system being built. This is required for integrating the subsystems. At the subsystem level, where implemented and tested elements are integrated, these members can perform but sometimes the implementers themselves can integrate the implemented tasks provided they have enough time and proper communication with other implementers.

11. Course Developer

Refer Elaboration Phase Role 3.

This role can be contracted out as it requires person who has experience in courseware development and must also have experience in delivering training to students.

12. Review Coordinator/Reviewer/Management Reviewer/Technical Reviewer

Refer Inception phase Role 10.

Mapping of Reviewers to the members of the STPD

The Project Manager can coordinate the review process. The Management Reviewer's process and the Technical Reviewer's process can be coordinated by the Project Manager. Management Reviewer's role can be played by Project Manager himself. He plans the project development and can take the responsibility of coordinating the reviewing its development.

This role is required only if the project team is very big. There can be many tasks development which need to be reviewed for the successful project completion. If the project team is small then no need of a separate role for coordinating. Project Manager and the Project Leader of STPD can coordinate their process.

Management Review tasks can be carried out by the Project Manager of STPD and the technical Review tasks can be carried out by the Project Leader. The Project Manager of STPD plans the project development, so he can efficiently play the role of Management Reviewer role and the Project Leader is responsible for the development of the project and has the knowledge of all the technical details of the project. He can play the Technical Reviewer's role. The tasks development can be reviewed by any cloud team member if reviewing of the developed task is made as a task. If the project team is a big team then there must be a coordinator to coordinate the process of Management Review process and the Technical Review process. In that case one among them can take the responsibility of coordinating the review process. If the project team is a small team then such a role is not required.

13. Technical Writer

Refer Inception phase Role 11.

Technical Writer role played by the (implementers) cloud team members or contracted out

This role requires technical writing skills. If the cloud team members (implementers) themselves possess such skills then they can write technical details about the tasks implemented otherwise this can also be contracted to a Technical Writer. The practice of documenting the tasks developed by the implementers must be carried out for the better technical documents of the project being developed.

14. Test Analyst

Refer Inception phase Role 12.

Test Analyst role mapping

The Cloud team members cannot play this role as they need the knowledge of the system or application under development. Cloud team members only implement the very low level tasks of the system, without having any knowledge of the system or the application under development. This requires the tasks uploaded must be defined in such a way that they do not require the knowledge of the system.

Project Manager of STPD mainly plans the project development and manages and coordinates its development. He has more of managerial knowledge of the project than the technical details of the low level tasks of the project.

Project Leader guides the entire team in developing the project. He provides assistance in technical details of the project to the team. If the developing team is small, he even plays the role of an implementer. He has the domain knowledge and about the system under development. If he possesses the experience in testing and if at all possible, then he can play the Test Analyst role.

If the Project Leader is over loaded with responsibilities, then one member from the Core Team can be assigned to this role. If there is no member in the Core Team experienced in testing, then one member can be trained and dedicated for this role.

15. Test Designer

Refer Inception phase Role 13.

Test Designer role played by Project Leader of STPD

Test Designer role can be assigned to the person playing the Test Analyst. It should be seen that the minutia of the Test Analyst role does not adversely affect the responsibilities of the Test Designer role. An individual can play both the Test Designer and the Test Analyst roles. Similar skills are required for both the roles and one or more such roles can be played by a person. The Test Analyst role identifies the test required and the Test Designer role designs the test approach to be carried out. These tasks are related and can be carried out single handedly.

It is better if a person from the Core Team is dedicated for the Test Analyst role than Project Leader playing. This is because if Project Leader is assigned then he has to play Test Designer role too. This leads to burdening the Project Leader role of STPD model. It is not realistic and efficient to do so.

16. Test Manager

Refer Inception phase Role 14.

Test Manager role mapping.

This role leads resource planning and management and resolving issues that impedes the test efforts. If the project team is a small team, then the person playing Test Analyst, Test Designer role can play this role too.

If it is a big project team for long term project development then assigning these roles to a single person is not efficient. Test Manager role can be carried out by the Project Leader, because he has knowledge of the system being built. He can identify the possible risks and plan the test approach. If Core Team is a big team then one member who has experience in testing techniques and tools can be assigned to this role.

17. Tester

Refer Elaboration Phase Role 23.

Tester role played by the Project Leader

The Tester is responsible for identifying the implement approach for a given test, setting up and executing the tests, logging the outcomes and verifying the results of the tests. If there are any execution errors during the test operation, then analyzing such errors and how to recover from such errors are the tasks of a Tester. The Project Leader can identify the test approaches for the tasks developed by the cloud team. He can then execute and verify these test approaches, if it is a small project involving 5-10 people. Otherwise this can be played by other members of the Core Team guided by the Project Leader.

18. User Interface Designer

Refer Elaboration Phase Role 11.

User interface Designer role mapping

A team of contracted members can participate in the design of the User interface and one highly skilled among them can play this role. The members must be trained and have experience in creative arts. User interface Designer coordinates the deigning of the user interface by the team members of the User Interface Design team.

This role cannot be played by either the Project Manager or by the Project Leader. Expecting Project Manager/Project Leader of STPD to possess many skills to perform roles even like User Interface Designer is hypothetical concept, practically such Project Manager/Project Leader are hard to find. User Interface Designer requires people with experience in arts. So this is contracted to experienced people.

5.3.4 Transition phase of RUP and mapping of its roles to the STPD

The Transition Phase is to ensure that software is available for its users. The Transition Phase can span several iterations, and includes testing the product in preparation for release, and making minor adjustments based on user feedback. At this point in the lifecycle, user feedback should focus mainly on fine tuning the product, configuring, installing and usability issues, all the major structural issues should have been worked out much earlier in the project lifecycle.

The roles involved in the transition phase are the same role playing in the construction phase. The roles like **Capsule Designer, Graphic Artist, Configuration Manager, Designer, Database Designer sand Course Developer** are not required. Since the transition phase is about developing, delivering and testing the built system, these roles have no much to play. All the other roles of construction phase are there in the transition phase too.

5.3.5 Recommendations to develop RUP projects using the STPD.

The following are the recommendations for developing a RUP project from the STPD Fig.1

1. The RUP process can be completely mapped to the STPD in its Construction phase only. This is because; it involves developments of components where the people forming cloud team (people on bench) can participate in developing these components. In all other phases it requires people to be associated with the project and possess the knowledge of the system to participate.

2. In the Inception phase, Elaboration phase and Transition phase only required roles for achieving the objectives of the phases are considered to avoid the burden on the three roles of the STPD.

3. Process Engineer and Project Manager are the roles in RUP played by the Project Leader and the Project Manager of the STPD in all phases of RUP.

4. Review Coordinator, Reviewer roles can be played by the same person.

5. Review Coordinator role is required if the project team is big. The Technical Reviewer or Management Reviewer can play this role.

6. Graphic Artist role is usually contracted to a person skilled in artistry. If any member is skilled in artistry he can be assigned to this role provided he is not overloaded with the responsibilities.

7. The roles like Technical Reviewer, Technical Writer, Test Analyst, Test Designer, Test Manager, and Tester can be considered only in the Construction and Transition phase. These have significant role only in these phases and to prevent Project Manager and Project Leader of the STPD from overloaded with work, they are not considered in the Inception, Elaboration phases.

8. Course Developer role can be contracted out as it requires in-depth experience in delivering training to students and also requires to understand the different teaching styles.

9. Capsule Designer and Designer can be played by the same person, however capsule Designer requires experience in concurrency handling issues. Designer can be trained in concurrency handling.

10. The components/tasks to be developed by the Implementers in the Construction phase must be made such that the Implementers can develop the components without requiring any knowledge of the task/component being developed or the system under development. This is most important as the Implementer role is played by the people from bench, who move in and out of a project without any knowledge of the system.

11. It is the responsibility of the Project Manager of a RUP based project to define the tasks which can be implementable without requiring the knowledge of the system. If this is possible then RUP based projects can be developed using STPD without any fuss if proper tooling is provided. Project Manager requires possessing specialized skills for defining tasks.

12. If the tasks/components must be defined such that they are implementable in the spare time of the employee on bench. If the tasks are big then they must be informed to responsible person for them to be broken into small subtasks tasks.

13. If the subtasks broken are small then they can be integrated by the Implementer who implemented them. Implementers can play Integrator role in subsystem level. At the system level it is the dedicated persons from Core Team assigned for integrating the components.

14. The core team members must contain experts from different fields because many different kinds of roles are mapped to its members.

15. Project Manager of STPD must be assisted by some staff members, as he has many tasks to perform in the STPD. He needs staff to assist in preparing reports, as preparing report is one of the tasks of the Project Manager.

16. System Administrator role can be played by the Project Manager of STPD but to prevent overloading of work, this can be assigned to one member from the core team.

17. Most of the roles of the RUP process, the skills required to perform these tasks match with the Project Leader or the Project Manager of the STPD model. But assigning then all possible roles would be unrealistic. They cannot efficiently perform. Core team size is at least one.

    If the core team size is one, then he is Project Leader and team is small (3-5 members). In such small teams he can play more roles. But if the core team size is more than or in case of big projects then core team must be big. It should at least contain 5-10 members other than the Project Leader as, 4 roles in the elaboration phase require to be assigned to the core team members and also some members from the core team are dedicated for Integrator role.

18. If the core team is not big then some of the members from cloud team must be hired and must be dedicated to work with the core team. As they are developers they can play Tool Specialist, Technical Reviewer roles if trained.

The RUP based project development using STPD is not commendable as it requires people to be associated with the project. It involves many roles than the roles in STPD which is difficult to play all the roles by the three roles of STPD. It also requires the tasks to be defined such that implementing them does not require the knowledge of the system being developed. If this possible then the RUP projects can be developed using STPD without much fuss. The above are some of the recommendations for developing RUP projects using STPD.

5.4 Open Source Tools

This chapter describes about the open source project management tools studied. The section 5.4.1 is introduction to open source tools and tools studied; section 5.4.2 describes the observations made in the studied tools and section 5.4.3 certain recommendations made for tooling used for developing projects using STPD.

5.4.1 Introduction to Open Source Tools

Open source project management tools aid in developing projects. They provide web based access and the tools required for developing the projects. It is difficult to manage projects which involve many people and tasks. As the project development proceeds, managing the project, communicating with the people of the project, sharing and distributing the project documents becomes tedious job. Open Source Project management tools are used in managing the project development. They

provide facilities for planning, organizing and managing the resources to achieve the project goals and also have many communicating media like forums, blogs and online-chat.

There are numerous project management tools freely available with many features and providing many facilities. There are tools supporting different software development processes. There are many different project management tools according to the requirements of the projects to be developed. The tools also intend in collaborative project development. It is required when people are located in geographically different places and are working on the same project. These tools ensure that the project is developed with good quality without requiring the people to be physically in the same room or close to each other.

The open source project management tools are studied with the perspective that they can be used in developing the RUP based projects in the Construction phase. These RUP based projects are developed in an environment where the development team involves the people of the STPD. The tools are required for the Construction phase of the RUP. This is because only Construction phase of RUP process can only fit in the STPD model. The Cloud Team of STPD can only implement tasks of the projects in the Construction phase and Core team and the Solution Manager continue to play in this phase also. So the tools are studied for the Construction phase of RUP, as STPD model can only be used in the Construction phase of RUP.

In the chapter 4 Task Creation and Knowledge transfer, we have formulated some of the important requirements for task creation of projects developed using STPD and in the same chapter 4 important requirements for Knowledge transfer in STPD model are formulated. These features must be available in the tools.

The chapter 4 also describes the important techniques for rapid knowledge transfer in the environment like STPD. Among the techniques mentioned, the following are more suitable for development in the Construction phase of RUP

1. Twitter like application integration

This allows the team members located in geographically different places can ask and get help related to developing of the project, instantly by tweeting about it.

2. Dashboard

A colorful dashboard gives the complete status of a project in a view. This helps the team members to learn the status of the project and its tasks rapidly.

3. Email

If any confidential matter has to be discussed with a particular person then this feature is required. In RUP based project development, if a task is big and has to be broken down into its smaller subtasks then a mail can be sent to the Project Manager requesting for making smaller subtasks of that particular task.

4. Progress bar

The progress bars show the status of the tasks. They represent how much percent of the task is completed/uncompleted.

Many project management tools were studied but all of them do not possess the requirements for task creation and knowledge transfer. Some of them which provide most of the features for task creation and knowledge transfer are considered. The following are the tools studied in order to use in the Construction phase of RUP based projects.

1. Zoho projects

Zoho feature is a feature-rich project management, collaboration and issue tracking tool that allows teams to collaborate and get things done [26].

This tool has features for defining tasks, task list, showing dependencies between tasks and also tasks once chosen can only be worked upon it by the person chosen it. It also has version control feature which is necessary for finding and fixing the errors when encountered and also tracks issues. The tool also has feature for billing people based on their working hours. The tool has the features of task creation.

In the chapter 4 we studied the features required for rapid knowledge transfer in STPD model. The Zoho tool has some of the required features discussed in chapter 4 for knowledge transfer. There is collaborative dashboard which gives one stop view of all the project related activities. It also has feature for sending mail, twitter integration, progress bar and many other forms of communication like online chat rooms, discussion forums.

This tool has all the required features for the project development using STPD model. The tool is freely available for 1 project with limited features; otherwise it is $699/year for unlimited number of projects and users with all the features.

2. Central Desktop

A Central Desktop is a wiki based software service. Central Desktop provides an Industry Leading SaaS-Based Social Technology Platform for Business Teams. Organizations are served by Central Desktop in technology, media, marketing and communications, professional services, architecture and design, manufacturing and many other industries [22].

This tool provides an option for creating tasks and task lists. The tasks can only be worked by one person at a time. The changes made to the files, which made, what changes made can be maintained by maintaining the versions of it. Version control facility is provided. This tool does not provide option for showing dependencies. But priority levels can be set for the tasks.

In relation to knowledge transfer features, this tool has emailing, twitter integration facilities. It also provides facility for creating wiki pages and blogging. This tool has most of the features but does not have option for showing dependencies. The tool pricing depends on the number of members and the number of workspaces. It is free for 2 workspaces and 5 members/workspaces [2].

3. Hyperoffice

HyperOffice offers integrated web based messaging and collaboration solutions to small and medium sized companies. This tool has the feature for creating tasks, task lists. The members of the team can work on the tasks using versioning, notifications. There is gantt-chart view of the project which is easy to get an overview of project status, task interdependence, sequencing and dates through it [28].

Hyperoffice offers full range of collaboration features through right on mobile browsers. E-mails can be sent; contacts, calendars and tasks can be pushed to mobile device instantly and can be kept with constant sync with the Hyperoffice. This is one of the distinguishing features. The Gantt-chart view of the project shows the status of the project. This feature can be an alternate for dashboard. The tool has most of the features. The price of the tool for Collaborative suite with e-mail is $527.42/year for 5 users. The price depends on the number of users. This tool is a costly affair if the project developed is a large project.

4. ProjectSpaces

ProjectSpaces is a simple, secure and powerful online workspace and extranet tool. It helps project teams, workgroups to easily share and collaborate. It helps to manage documents, coordinate projects and activities and share knowledge and information [29].

This tool has feature for uploading tasks, locking of the tasks. It also provides maintaining the versions of the tasks. It does not have feature for showing dependencies between tasks. There is no dashboard but there is Gantt chart for providing the complete view of the project status and tasks. If any changes is made to files e-mail notifications are sent. It provides facility for e-mailing. It does not provide option for integrating twitter like social networks. The tool is available freely with limited features. The price of the tool varies depending on the number of projects. For five projects and unlimited number of users, it is $79/year. For 1 project and 25 users, it is freely available.

This tool is an option if there is requirement for only one small project development at a time and e-mail and RSS feeds are the modes of communication. This is because it does not provide option for showing dependencies.

5. ActivCollab

ActivCollab is a project management and collaboration tool that can be set up on a server or local network. It is used to share, distribute and collaborate with the project team. It provides platform for planning, progress tracking and communication [21].

This tool has task creation features for uploading any number of projects. The tasks can be tracked and have assignee, priority, start date and due date set. It has a graphic indicator which shows how a project is progressing showing the tasks completed and the tasks to be done. There is facility for version control. Files can be uploaded and changes made to them can be kept track of them.

As it has assignee attribute to be set for each task, it ensures that only one person can work on a task. It has no facility for showing dependencies between tasks.

For communication between the team members, it has discussion forums, e-mailing and mailing-list facilities are provided. It also provides an option for integration with other applications. The price of the tool is $499/year.

6. EasyProjects .Net

Easy Projects .Net is a web based project management tool, making team collaboration hassel free and straightforward. It is highly sophisticated system based on the Microsoft .NET technology and MS SQL server. These technologies provide with scalable and flexible solution to manage and track projects of any complexity level [30].
The tool provides many features related to task creation. It allows any number of projects and provides project templates and also allows changing assignee, status and priorities of a project. There is a Gantt-Chart representation of the project allowing creating unlimited number of task subtasks and provides an option for showing dependencies between tasks/subtasks. It also provides global role-based and project level permissions. It also allows creating issues by sending mails. These mails are parsed and automatically issues are added assigned to the dedicated address.

The features related to knowledge transfer are dashboard and e-mailing facilities are provided. It also has discussion forums. The price of the tool is $648/year for unlimited number of projects and users. For one user and one project this tool is free but with very less storage 100 Mb.

7. Goplan

GoPlan is an online project management and team-based collaboration tool. It has an intuitive user interface. It is being used by companies like Mozilla, Boxee etc. It has many salient features. The tasks and task lists can be created. It has no option for locking or showing dependencies between tasks. But provides version control and issue control facilities. These are the features provided which are important for task creation of project developed using STPD [31].

The tool has discussion forums for communicating with the team members. This is how information is conveyed to the team members. This is suitable for small projects. There is a free plan that lets to manage 3 projects with 2 users and 2 collaborators. The price of this tool is $80 USD/month [7].

8. Gantt project

GanttProject is a cross-platform desktop tool for project scheduling and management. It runs on Windows, Linux and MacOSX, it is free and its code is opensource. It is not web based tool. This tool is used to schedule and manage the tasks of a project [32].

Gantt project has important features required for task creation of a project developed using STPD. It allows creation of any number of tasks of a project. A task of a project is sometimes dependent on other tasks. It provides an option for showing the dependency between tasks. This feature is important for the projects which are RUP based and many open source project management tools do not provide this feature. Tasks are assigned and if a person is assigned to more than one task, it alerts. As it is not web based, tasks cannot be possible to choose. It also shows the status of tasks and the project by progress bars. It not only schedule and manage projects but also manage

resources. It has resource chart. This tool does not provide features for knowledge transfer. It does not provide facilities like sending e-mail, time tracking of tasks.

5.4.2 Observations about tools

The various open project management tools were studied in order to use them for the Construction phase of RUP. There were some common characteristics observed among them. These are listed out below

1. All the required features for task creation and knowledge transfer are difficult to find in one tool.
2. There are many open source project management tools with features for communication/knowledge transfer but they do not provide much features related to task creation.
3. Showing dependencies between tasks is one of the important and required features for a project based on RUP to be developed using STPD. This feature is not provided by many open source project management tools.
4. Another important feature for task creation in STPD is locking of tasks. This is not found in many tools.
5. There are tools with required features for knowledge transfer but not with all the required features for task creation.
6. There are task management tools with all the required features but are not web based. They do not provide facility for storing files and communicating among team members. Eg: Gantt Project, WorkBench.
7. The tools are expensive. They are also available for free but with less features and limited resources.

5.4.3 Recommendations for selecting tool used in developing RUP based projects using STPD

In the previous sections we studied about the open source project management tools and also made some observations about them. Based on the study and observations about the tools some conclusions are made as recommendations for using a tool for the Construction phase of RUP. The following are some of the recommendations for using a tool for developing a RUP based project using STPD:

- Zoho project and EasyProjects.Net can be used. They provide most of the features required for developing RUP based projects using STPD. They are expensive but based on the number of projects and users an appropriate plan can be chosen.

- There are many open source task management tools like Gantt project, Workbench etc which are not web based and provide all the required features for task creation. They do not provide facilities for knowledge transfer and also storage facility.

  If these tools can be extended such that they are made web-based with knowledge transfer feature like sending e-mail or blogging then can be used for developing RUP based projects using STPD.

- Agile Dashboard project explained in Chapter 3, uses a tool for developing projects. This tool has a dashboard and also provides version control facility. It provides blogging as a means of communication between the team members. This tool has features for task creation and if it is extended such that twitter like social networks can be integrated then this tool can be used for developing RUP based projects. This requires only extending the tool and does not involve any cost.

The recommendations are made by looking for the requirements for task creation and knowledge transfer in the tools. The recommendations also involve extending of the tools Capgemini is using already.

6 Agile & Model

In this chapter section 6.1 describes the Agile software development methodologies. Section 6.2 describes how much scrum fits/ misfits STPD. Section 6.3 presents recommendations on process. Section 6.4 presents tools survey. Section 6.5 gives recommendation on tool.

6.1 Introduction to Agile

This section presents Agile software development methodologies. Scrum process is recommended to use with STPD. All the Agile software development methodologies presented in this section are studied thoroughly from literature. These methodologies presented are based on the understanding from the study.

Agile software development is used to refer a set of software development methodologies. Some of the Agile methodologies are Extreme Programming, Scrum, Feature Driven Development, Dynamic Systems Development Method, Agile Modeling, and Pragmatic Programming [2].

1. Extreme Programming

The main aim of Extreme Programming is customer satisfaction. It addresses the issue of changing requirements. In Extreme Programming the product is developed in short development cycles. Extreme Programming is the most documented methodology of all the agile software development methodologies. Extreme Programming is best described with the phases, roles and best practices.

Extreme Programming consists of six phases [2].
Exploration phase: In the exploration phase, the customer writes the story cards for the first release. The team familiarizes itself with the tool, technology and practices to be used in developing the system (project). Each story card written by the customer represents a feature to be implemented in the first release. It is during this phase, the team explores the possible architecture for the system, by building prototypes. The exploration phase lasts for few weeks to few months.

Planning phase: In the planning phase, the stories are prioritized. A set of stories/features are selected for the first release. Effort estimate to implement each story is made and a schedule is drawn. Usually the schedule is such that, the first release is two months after the start date (start date of implementation). The planning phase spans for couple of days.

Iteration to release phase: This phase itself includes a number iterations. Each iteration spans from one to four weeks. In the first iteration the architecture of the system is built. The customer decides on the stories to be implemented in the iteration (he does this for every iteration). At the end each iteration functional tests are carried out. It is at the end of last iteration the system is built.

Production phase: In this phase, the system is tested and performance is checked. New changes may be identified and a decision has to be made, whether to incorporate them in the current release or postpone to next release. In case the changes need to be incorporated, the time span of iteration is reduced to one week. If the changes are postponed, they are documented. At the end of production phase the system is released to customer.

Maintenance phase: When the system is released to the customer, development team is required to respond to the changes suggested by the customer. After the first release the team will continue to work with the next release. To respond to the customer requests, the team organization is changed.

Death phase: This is phase when the customer has no more stories. It means that the system satisfies the customer needs. In this phase the final documentation of the system is written.

Extreme Programming consists of seven roles. The roles and responsibilities of each role is described below.
Programmer: Programmers writes the test and code. The objective is to keep the code simple.
Customer: The customer writes the stories, functional tests and decides when each requirement is to be implemented. The customer also assigns priority to each story.
Tester: The tester helps the customer to write functional tests. The tester runs the test regularly and broadcasts the results.
Tracker: Tracker keeps track of progress and gives feedback. He also evaluates whether the goals are in reachable limit.
Coach: Coach is the person responsible for the entire process. He must have deep understanding of Extreme Programming.
Consultant: Consultant is an external member possessing the necessary technical knowledge.
Manager: Manager makes the decisions. He achieves it by communicating with team members.

The following are the practices in Extreme Programming [34].
Whole Team: According this practices, it is required that all the team members must sit together in one place. This helps in open communication among the participants.

Planning Game: Both customer and developer participate in the planning game. The practice deals with the planning methodology. In the planning game, two questions are addressed. What features are implemented in first release? What features will be implemented in the next release?

Customer Tests: In Extreme Programming, the testing of the system is based on the customer scenario. This ensures the customer that the system is satisfying the customer needs.

Small Releases: This practice suggests small parts to be implemented in every release. Any customer request for change is incorporated easily.

Simple Design: In Extreme Programming, the aim is to keep the design of software system simple. The design is incremented by continuous testing and improvement. The design should be made with respect to current release.

Pair Programming: The success of Extreme Programming comes from Pair Programming. In Pair Programming, two persons sit together and work on one computer. They work on the same design, algorithm, code and test. When one of them is coding the other programmer reviews the code, to identify any syntax errors, wrong implementations, spelling mistakes etc. The two programmers exchange the roles (reviewer and code developer) often. Pair Programming increase the quality of the code developed without having a negative impact on the release date.

Test Driven Development: Extreme programming works on the principle of feedback. Therefore the system built must be tested thoroughly. The developers are required to write white box test cases. The Test Driven Development proposes to perform this testing on every module before the actual code integration.

Design Improvement: Extreme Programming requires continuous design improvement. This practice is also known as refactoring. Design improvement involves removal of duplication, lowering of coupling, increasing the cohesion. The continuous design improvement results in the simple design.

Continuous Integration: According to this practice, it is required to break the whole system into small modules. These modules are built and integrated. The suggested effort for every module is not be more than one day. The continuous integration requires single source repository. Code integration is done every day. Continuous integration reduces risk.

Collective Code Ownership: In the traditional software development methods, only the developer of the code has the right to change the code. In Extreme Programming all the code is owned by all the members of the team. This means that any pair of programmers can make changes to any code. In this way the defects are reduced and code quality is increased.

Coding Standard: According to this practice, it is required that every team member must follow the same coding standard. This makes the code easily understandable by all members of the team.

Metaphor: This practice proposes that the team should have common understanding or picture how the system works.

Extreme programming cannot be employed in STPD because of the following reasons.

- Exploration phase lasts from for few weeks to few months and planning phase spans over couple of days. This means actual development of system begins only after few months.
- Planning phase lasts for couple of days. This is not suited for cloud team because more time is spent in planning.
- After the first release, every iteration begins from planning phase. This is not suitable for STPD for the same reason mentioned above.
- The major reason being Pair Programming. Pair Programming demands two developers working on the same code. This is not possible in the STPD. The cloud team members are from bench. They may leave STPD any time they get a billable assignment. Consider, a scenario, two developers are working on a piece of code. And one of them leaves from the project. In this situation if a new developer takes his place, more time is spent in things like, understanding the code, getting adjusted with the other programmer, getting to know the pace of development rather than developing the code. The problem is not only to the new developer coming in but also to existing developer. Pair Programming is a skill it will take time to learn. Therefore the conclusion is Extreme Programming is not suitable for STPD and must not be used.

2. Feature Driven Development

Feature driven development is an iterative and incremental development methodology. In this method the features to implement are discovered first and then feature by feature implementation takes place [2].

Feature Driven Development methodology consists of four phases

Develop an overall model: Domain experts, chief architect and team members are involved in these phases. This phase does not address the issue of requirements gathering. The domain experts have knowledge about the requirements, context and scope of the system to be built. Documents containing use cases and functional requirements are available in this phase. Domain experts conduct a walk through and build the high level description of the system. This high level description is presented to the team members and chief architect. The overall domain is now divided into different domains. Again a detailed walk through is conducted for each domain by domain experts.
The development team is broken down to small groups. After every walk through a group works together to produce object model. This is done for every domain area. The groups then discuss and decide on objects models for each domain. At the end an overall model is produced for the system.

Building a feature list: Walk through, object models and requirements documentation is required in this phase. They help to build the feature list for the system under development. The development team presents client valued functions as feature list. The functions are presented for every domain area. These functions are called major feature sets. The major feature sets are again divided into feature sets. These new feature sets are domain specific. Finally the list of features is reviewed by sponsors and users.

Plan by feature: The Plan by feature includes the following activities.

- Creation of a high level plan. The high level plan consists of features ordered by priority and dependency assigned to chief programmers.
- The classes identified in “develop an overall model” are assigned to individual class owners.
- Schedule and major milestones are set for feature sets.

Design by feature and build by feature: A small set of features are selected. To develop these selected features, teams are formed by the class owner. Design by feature and build by feature are iterative procedures. The selected features are built in iterations. Each iteration spans over couple of days to two weeks. The teams formed work concurrently and build the set of features assigned to them. The tasks in iteration are design inspection, unit testing, coding, integration and code inspection. After the completion of iteration, the completed features are given to main build. The teams continue with new set of features.

Feature Driven Development consists of 13 roles. The roles and responsibilities of each role are described below.

Project manager: He is the administrative and financial leader. He has the ultimate powers to decide on scope, schedule, staffing of project. His responsibilities are

- To protect the team from outside distractions.
- To keep the team working, by providing appropriate working conditions.

Chief architect: The chief architect makes final decisions on any kind of design issues. The overall design is given by chief designer. The chief designer leads the design sessions. If required, the role of chief architect can be divided into domain architect and technical architect.

Development manager: He leads the development activities. He solves any conflicts occurring within a team. He is responsible for solving resource problems.

Chief programmer: The chief programmer is an experienced developer and participates in requirements analysis and design. The chief programmer acts as a leader to small teams and helps them in analysis, design and development of features. The chief programmer selects the set of features for the next iteration of "design by feature and build by feature" phase. The chief programmer identifies the class and class owners for the next iteration. The chief programmer gives the weekly progress report. The chief programmer along with other chief programmers solves technical and resourcing issues.

Class owner: A class is assigned to the class owner. The class owner is responsible for developing the class. The chief programmer guides the class owner in the tasks such as, designing, coding testing and documenting. In an iteration only the class owners whose class is present the feature (implemented in the current iteration) are involved.

Domain expert: The domain expert is a user or client or sponsor or business analyst or a combination of these. The domain expert has the knowledge of how the requirements should perform. He gives this knowledge to developers, so that the right system is built.

Domain manager: He the leads the domain experts and resolves any difference of opinion.

Release manager: The release manger reviews the progress reports submitted by the chief programmers and holds short meetings with them. He controls the progress of process. He submits the reports to project manager.

Language lawyer: Language lawyer is a team member possessing the in depth programming knowledge or thorough knowledge in technology. Language lawyer is required, if the team is dealing with new technology.

Build engineer: A person responsible for setting up, maintaining and running the build process. He is also responsible for managing version control systems and publishing documents.

Toolsmith: Toolsmith is a role for building small tools for development teams, test teams and data conversion teams in the project.

System administrator: The system administrator configures, manages and troubleshoots the servers, developing and testing environments and network of workstations. He can also involve in productionizing.

Tester: The tester tests the system to ensure that it meets the requirements. The tester could be independent team or part of the project.

Deployer: The work of the deployer is to convert the data to a format, which is required in the new system. Deployer also participates in new releases. Deployer could be an independent team or part of the project.

Technical writer: Technical writers prepare the user documentation. They could be separate team or part of the project.

The following are the practices in Feature Driven Development

Domain object modeling: This involves analyzing of application domain model and producing object models.

Developing by feature: This practice proposes to build the system in number of iterations. In each iteration a set of features is built. The feature selected for each iteration is based on client's decision.

Individual class (code) ownership: This practice proposes that for every class defined, there must be one person designated as owner. This ensures consistency and increases performance.

Feature teams: This practice proposes that small teams must be used Feature Driven Development.

Inspection: This practice proposes to use defect detection mechanism for the entire process.

Regular builds: This practice ensures that, there is always working software available. The software in the previous build forms the baseline for the current build.

Configuration management: This practice states that all the previously completed versions must be kept track of.

Progress reporting: This practice states that there must be reports available to people at all organizational levels.

Feature Driven Development cannot be employed in STPD because of the following reasons.

- In this first phase "develop an overall model" walk through is conducted twice. This consumes time. The amount of time a cloud team member can spend in STPD is unknown. Thus the available time should be used in development activities.
- The list of features to be implemented is determined in the second phase. STPD is characterized by people moving in and out of the project, which means that amount of time spent in software development is not known. A ready list of features would be a better option for STPD model. Hence the recommendation is not to use this methodology for STPD.
- The planning begins in third phase. Again the same reason holds. If this method is employed more time will be spent in other activities than development.
- Feature driven development method has many roles.

3. Dynamic Systems Development Method

The fundamental idea of Dynamic Systems Development Method is to fix the time and resources at first and then adjust the amount of functionality to be developed accordingly. In Dynamic Systems Development Method the iterations are time boxed. The time box is a span of time. The iterations must not exceed this time box [35].

It consists of five phases. The first two phases are carried out sequentially. In the last three phases the actual development takes place. The last three phases are iterative and incremental.

Feasibility study: In the feasibility study the suitability of the project is checked with the Dynamic Systems Development Method. The criteria for suitability are type of the project, people and organizational issues. The feasibility study is associated with two aspects of the project, they are, technical possibility and associated risks. The risks are calculated. At the end of feasibility study a feasibility report and outline plan for development is produced. It there is no sufficient information about the technical and business aspects of the project, a prototype can be created. This prototype then helps to decide the suitability. The duration of feasibility study is few weeks.

Business study: This phase begins with the analysis of business and technology characteristics. The analysis is carried out by conducting workshops. Good number of customer's experts attends this workshop. In this work all the characteristics of the system are discussed and development priorities are set. The business area definition describes the affected business process and user class. This affected user class are key the people in customer's organization. These people are involved with the project from the early stage. Business area definition also describes processes in the form of ER diagrams or object models. In this the architecture definition of the system is produced. Also the outline prototyping plan is produced. The architecture definition may change during the development period. The prototyping plan consists of prototyping strategies for the subsequent stages and a configuration management plan.

Functional model iteration phase: This phase is the first iterative and incremental phase. At the beginning of iteration, the items to be implemented and the approach to be followed is planned.
At the end of iteration, the complete iteration is analyzed and results are produced for future iterations. In every iteration analysis and coding is carried out, prototypes are built. The experience from the previous iterations is used to improve the analysis models. The prototypes built in the current iteration are not disposed but they are improved. Such improved prototypes exist in the final system. A functional model containing the analysis models and prototype code is produced. Testing of code is carried throughout the phase. Other outputs in this phase include prioritized functions, functional prototyping review documents, non- functional requirements and risk analysis of further development document. These outputs are produced at different stages in the same phase. Prioritized functions are set functions to be implemented in the iteration. The functional prototyping review documents contain the comments from the customer about the current increment. This document helps in improving the subsequent iterations. Non-functional requirement serve as an input to next iteration. Risk analysis of further development document helps in understanding of the future problems.

Design and build iteration: In this phase the system is actually built. At the end of this phase, a completely tested is produced. All the agreed set of requirements is present in this system. Design and build phase is iterative in nature. The users review the design and functional prototypes. The user comments serve the basis for further development.

Final implementation phase: It is in this phase the system is actually moved from development environment to production environment. The users are trained to use the system and then system is given to them. If the system built is used by many users, the final implementation phase becomes iterative. Other outputs of final implementation include user manual and project review report. The project review report contains opinions about the system. This information helps in further development. At the end of final implementation phase there may four kinds of scenarios. Scenario one, the system built satisfies all the requirements, then no further development work is carried out for the system. Scenario two, if the system built does not include many requirements, the whole process is run again from start to end. Scenario three, if the system built does not satisfy some less critical functions, the process is run again but from functional model iteration phase to final implementation phase. Scenarios four, if because of time constraints some technical issues are not addressed, the process is run from design and build iteration to final implementation phase.

The roles and responsibilities in Dynamic Systems Development Method are

Developers and senior developers: These people are involved in the development of the system. Seniority is based on the experience. The developer and senior developer roles include analysts, programmers, designers and testers. The senior developer leads the team.

Technical coordinator: The technical coordinator is responsible for the technical quality in the project. He defines the architecture of the system. He is also responsible for software configuration management.

Ambassador user: The ambassador user brings the knowledge from user community to the project and gives information about the progress of the project to the user community. This ensures sufficient feedback from user community. The ambassador user belongs to user community, who will use the system.

Advisory users: These are the users who represent the important viewpoint with respect to project. The advisory users are either financial auditors or from IT staff

Visionary user: This is the user who has accurate perception of the business objectives of the system. This user is usually the person who had the initial idea to develop the system. The responsibility of the Visionary user is to make sure the requirements are identified early. Another responsibility is to ensure that project is progressing in the right direction.

Executive sponsor: The executive sponsor has the ultimate power to make decision. He belongs to the user organization and has financial authority and responsibility.

Project manager: He manages the project. He belongs to user organization or IT staff.

Tester: The tester tests the system. He provides comments and documentation.

Scribe: He gathers and records information like requirements agreements and decision made in everyday workshop.

Facilitator: He is responsible for managing workshops progress.

For a successful Dynamic System Development implementation, nine principles are followed strictly.

Active user involvement is imperative: This is the first and the most important principle. Continuous user involvement with the system being developed helps developers in understanding the user perception. This reduces the error costs. Dynamic System development method provides guidelines to select user and advises not to include large number of users. Small set of users are involved till end of project.

Teams must be empowered to make decisions: To speed up the process the friction in communications of project managers and project members should be avoided. One of the reasons to hamper the development speed is waiting for an approval for simple requirement changes. To solve this problem the users and development team members must be given limited authority to make decisions about changing requirements, the functionality to be included in the current increment, small details of the technical solution, prioritization of features and requirements.

Focus on frequent delivery: With frequent delivery of implemented features error can be detected easily. The errors can be tracked down and corrected. This ensures the correct system being built. With frequent delivery not the errors in the code but also the errors in requirements document and data models can be identified.

Fitness for Business in criterion for accepted deliveries: The aim of Dynamic System Development method is to deliver a system that satisfies the business needs. Any other requirements are included in the next release. Refactoring, design engineering and feature enhancement are part of the any software development cycle. These attributes must be take care during the project rather than enforcing them at the end. Dynamic Systems Development method does not provide complete system as whole. Instead it provides a system that satisfies the business needs. The attributes mentioned above are taken care in iterations. Even in Dynamic Systems Development method key issues needs to be identified, this requires a robust design. Agile software development methodologies require more understanding of patterns and architecture compared to traditional software development methods. This is because it the architecture is not good then it may not possible to carry out many iterations. The development of the system may stop.

Iterative and incremental development is mandatory: In order to tackle the complexity problems, the system to be built is broken down to a set of features. In the iteration a set of features are implemented. Finally after a number of iterations the complete system is built. This system satisfies the business needs. This principle addresses the issue of changing requirements by small increments.

All changes during development must be reversible: A change request may arrive for reason like the priority of a requirement may change. To adopt this change to the system during iteration demands changes in the system configuration. Many tools support this dynamic configuration changes. If a development process is reversed the previous work is lost. In Dynamic Systems Development method the system is developed in small increments. Hence if the development process is reversed, it will result in the loss of small amount of work.

Requirements are base lined at high –level: The requirements keep changing during the development process in Dynamic Systems Development method. To avoid the requirements from changing frequently, some high level requirements are established. These high level requirements form the baseline. This base line is also referred as requirements freeze. These base line requirements are established in the business study phase.

Testing is integrated throughout the life cycle: In many software development methods the testing is performed at the end. But in Dynamic Systems Development method it required to start testing early. The testing is also carried out for interview documents.

Collaborative and cooperative approach: In Dynamic Systems Development method it is necessary to avoid separation between the business staff and IT staff for the successful development of the system.

The Dynamic Systems Development method was described above in detail. Dynamic Systems Development method has few phases and many roles. It uses best practices to avoid failure of projects. But the approach is not suitable for STPD.

- If the Dynamic Systems Development method is used for STPD, then cloud team will spend time in feasibility study and business study than actually developing the model.

- In Dynamic Systems Development method the time and resources are fixed and then amount of functionality chosen in respect to software development. This is not suitable for STPD model, because the resources (in terms of software developers) are not fixed. This obstacle between the STPD and Dynamic Systems Development method cannot be removed. Hence the recommendation is not to use Dynamic Systems Development method for STPD.

4. Agile Modeling

The underlying idea in Agile Modeling is encourage developer to produce enough models to support acute design problems and documentation purposes, while the number of models produced is still low. The main focus on culture issues is reflected in the support communication, team structures and the ways the teams work together. The values behind Agile Modeling are programming, simplicity, feedback and courage. The core values behind Agile Modeling promote the importance of functioning software as a main goal. Agile modeling strongly focuses on practices. There are several practices defined for each of this categories iterative and incremental, modeling, team work, simplicity, validation. To model in an Agile manner the best practices needs to be applied. The best practices in Agile modeling are active stake holder prioritization, prioritized requirements, model a bit ahead, architecture envisioning, document continuously, model storming, requirements envisioning, just barely good enough, document late, multiple models, single source information, iteration modeling , test driven design, executable specifications. The practices range from selection of proper modeling technique to model the item selected, to the benefits of team work. In Agile Modeling approach, a high level modeling is applied at the beginning of the project to understand the scope and architecture of the system. During the development iteration, the model is a part of iteration planning. Agile models are good, if they are simple, understandable, sufficiently accurate, consistent and detailed, fulfill their purpose. Agile Modeling favors small teams, customers presence and close communication. The Agile Modeling does not impose any restriction on size of the system built. Agile Modeling is not sufficient by itself. It needs supporting methods. Agile Modeling is not suitable for large teams and teams which are not collocated [2].

Agile Modeling is methodology for effective modeling and documentation of software based systems. Agile Modeling is not complete software development methodology. It requires supporting methods. Therefore Agile Modeling is not recommended for STPD.

5. Pragmatic Programming.

Pragmatic Programming does not a process, phases, distinct roles and responsibilities. The philosophy behind pragmatic programming is

- Take responsibility for what you do, think solutions instead of excuses.
- Don’t put up bad design or coding.fix inconsistencies as you see them or plan them to be fixed very soon.
- Take an active role in introducing change wherever you feel it is necessary.
- Make software that satisfies your customer but know when to stop.
- Constantly broaden your knowledge
- Improve your communication skills.

Pragmatic Programming demonstrates simple, responsible and disinclined software practices. The practices are mostly from the programmer's viewpoint ranging from how to avoid typical coding and design errors, to communication and teamwork issues [2].

Pragmatic Programming does not have a process, phases, distinct roles or products. The Pragmatic Programming relies on six philosophies, one which states "take responsibility for what you do". This is hard to follow in STPD model, as people move in and out of the model. Consider a scenario, a cloud team member say 'A' is working on task 1 and leaves the STPD without completing task 1. In the above situation, the first philosophy does not hold. Therefore Pragmatic Programming is not recommended for STPD.

6. Scrum

The main aim of Scrum is to facilitate the development of software in an environment where requirements, time frame, resources and technology are subjected to change. Below is detailed description of Scrum [36].

The Scrum begins when the product owner has an idea (product to be developed). The product owner comes up with a plan for the product. The product owner can be a client or a customer proxy. For product oriented companies, the client is market. Therefore for product oriented companies market is the proxy product owner or can also be called customer proxy. Before approaching the Scrum team, the product owner needs to have, a vision, the purpose of the product, a business plan which shows the anticipated profits, a road map which shows the releases and list of features to be implemented. The product owner then prepares the list of requirements and prioritized them. Priority can be assigned based on the business value. This list is called product backlog. These are the requirements to be present in the final product. The Scrum team helps the product owner to prioritize the list of features, by provide the cost and effort estimate to develop the features. The product backlog includes all the requirements of mentioned by product owner and also includes the technical requirements needed to develop the features. Once the product backlog is defined, the backlog features need to broken into tasks. The highest priority features from product backlog are chosen first. All the product backlog features are such that, each feature can be developed with approximately 10 developer working days. Features that will be implemented in the next sprint are dealt.

Once the product backlog is defined and features in product backlog are prioritized, the actual development of the backlog features begins. In Scrum the product is not built as a whole. Instead the product is developed in iterations. After every iteration the product is available with a set of features. In Scrum an iteration is called sprint and every sprint is 30 days long. A sprint planning meeting is held before every sprint, in which a detailed plan for the sprint is prepared. In the sprint planning meeting the product owner and Scrum review the vision, the roadmap, product backlog and release plan. The team reviews the product backlog and checks if the description provided is sufficient. Scrum team decides on the set of features it can complete in the sprint. This decision is based on team productivity, Scrum team size and available hours.

Once the Scrum team decides on features to implement in the current sprint, the team breaks the features into sprint tasks under the supervision of Scrum master. This part of sprint planning

meeting should not exceed four hours. Each feature is broken into set tasks such that it will require not more than 2 developer working days or 16 hours. When all the features are broken into tasks, the total effort required is calculated. This calculated effort is compared with the original estimate. If there is a large difference between the calculated effort and original estimate then the team negotiates with the product owner and takes appropriate amount of work.

Throughout the Scrum, daily Scrum meetings are held. The Scrum master leads the daily Scrum meeting. This is fifteen minutes meeting between the Scrum team members and Scrum master. Other people can attend this meeting, but only the Scrum team members are allowed to speak. In this daily Scrum meeting the Scrum team members answer three questions namely what did I do yesterday? What will I do today? And what are the impediments in my way? The aim of the meeting is to discover dependencies, to address any personal issues, adjust the plan and to know the project status.

The responsibilities of the Scrum master in the daily Scrum meeting are

1. The Scrum master should know, the completed tasks, newly added tasks, tasks under progress and any estimate to be changed. All this information helps the Scrum master to update the burn down chart. The burn down chart the remaining work. The Scrum master must keep track of the tasks in progress.
2. The Scrum master needs to address the impediments. Some impediments can be resolved within a team, some impediments need management attention and some impediments can be resolved across teams. These impediments are prioritized and tracked by Scrum master. The Scrum master prepares a remediation plan for the impediments in the order of priority.
3. The Scrum master also needs to resolve the conflict within team members by means of communication.

At the sprint planning meeting the team decides on the set of features to be implemented in the sprint. The cumulative backlog contains the estimated effort for the sprint. As the sprint proceeds, the tasks are completed. Scrum master recalculates the amount of work remaining. Thus the sprint backlog decreases or burns down. At the end of sprint an empty sprint backlog implies successful sprint. The sprint backlog is fixed but can change for the following reasons.

- As the sprint proceeds, the developers get better understanding of the product. This may result in adding of new tasks to product backlog or sprint backlog.
- During the sprint, the defects identified are added as new tasks. Such tasks are reviewed and prioritized.
- The product owner may work with the team. He helps the team in better understanding of the product.

The burn down chart is used to guide the team for successful completion of the sprint.

A sprint review meeting is conducted at the end of every sprint. In the first half of the sprint review meeting, the Scrum team demonstrates the product built. The product owner leads the first half of the meeting. This meeting is attended by other stakeholders. The state of market, business and technology are reviewed. The product owners reviews the features developed and determines the features to be built in the next sprint. The product owner, the Scrum team and other stakeholder

together prioritize the features to be implemented in the next sprint. The first of half of sprint review meeting lasts for four hours. Scrum master leads the second half of the sprint review meeting. Only Scrum master and Scrum team members can attend. The Scrum master asses the way the team worked together.

Scrum roles

In Scrum there are primarily three roles

Product owner: The product owner defines the product and features. The product owner and the customer can be same. If the product owner is market need, then the project manager is the product owner.

Team members: Team members build the product. The minimum team size is three and the maximum team size is 15. The ideal tem size is 5-7. The team must possess all the expertise to complete the work. If the project has more than 15 members, then multiple teams are formed. Each team focuses on certain set of features. More coordination effort is required with multiple Scrum teams. The teams can split. It is possible that a member of a Scrum team can move to another Scrum team. This reduces the productivity.

Scrum master: The Scrum master facilitates the meetings. The Scrum master is not the project manager. The Scrum master must remove the impediments. Some Scrum teams have a dedicated Scrum master. In some teams, a team member plays the role of Scrum master.

6.2 How much of Scrum fits/misfits in Model

Similarities between Scrum and STPD

- The number of members in the scrum team varies from 5-7. This matches with our requirement of the model that the number of people on bench is not fixed thus if there are few members on bench they can form a scrum team.
- The sprint duration is helpful as the people on bench are not available for the long duration
- Scrum involves implementing the prioritized features first, rather than the whole project.
- In the initial phase of the scrum the tasks are broken down to tasks requiring work of maximum two days.
- The product owner comes with list of features and technical requirements. This saves time and problems arising from feasibility study and requirements collection.
- The team member choose the tasks, it is not assigned.

Differences between Scrum and STPD

- The model requires more documentation than compared to scrum.
- The team is not fixed.
- In normal scrum the team decides on how much work it can take based on the current team size. In the model, this would be challenge as the team size is not fixed.

The differences between the Scrum and the STPD are not the major differences, with some alterations the Scrum can be made to work with STPD. Having the Scrum process described. One more requirement is made on STPD model

- The STPD model consisting of only cloud team members (no core team member) and future client.

6.3 Recommendation on process

The following are the recommendations to be followed along with the Scrum.

- After the features required to implement in the current sprint are determined, it is required to build a picture of how the end product with the current set of requirements would look like. In this way information required to build the right product would be carried till the end of the sprint.
- Therefore in the initial phase the team members must try to build this picture by dividing the set of features among them.
- The functionalities of the requirements can be best expressed as a usecase diagram.
- Another challenge would be to decide on the team productivity, but as the assumption is made that the team would consists of only the software developers working on the project, the initial team can pull out the items to be delivered in a sprint of 30 days.
- The daily scrum meeting has to be carried out without the scrum master, where the team members discuss the answers to 3 questions. Finally the answer to the question “what are the impediments in my way” has to be resolved among the team members by listing the impediments and prioritizing them. One more way to resolve this problem is to use post it view, where all the impediments can be quickly glanced. The answers to all the three questions can be presented in the form of post it view. Some tools support it. The answer to the question “what did I do yesterday?” can be set as a status message the social networking media like twitter, where the team forms a group.
- A shared document can be used to keep track of the daily meetings.
- Every team member updates the information about the tasks started/completed/ remaining to burn down chart.
- Every team member must take the responsibility to save the necessary information about the task carried out by them with the help of tooling support. This helps a new member coming into the project to get acquainted with the tasks quickly.
- The absence of ScrumMaster, this problem can be resolved in the following way. If a team member is available for the entire duration of one sprint or for most of the sprint, he can take up the responsibility of the ScrumMaster.
- If the ScrumMaster had to leave the STPD before the sprint ends, a new ScrumMaster can be chosen from the team members. The criteria for choosing ScrumMaster would be oldest (in terms of the amount of time spent in STPD) Scrum team member.
- To be a team member of the STPD, he must be available for a minimum period of two days.
- The number of members in the STPD is not fixed.
    - If number of members is one or two, then the members can work on the project in two ways. If it is the initial phase of the project then they can build picture of what the end product will look like. If the project is in development phase then they can take up the tasks and work on them. In this case there is no ScrumMaster required.
    - If the number of members is three or more, then they form a scrum team. The Scrum team size varies from 3 to 5-7. If there is more number of members in STPD then they form multiple Scrum teams.

- The team members can use mailing lists. This helps in solving impediments and also allows people away from the STPD model to help resolve the development problems.
- Good personal information (like email) of the cloud team members should be maintained at least till the end of sprint. This can be achieved with tools or by maintaining a shared document.

Mapping of roles.
The product owner in the Scrum can be mapped to the future client in STPD or the Solution manager delegated as the product owner. The ScrumMaster in Scrum can be mapped to a team member (if any) who is available for the entire sprint or for most of the sprint. The team members of Scrum are mapped to team members of STPD.

All the assumptions mentioned in the section 2.4 still hold valid in the adapted Scrum process. The adapted Scrum process satisfies the high – level requirements mentioned in section 2.4

6.4 Tools applicable

This section describes a set of tools which support Agile software development. All the tools surveyed are Open Source project management tools. STPD requires both process and tooling support. All the tools are described in three parts. The first part briefly describes the tool. The second part lists all the features supporting the task creation and management. The third part lists all the features supporting knowledge transfer. The tools are studied from respective tool's homepage. The description presented below is based on executing the tool.

1. Agilefant
   - **Description**
     Agilefant is a web based tool for agile software development activities such as product backlogs, iterations and releases. It also supports versioning [3].

   - **Task creation and management**
     - It allows for the creation of tasks.
     - It allows for moving the features between the sprints.
     - It supports auto assign of tasks, which means the user is free to choose the tasks.
     - It has support for add/delete people to tasks and features at any time in the sprint.
     - It supports the iteration backlog. In this iteration backlog the stories can be divided into tasks.
     - It supports a product backlog and for every product a new project, iteration, user, team and story can be created.
   - **Knowledge Transfer**
     - Team member information is available to all other members. Team alteration in terms of adding a new member or deleting an existing member is supported.
     - It provides a feature called Time sheet. Time sheet has a built in time tracking system, which allows the user to log effort to backlogs, tasks and stories. This information can be stored in the form of webpage or excel sheet.

- It supports project portfolio management by collecting all the upcoming release projects and under progress projects in one place.
- It provides a feature daily work. This feature places all the tasks of a user from all the iterations in one place. It provides an option to arrange all the tasks of single user in the form of queue. The daily work feature is public. It means a user can see the daily work other user.

2. Agilo for Scrum

- **Description**

  Agilo for Scrum is a web based tool to support the Scrum process. It supports multi team and multiple projects. It supports teams, Scrum master, product owner, all stake holder involved in the project. It supports versioning [4].

- **Task creation and management**
  - It provides hierarchical tickets to link tasks to stories.
  - It provides a facility to edit tasks in the backlog view and to modify its attributes. In the product backlog view tasks are shown with different colors.
  - It provides option to link stories to multiple requirements, this helps in improving traceability.
  - It supports traceability (bidirectional links) between the requirements, user stories, and tasks.
  - It supports product backlog with prioritization. It also support mapping of stories to sprints from product backlog.
  - It supports bug backlog and impediment backlog.
  - It supports complete configurability, adding custom types and field type to existing types, to define custom work flow for tickets etc.
  - It supports scrum dashboard and roadmaps. It supports reports for every role.
  - Scrum dashboard provides a quick view of sprint. Here the user can add items, create modify or delete tickets and navigate to product backlog.
- **Knowledge transfer**
  - It supports all the three roles of Agile development process, the scrum master, the team members, and the product owner. It provides the necessary feature support for all the three roles.
  - It can be extend with (free) plugins.
  - It supports integrated team management.
  - It supports integrated wikis.
  - It supports subversion integration, bugs, tasks can be closed with commit message.
  - It supports tight integration with the source code. It facilitates to browse code, link to change sets, and displays differently with different version control (Subversion, Git, Perforce and Bazaar).
  - It supports export of CSV files to excel and import with edit and delete options
  - It supports burn down charts for distributed teams.
  - Updating of task status is done with a commit message.

3. Agile Tracking Tool
   - **Description**

     Agile Tracking Tool can be used for Agile software development process. It has little or no information about the team members. It has a support for maintaining multiple projects. It has support for iterations [6].
   - **Task creation and management**
     - It supports prioritization of backlog tasks.
     - It supports categorization of backlog tasks.
     - It provides support to add acceptance criteria and comments to tasks in backlog.
     - It supports breaking of tasks into subtasks.
     - It supports the blocking of tasks.
     - It supports current iteration.
     - It supports future iteration.
   - **Knowledge transfer**
     - It shows how much work is in progress and completed.
     - Status of each task in the current iteration can be seen.
     - It has support for export and import of data.
     - It also supports for forecast of when the work will be completed [6].
     - It supports burn down charts.
     - It supports bug history in the form of graph.
     - It supports flow history in the form of graph.
     - It supports iteration history in the form of graph and text representation.

4. Digaboard
   - **Description**

     It is a visual project management tool. It uses the whiteboards with stickies for visualizing the tasks. It has multi team support and multi project support. It consists of team board, which consists of team members and the project they are working on [7].
   - **Task creation and management**
     - It supports visual tracking of tasks.
     - Idle tasks, recently moved tasks, and blocked tasks are shown in different colors.
     - For every task, it displays status of task and assigned team members for that task.
     - It supports adding comments and task information.
     - It supports prioritization of tasks.
   - **Knowledge transfer**
     - It supports addition of links (containing necessary information) to the tasks.
     - It has no support for reporting.
     - It does not support burn down chart.
     - It supports team boards. Columns in the team board represent teams. Row split represent the active and inactive projects.
     - It supports queuing of items for future implementations.

5. Express - Agile Project Management
   - **Description**

     Express is an open source Agile project management tool. This tool focuses on iteration and backlog management. It has support for backlog management, virtual wall, project creation and managing the user details [16].
   - **Task creation and management**
     - It supports product backlog list.
     - It supports iteration backlog list.
     - It supports addition of tasks to stories.
     - It also supports prioritization of tasks and grouping of tasks.
     - It supports status of tasks (open, in progress, test and done).
     - It supports drag and drop functionality in the backlog management both iteration backlog and product backlog.
     - With every story acceptance criteria can be added.
     - It supports adding of impediments.
   - **Knowledge transfer**
     - It supports project summary.
     - The virtual wall displays the stories in the form of virtual index cards and tasks are of individual team members is displayed in the form of swim lanes.
     - The virtual wall support makes this tool useful for distributed teams.
     - It also supports burn down chart and velocity comparison chart.
     - It supports project summary. In project summary, it displays the information about the project, supports management of developers and themes and allows export of product backlog to csv.
     - It also supports iteration summary. It display information about the iteration, it displays the iteration burn down chart and allows export of stories to csv.
     - It supports tracking of impediments.

6. FireScrum
   - **Description**

     FireScrum is a Scrum projects management environment. It supports product management, sprint management and release planning [8].
   - **Task creation and management**
     - It supports task management and task board.
     - For every task it shows, task description, owner of the task, effort spent, effort remaining to complete the task, status of task.
     - With every task it provides graphical icon support, to allocate, to de-allocate and to impediment.
     - It supports product backlog and committed backlog.
     - For every product it supports creation of test suites and test plans.
   - **Knowledge transfer**
     - It supports task board view of backlog items, impediments, completed items, incomplete items and to-do items of the user.
     - It supports reports in the form of burn down charts.
     - It supports bug tracking.

- It supports planning poker.
- It supports Impediments management.
- It supports agent desktop.
- It supports sprint review.
- It supports sprint retrospective.
- It supports user management.

7. IceScrum
   - **Description**

     IceScrum is Scrum project management tool. It supports releases. It supports multiple projects, acceptance testing. It supports multiple products but allows only single sprint and single release for a particular product. It is not suitable for large products [9].
   - **Task creation and management**
     - It supports task hours.
     - It supports task board view.
     - It supports story points.
     - It supports hierarchy of backlog items.
     - It supports ranking of backlog items.
     - It also supports locking of tasks.
   - **Knowledge transfer**
     - It supports iteration burn down chart.
     - It supports roadmaps.
     - It supports impediment tracking.
     - It supports planning poker.
     - No support for reporting.

8. Planigle
   - **Description**

     Planigle is a web application for managing the backlog iterations and releases. It supports team management [10].
   - **Task creation and management**
     - It supports task creation.
     - It supports prioritization of tasks.
     - It supports blocking of tasks.
     - It supports backlog creation.
   - **Knowledge transfer**
     - It has little support for reporting.
     - It supports the display of status of tasks.
     - It supports multiple projects.
     - It supports iteration planning and release planning.
     - It supports multiple roles with different permission for each role

9. Retrospectiva
    - **Description**

      It is a web based project management tool. It supports full development life cycle [11]. It has poor user information management [11].
    - **Task creation and management**
        - It supports creation of tasks.
        - It supports prioritization of tasks.
    - **Knowledge transfer**
        - It supports to browse code.
        - It supports blogs.
        - It supports changesets.
        - It supports milestones.
        - It supports RSS.
        - It supports tickets.
        - It supports wiki.
        - It supports notification of tasks.
        - It supports status of tasks.
        - It has support for tracking issues.

10. Scrum Dashboard
    - **Description**

      The Scrum Dashboard is a web front end for Scrum which aims to simplify the daily work with the Scrum artifacts. All features are sprint centric for daily work in sprint [12].
    - **Task creation and management**
        - It supports creating, adding and updating of product backlog.
        - It supports creating and updating of bugs in the sprint backlog.
        - It supports creating and updating of tasks in the sprint backlog
        - It supports creating and updating of impediments in the sprint backlog
        - It supports drag and drop of items between states.
    - **Knowledge transfer**
        - It supports inline editing.
        - It supports burn down chart.
        - It automatically displays new projects, sprints and teams.

11. Scrum Time
    - **Description**

      Scrum Time is a web-based Scrum project management tool. It supports to add projects [13].
    - **Task creation and management**
        - It allows addition of tasks to stories
        - It supports prioritization of tasks.
        - It has support to close and open the tasks.
        - It supports reporting of bugs.
    - **Knowledge transfer**
        - It supports burn down charts.

12. Tackle

- **Description**

  Tackle is a web based tracking site, it helps manage small and large teams. It supports multiple Scrum team management, multiple product backlogs per Scrum team [14].
- **Task creation and management**
  - It supports product backlog entry per product.
  - It supports intertwining of all product backlogs into a single product backlog for prioritization.
  - It supports for splitting of tasks.
- **Knowledge transfer**
  - It supports auto generation of sprint backlogs based on product backlog prioritization.
  - It supports multiple reports including sprint totals, sprint tasks per product or owner, burn down graphs, status reports, and work per day reports, cross-scrum team reports, and sprint review documents.
  - It supports Sprint retrospective note tracking.
  - It supports Personal status report tracking.

13. The Scrum Factory

- **Description**

  The Scrum Factory is a client server application that allows creating and managing project backlogs, assigning daily tasks, plan sprints [15].
- **Task creation and management**
  - It supports to add backlog items, measure its size, to estimate the effort to implement them.
- **Knowledge transfer**
  - It provides instant view of each project development team and what a particular team member is working on.
  - It supports daily meetings with the post it view.
  - It supports burn down chart to keep track of progress.

6.5 Tool recommended

Among the tools surveyed, some tools provided good support for task management but lack support for knowledge transfer. Some tools did not support release. The tool Agilo for Scrum is an open source tool (available for free) and has support for both in terms of task management and knowledge transfer. The tool can be adapted to new needs. This is the recommended tool for the STPD.

Reasons for not selecting other tools is
Agilefant: This tool is not suitable for large projects.
Agile Tracking Tool: This tool has good support for task creation and knowledge transfer but does not support wiki. Reports are in the form of burn down charts.
Digaboard: This tool does not support wiki, bug reporting.

Express – Agile Project Management: This has good support for task creation and knowledge transfer. It does not support traceability and reports are in the form burn down chart.
FireScrum: This tool also has good support for task creation and knowledge transfer. It does not support traceability and reports are in the form burn down chart.
IceScrum: This tool does not support wiki and traceability. Reports are only in the form of burn down chart.
Planigle: This tool does not support wiki, bug reporting and traceability.
Reterospectiva: This tool has good support for task creation and knowledge transfer. It does not support traceability. It has poor user information management.
Scrum Dashboard: This tool does not support wiki and traceability.
Scrum Time: This tool does not support wiki, bug reporting and traceability.
Tackle: This tool has good support for task creation and knowledge transfer. It does not support wiki and traceability.
Scrum Factory: This tool does not support wiki traceability and bug reporting.

7 Conclusion

This chapter provides conclusion for our study on RUP and agile process and STPD.

Conclusion for developing RUP based projects using STPD

The recommendations on developing RUP based projects using STPD are given in Chapter 5 section 5.3.5 and recommendations on tooling used for developing RUP based using STPD are given in section 5.4.3. In this section some of the recommendations are mentioned though they are mentioned in sections 5.3.5 and 5.4.3 of Chapter 5.

The RUP process does not completely fit into STPD. Only the Construction phase of RUP fits.

The main requirement is to define the tasks in such a way that they do not require the knowledge of the system under development to implement them. This is required as Cloud team members move in and out of a team. In their spare time it is difficult get the knowledge of the system and implementing its tasks. So this is one of the main requirements for a RUP based project to be developed using STPD. The core team size must be big for many roles of RUP to be mapped to the roles of STPD.

In tooling used for RUP based projects, many project management tools were studied for the Construction phase of RUP. Most of the project management tools do not provide the feature for showing dependencies between tasks of a project. This is an important feature required in tool used to develop RUP based projects, as RUP based projects have many and interdependent tasks. The tasks which are not dependent on any task are chosen and implemented then the dependent tasks can be implemented. The Zoho project management tool provides this feature but expensive.

Agile Dashboard has a tool which can be used if extended with any of the knowledge transfer techniques mentioned in Chapter 5, section 5.4 subsections 5.4.1. Task management tools provide all the features for task creation but do not provide feature to communicate. Task management tools like Gantt project are available freely which can also be extended with any of the knowledge transfer techniques.

Conclusion (whether to employ STPD or not)

The study of Agile software development methods is presented in chapter 6. The chapter began with the study Agile software development methodologies. Agile software development methodologies like Extreme Programming, Scrum, Feature Driven Development, Dynamic Systems Development Method, Agile Modeling, and Pragmatic Programming were described in detail. These methodologies are not suitable for STPD, valid reasons are presented after the description of each methodology.

The main objective of the STPD was to employ people on bench to develop products. The solution manager and the core team members form the permanents members of STPD. After the study of Scrum, It was possible to propose a set of recommendations (in section 6.3) which supports the execution of the STPD with only the cloud team. This means that there is no core team. STPD purely consists of future client and cloud team members.

Even though the STPD is facilitated with the normal Scrum and set of recommendations (in section 6.3), there are certain limitations.

- The adapted Scrum does not address the situation, when is the project is half way (or in progress) and there are no people on bench.
- Using this adapted Scrum a product may be completed but does not always guarantee the business value of the product. This is related to first limitation mentioned.

The conclusion of this study

1. The recommended process to implement STPD is Scrum with the set of guidelines.
2. The recommended tool to implement STPD is Agilo for Scrum.
3. The STPD should be implemented. It will serve the purpose of educating the people in Scrum development process at minimum.

The STPD will be of use, if there are on an average of at least 3 members on bench for most of the time. This is because the minimum size of a Scrum team is 3. When the STPD is implemented, the first project could be to improve the tool, instead of real product development.

8 References

[1] Capgemini company profile, http://www.amdocs.com/About/Partners/Profiles/Pages/Capgemini.aspx

[2] Abrahamsson, P., Salo, O., Ronkainen, J.: Agile software development method review and analysis, VTT publications, Finland(2002).

[3] Agilegant,http://www.agilefant.org/

[4] Agilo for Scrum, http://www.agile42.com/cms/pages/agilo/

[5] Agilo for Scrum, http://www.opensourcescrum.com/

[6] Agile Tracking Tool, http://sites.google.com/site/agiletrackingtool/home

[7] Digaboard, http://www.digaboard.net/

[8] FireScrum, http://www.slideshare.net/ericoc/firescrum

[9] IceScrum, http://olex.openlogic.com/wazi/2009/comparing-open-source-agile-project-management-tools/

[10] Planigle, http://planigle.com/

[11] Retrospectiva, http://retrospectiva.org/

[12] Scrum Dashboard, http://scrumdashboard.codeplex.com/

[13] Scrum Time, http://sourceforge.net/project/

[14] Tackle ,http://tackle.codeplex.com/

[15] The Scrum Factory, http://www.scrum-factory.com/

[16] Express –Agile Project Management, http://agileexpress.sourceforge.net/

[17] Gantt Project, http://www.ganttproject.biz/

[18]HotProject, http://www.hotproject.com/

[19]Teamwork Project management application, http://www.teamworkpm.net/

[20]@task, http://www.attask.com/

[21]activCollab, http://www.activecollab.com/

[22]CentralDesktop, http://www.centraldesktop.com/

[23]WorkBench, http://www.openworkbench.org/

[24]Planzone, http://www.planzone.com/

[25]COMINDWORK, http://www.comindwork.com/

[26]Zoho Projects, http://projects.zoho.com/

[27]Capgemini RUP process, http://deliver2.capgemini.com/components/rup_ddf/index.htm

[28]HyperOffice, http://www.hyperoffice.com/

[29]ProjectSpaces, http://www.projectspaces.com/

[30]EasyProjects .Net, http://www.easyprojects.net/

[31]GoPlan, http://goplanapp.com/

[32]Gantt Project, http://www.ganttproject.biz/

[33] Tejaswi J.M., IT companies reducing bench strength, *The Times of India*, June 11 2010.

[34] Jami, S.I., Shaikh, Z.A.: Teaching Computer Science Courses using Extreme Programming(XP) methodology. International multitopic conference, Karachi(2007).

[35] Voigt, B.J.J.: Dynamic System Development Method, University of Zurich, Zurich(2004).

[36] Sutherland, J., Schwaber, K.: The Scrum Papers: Nuts, Bolts, and Origins of Agile Process, Washington(2007).