International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 8, No.5 (Sep-2019)

http://dx.doi.org/10.12785/ijcds/080502

Analysis of Software Development Methodologies

SoobiaSaeed[1], NZ Jhanjhi [2], Mehmood Naqvi [3] and Mamoona Humayun [4]

[1]*Department of Software Engineering, Universiti Teknologi Malaysia*
[2] *School of Computing & IT (SoCIT), Taylor's University, Subang Jaya, Selangor, Malaysia*
[3] *Department of Computer Science, Sheridan College, Canada*
[4]*College of computer and information Sciences, Jouf University, Saudi Arabia*

Received 2 Feb. 2019, Revised 3 May 2019, Accepted 2 Jun. 2019, Published 1 Sep. 2019

Abstract: The researcher focuses on the analysis of most common diverse methodologies of software development to choose the best one on the basis of different factors such as project type, size, development environment, and available resources. Software projects provided are positive and negative impacts and provide the stages of software development methodology. Subsequently, the author gives brief details about the common stages of software development in this paper. These stages are mostly used in every software development methodologies (SDMs). The main motive of this research is to provide the details of figures of steps and stages about currently available most common twenty-one (21) SDMs. Software projects are on the functions or stages of the methodology, the project owner's feedbacks in each methodology and suitability of methodology on the small, medium and large size of projects. The Result conducted based on an analysis between them by applying different strategies, development environments, and common practices and based on available resources, which can easily be understood to choose the best methodology, which can be feasible for Small, and Medium Enterprises (SMEs).

Keywords: Software Development, Project Management, methodology, Software Development Life Cycle

1. INTRODUCTION

Each project contains a system that follows these systems to achieve project completion. Such as a software project, that also has many systems or development methods according to the nature or size of the project. However, a difficult decision that takes time for the project manager is the right method for the project. The project manager selects the development method, taking into account the time and budget of the project in particular. for success Many software development methodologies have been introduced and knowledge of all software development methodologies is difficult for the project manager [1]. It began in the 1960s to make this method meaningful in time and budget. It was the era of major computers and took place in the flowchart in ways or stages. Technology changes and how they succeed change day after day. The technological age is rapidly changing year after year. There are many methods for developing computer programs and the appropriate method for the desired project is difficult to determine. This study provides details of the most common stages and stages of program development methodologies.

The comparison table shows the most common software development methodologies for their strengths and weaknesses. The researcher provides a comparison between software development methodologies, but only some of them in previous stories [2]. The comparison table shows the most common software development methodologies for their strengths and weaknesses. The researcher provides a comparison between software development methodologies, but only some of them in previous stories. The Developer Project Manager deals with the most important challenges and solutions for software development. There are benefits of a project management process software development that can help manage the burden of management [2].

The software development project's planning and notorious specifications are often changed. Some social groups discussed key reasons to change the specifications of the project. In LinkedIn, 11 groups began a discussion on frequent changes in the initial planning and specification of renowned software projects. What are the main reasons for changing requirements in the development phase? There are only three groups dealing with project managers. Information gathered from these groups on the facts of the specification changes:

E-mail:soobiasaeed1@gmail.com, noorzaman.jhanjhi@taylors.edu.my, dr.naqvi@gmail.com, mahumayun@ju.edu.sa

- ✓ Owner The project owner or customer identifies and wishes to add new business opportunities to the project being developed
- ✓ Requirements Customer or project owner requirements are not properly delivered to the project manager
- ✓ The project team cannot carry out the planned functions due to a lack of technological knowledge, etc.
- ✓ Some new technologies or software are being launched.
- ✓ Changes in planning during the development process have a negative effect and have a negative impact on the project budget and deadlines.

The area of software development is a broad area and grows with new and future standards and technologies every day. Programming languages are introduced on the market almost every month using new versions and frameworks. New programming language changes also offer new features and technologies that facilitate and sometimes change your project everywhere. For this reason, the project manager must monitor future versions of the programming languages in order to meet the requirements of the project owner and to coordinate with the project team [3].

Designed with strong technical expertise by highly trained and trained software development teams. Highly trained and trained people also need the highest return from their work and are based on hourly or daily work. The project manager will, therefore, take into account the price costs for each business day and the estimated time to complete the project by budget control. People with high qualifications don't like joining a better team because they tend to cherish their own work and self - absorption. Although low - skilled people mean that it takes more time to complete the project and less coordination in relation to the project, but with lower employment rates. The project manager is therefore prepared to hire highly trained people in less time to prepare an effective project. The project manager must be able to use a highly competent person's ego in the best way to complete the project by coordinating it with the team with any differences between them. [4].

Members of the program development team are often coordinated from all over the world. To communicate face - to - face, software development is not required. Many online management tools are available to manage tasks and track tasks, such as base camp, pivotal tracker, product, and asana. Great exchange of files, such as Live Drive, Google Drive, Dropbox, etc. Meetings can be held online using Skype and instant messaging using WhatsApp, Viber, etc. [5]
The table below summarizes the positive and negative effects of software development projects characteristics:

TABLE .1 SOFTWARE DEVELOPMENT PROJECTS CHARACTERISTICS

Characteristics	Positive Impact	Negative Impact
Changing in planning and specification	Nil	Exceeding the budget. Development team got stress. Deadlines may also be exceeded
New technology and standards	Give new opportunities regarding design, coding, and security	Developers need more time to research new technologies.
Skilled workforce	Increase the chances of achieving innovative results	Highly skilled workforce means high pay
Global teams	Cultural creativity separately	A little bit hard to monitor

2. PROJECT STAGE

A. Stages of Software Development:

Software development is a process of different stages but related to each other. Each stage has a specific time frame in which the result is delivered. Each stage of the weight depends on the project. These stages are research, planning, design, development, testing, configuration, and maintenance. Now select them briefly. [6]

B.Research

It is the beginning of almost all software development projects. At this stage, the project owner, project manager and project team meet and exchange information about the project. The entrepreneur fulfils his or her objectives by searching for markets for those persons or organizations with similar goals to help with the budget or any other purpose, then formulating them in documentary form and then investigating a company that has the ability to achieve this goal in the time and time required budget. The project owner delivers his / her exact objectives to the project manager. The project manager is responsible for receiving the project owner's requirements in full, evaluating them and moving to the project team with technical specifications. The project manager must take care of both the business and technical perspective. The project team is responsible for meeting the requirements from a technical perspective. The project team should investigate the programming language, framework, libraries, version creation tools and infrastructure needed to build the software project according to requirements [7].

C. Planning

Planning is a stage in which elements are assembled and organized in a way to complete the program product. The large project must be divided into a small flow and easy management of subgroups. The project manager places all subsets, functions, and database at the front of the project team to focus on the appropriate technology to achieve the objective and should decide on the best management methodology for use and the protocol to be followed to complete the project, In the budget and in the range [8].

D. Design

At this point, the application design is created. Mobile and web applications make the design more effective than desktop applications. The design is entirely dependent on the nature of the project, project, function, and purpose. Like the banking application, they have less design and specific design, whereas the museum's web application needs excellent graphic designs to attract people. This stage is very important because at this stage the application design preview is displayed to the project owner so that he decides to finish it or change it. The entrepreneur comes with some will, and they must be added to the research and planning as well and after the implementation of this function in the project [9].

E. Development

The implementation of the program is already evolving this stage. This stage has two surroundings. The development and test environment always simultaneously use the same protocol. The code needs to be written in the development environment and these codes need to be loaded into the test environment using the same synchronization protocol. The main aspect of this phase is the monitoring of progress and the project manager's implementation. The project manager monitors the update progress and updates the project owner on the project progress. Developers always carry out a debugging process to help remove project errors and load error-free test environment codes. Developers also write comments during encryption and make it easier for other developers to understand them [10].

F. Testing

At this stage of programming and design, errors were found and fixed. Testing the function of each function and seeing the result find programming errors. If the output results are assumed to be incorrect or the applications fail or behave in a way that is not supposed to be, they are programming errors. Data from the application or hackers are easily stolen access to the application and a programming error also occurs. Although the project owner should be notified of the design error, the project owners know the requirements that project managers must meet and what the project team does. If an error occurs, these errors occur during the planning phase. To determine the design error, the project owner must be involved because it is the one who formulated the requirements [11].

G. Setup

The application is installed in the direct environment during the configuration phase. The actual setup includes source code, database, etc., almost everything used to compile programs where applications from third parties are required, APIs, etc. The application also undergoes a different test cycle when it is fully installed in the real environment. After testing, content is added to the application.

H. Maintenance

This phase covers the development of programs after training and the implementation of the schedule. By monitoring the firewall, mail logs, HTTP, FTP, MySQL, and SSH errors make sure the application is running properly. Monitor traffic and input data.

These early stages represent the cornerstone of all software development projects agreed upon by software development communities. Depending on the software development methodology, since phase names are changed in some methodologies, others are mixed and others overlap.

This document provides the characteristics of software development projects and their positive and negative impacts. Provide the stages of system software development. Then, briefly identify the most common programming methodologies that are currently being used. The author has submitted the comparative table to complete the document [12].

3. PREVIOUS MODELS THEORIES

The Methodology Software development process is a set of rules that are used to address all the stages specified above in order to succeed in software development. In this article, we review the most popular and popular software development methodologies 20 and show their main characteristics. Including the determination of the size or size of the projects; what is the appropriate methodology, feedback from the stage project owner and presentation of the flowchart representation of the methodologies.

A. Waterfall

It is known as the first software development methodology. The term cascading is not used in Winston W's article. Royce. The layout emphasizes carefully and the results are a lot of documentation. Each process is sequentially performed in cascade. You must complete a phase in this methodology to complete the next step. Comments on software applications received from project owners after the development and testing process has been completed. The succession sequence is suitable

for small software projects where requirements are clearly defined by the project owner and the project manager with the project team can easily and accurately plan as shown in fig.1, which is given below.

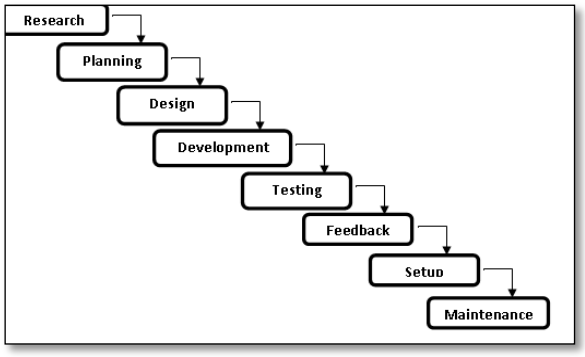


Figure 1. Waterfall Methodology [13]

B. Prototyping

In this methodology, a prototype is defined with the main or selected function to see the result and to discover what is missing in the project. In simple words, the entrepreneur creates and tests the program's specific function for comments if changes are made and then returns to the planning stage and meets the requirements. However, this does not mean that the prototype has evolved more in the real project. A prototype of a software project must be developed quickly and often ignore best programming practices [13].

In fig.2, the project owner and the project team communicate with each other in this methodology for improved results and comments. This methodology is mainly suitable for large software projects as well as for new innovations and software projects of this kind, which have not previously been developed.

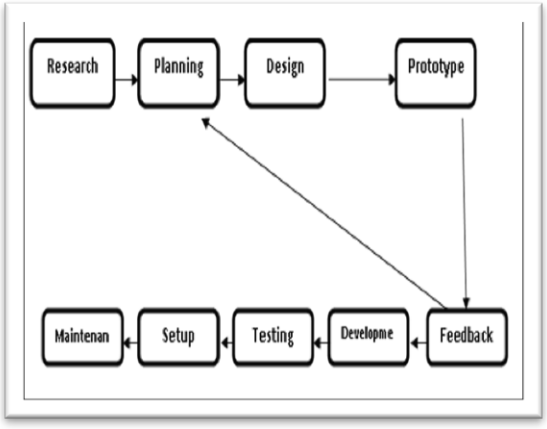


Figure 2.Prototyping Methodology [14]

C. Iterative and Incremental

In this software application methodology, one-step is built each time on the development form in the form of expanding this model. The initial specifications are created, and you receive feedback from the project owner if no problem is found, and go to the following program specifications. It differs from the prototype; the model is designed so that it is not useless, adds additional specifications, and then receives feedback from the project owner. This process continues until the actual application is developed.

The development process of each model is received, called repetition and responses of the project owners after the completion of each repeat process. In the fig.3 show the methodology that focuses on design documents and is suitable for medium and large enterprises [14].

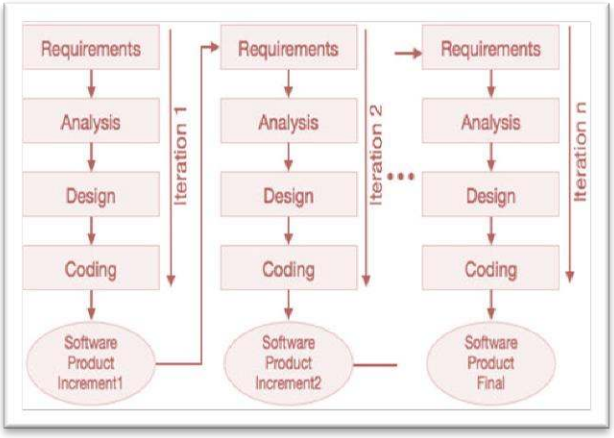


Figure 3. Iterative and Incremental Methodology [14]

D.Spiral

This methodology focuses on setting goals and analyzing other useful options for the best-documented projects [15]. The methodology of the spiral has four stages: planning, risk analysis, development, and assessment. The project follows several times each stage to reach the final stage in which program configuration is carried out in a real environment as shown in fig.4. The risk analysis phase uses several options before the addition of the program continues. Upon completion of the first repetition, project owners receive comments. This methodology is suitable for projects that need to identify risks and is also suitable for medium and large projects [15].

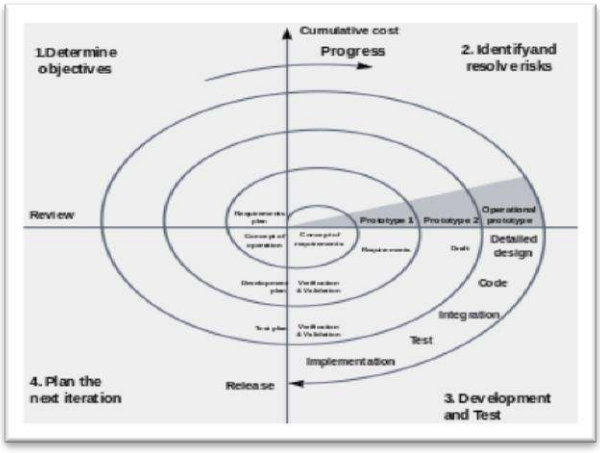


Figure 4. Spiral methodology [15]

E. Rapid Application Development

This methodology develops a life cycle design to provide rapid development with the high quality compared to old and long methods. Its design is to take advantage of the excellent opportunity for a strong program of development [16].

This method is less focused on planning and focuses more on development. Several development cycles can, therefore, be created simultaneously. Every cycle has two development and test phases, which are called modules. Comments from project owners are received after each unit has been completed. This methodology is suitable for small, medium and large enterprises, but make sure that the project must be divided into units. This methodology is suitable for small, medium and large companies, but it is important to divide the project into units as shown in fig.5 [16].

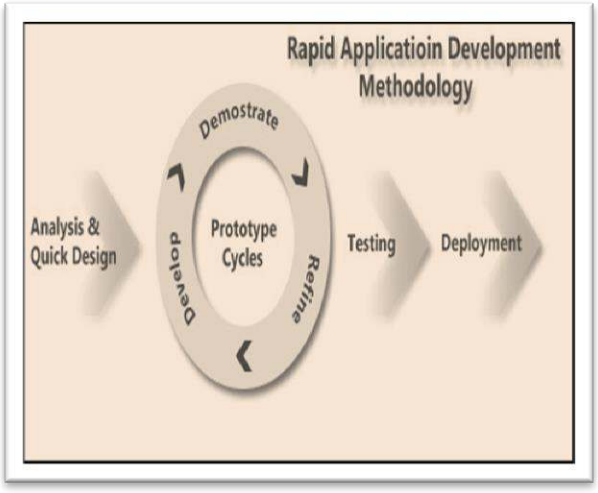


Figure 5. Rapid Application Development Methodology [16]

F. Extreme Programming

This method breaks the software development process into small parts in order to manage them back into the actual process. Rather than planning, designing, and developing complete software when dividing specifications, they reduce the cost of changing the program to do all of these activities bit by bit throughout the development process [17].

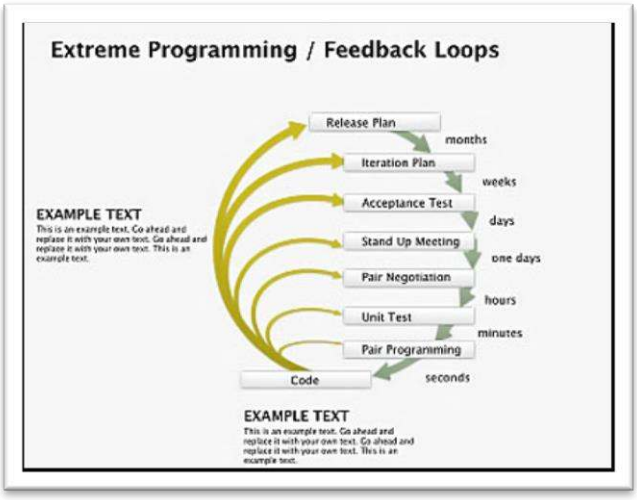


Figure 6. Extreme Programming Methodology [17]

The tasks are separate from the involvement of any programmer, even if the program is not written. The code is written and the codec is viewed like two developers using the same computer. Project owners can easily add new requirements to the process by using this method. This is almost the same as the agile process of development. The project owner often receives feedback from the development team and ongoing cooperation. It also suits small, medium and large enterprises as shown fig.6 above.

G.V-Model

This methodology is an extension of the process development model of sequential software. It focuses on tests that combine each phase with the same testing phase. In fig.7 mention, the project owners' opinion is received after the full development of all programs in the form of acceptance tests as is more suitable for small and medium enterprises [18].

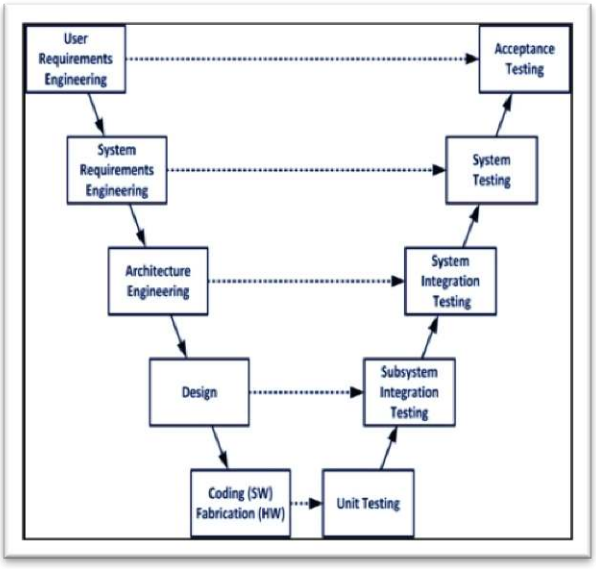


Figure 7. V-Model Methodology [18]

H. Scrum

Construction programs are carried out in a complex environment under this methodology. The software requirements created by the project owner on a priority basis are called stories. All stories make up the product portfolio.

In four weeks, not more than four weeks, this methodology develops the development cycle. Sprint Backlog is a race of all stories. The focus on progress is based on the daily meeting of 15 minutes called the Daily Scrum. A task cannot be allocated or defined by the project manager or anyone else. It is an independent Scrum development team, which makes the task a process with all the members of the team. The Master Scrum follows all operations. Feedback is received after the end of each sprint by the project owners. This methodology is suitable for small, medium and large companies of three sizes [19].

I. Cleanroom

This research of the methodology believes that the prevention of defects is much less costly than eliminating them. This methodology focuses on the prevention of defects.

The goal of the research room methodology is to build a complete program without any defects during development. This methodology is based on the structural monetary method of software design. It also uses a statistical test method and does not test any code developers because the test team will test it. After getting the positive clarity test (called the increase in general), comments will be received from the project owners. This methodology is also suitable for small, medium and large enterprises as shown in fig.8 [20].

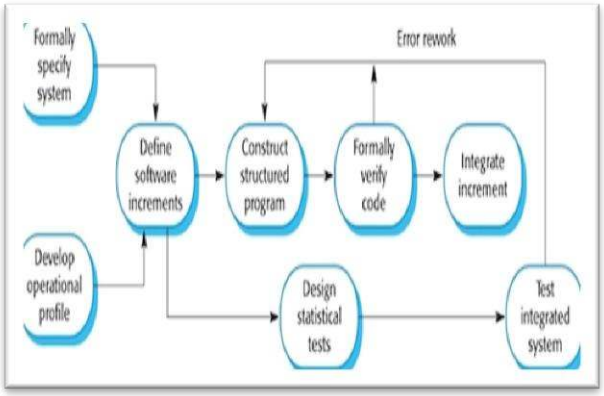


Figure 8. Cleanroom Methodology [20]

J. Dynamic System Development Methodology

It focuses on software applications, which meet the requirements of companies [14]. It also takes the time-consuming approach and priority action of Moscow. At the beginning of the project, the quality criteria are defined and fixed deadlines are set. In fig.10 mention the testing process is carried out continuously throughout the development cycle. In this methodology, project owners and the project team communicate with each other to share information at work or at all project stages. For small and medium-sized enterprises, this methodology is appropriate [14].

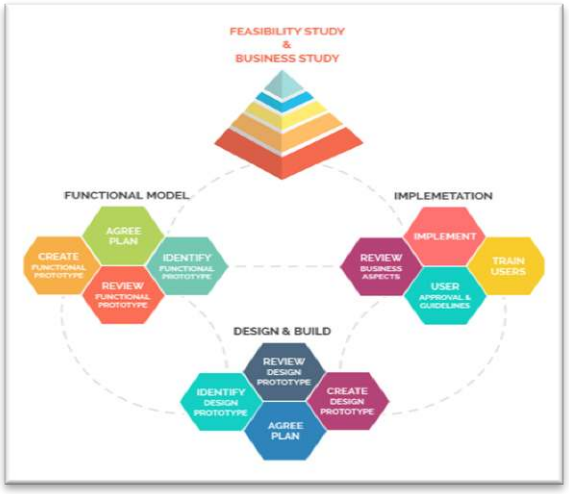


Figure 10. Dynamic System Development Methodology [14]

K. Rational Unified Process

It provides a disciplined approach to program development. It has many ready-made work plans for different types of projects and provides a guide to the entire project process. The project team does not participate in any specific task in this methodology. It

also serves as a guide to assist the project manager in adjusting the process if no ready-made work plans are adapted to the project [15]. It consists of four stages: start-up (research), development (planning and design), construction (development and testing) and transition (settings and maintenance). At the beginning of the project, the feedback of project owners is decided by cooperation between the project team and the project owner. This methodology is suitable for large, medium and small software projects as shown in fig.11 mention below.

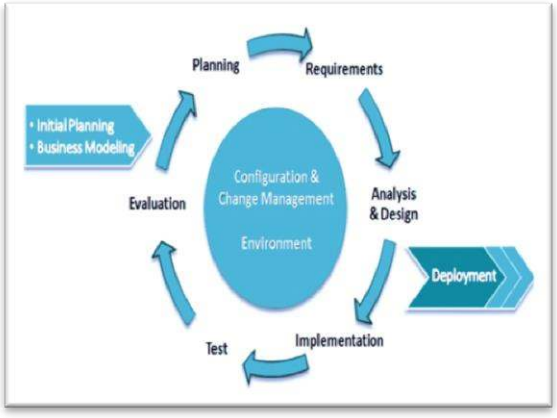


Figure 11. Rational Unified Process [16]

L. Lean Software Development

This methodology is a model for project development with a holistic approach, giving value to the project owner and eliminating waste, empowering people and improving them continuously [9]. Motivate team members to decide on the application by training them.

This methodology does not require work in a particular project building process. The project manager and project team members will freely choose the process and the time it is corrected. The project owner's notes are attached to each method. This method is also suitable for small, medium and large enterprises of three sizes.

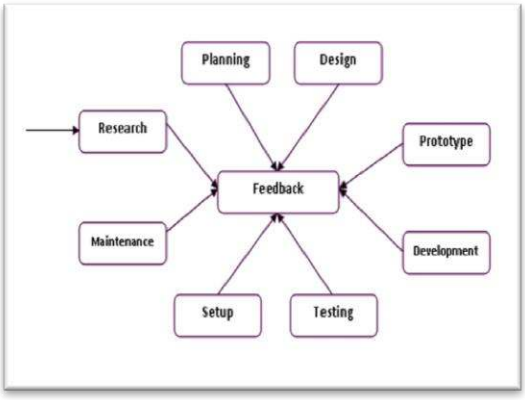


Figure 11. Lean Software Development [16]

In fig.11 show the project detail about the three sizes (small, medium and large enterprises) for Lean Software Development.

M. Test-Driven Development

On unit tests, this methodology was developed. Before writing real code, developers need to automatically write test cases for new jobs. If the test results are positive, developers will not need to write any code because the function already exists.

Usually, this is the same when it comes to inherited encryption (inheritance). If the test result is negative, the developer will have to write the code and retry. The entire process continues until all requirements are fulfilled [17]. Comments are received from the project owners after giving the positive result of the development process. This methodology is suitable for medium and large enterprises as shown in fig.12, which is given below [17].

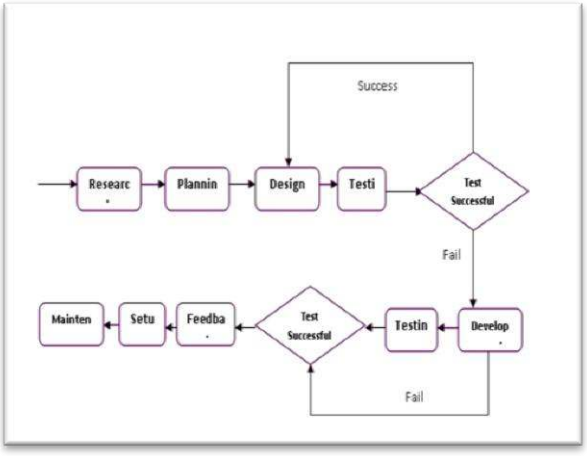


Figure 12. Test Driven Development [17]

N. Behavior-Driven Development

The project owners in a standard form called an acceptance test write all requirements. This method is based on acceptance tests. The acceptance test is defined as stories, which include a title, a narrative, and criteria for acceptance [18]. The developers will implement the function by focusing on acceptance tests. You will attempt to use the same acceptance test criteria after you have developed the function. I have a positive outcome, and the test code goes into life. The whole process is repeated until all demands are fulfilled. The project owner's opinion is received following the positive results of the code tests. This methodology is also appropriate for small, medium-sized and large companies [19].

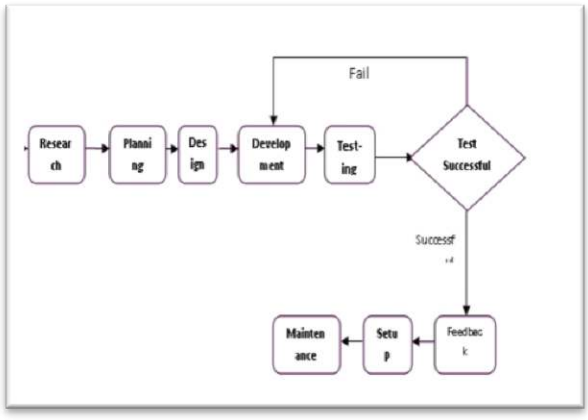


Figure 13. Behaviors Driven Development [20]

Fig.13 shows that the methodology appropriate for small, medium-sized and large companies of Behavior-Driven Development model.

O. Feature-Driven Development

This methodology concentrated on the real functionality of the software project required. Each feature of this methodology is an understandable requirement for the project owner, has a real commercial meaning and describes the true value of the work [21]. During the project, the project owner and the development teams interact constantly. Get feedback from the project owner when you set up the app settings. This methodology is suitable for three small, medium and large firms as shown in fig14 [22].

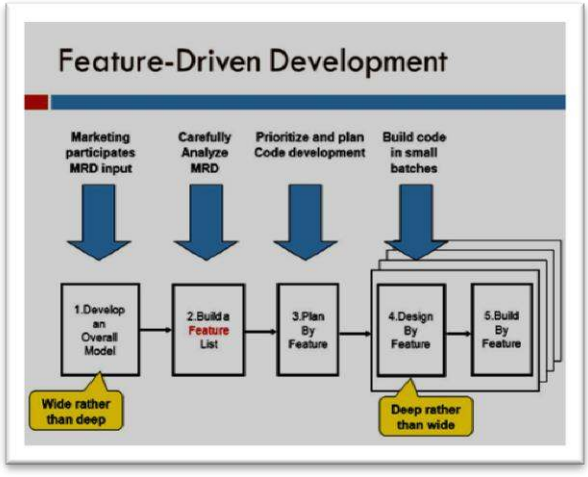


Figure 14. Feature Driven Development [22]

P. Model-Driven Engineering

The requirements of the contractor are specified in this methodology in the Metamodel. The definition form is defined on the basis of specific requirements. This is a complex methodology. Models are used as a means of meeting demands [23]. The metamodel is an independent model platform that can be adapted to any environment or migrated. UML is usually used to build a metamodel. The meta-model will then become a specific development platform model. The actual code is then written on the basis of these forms. Following positive results for the code tests, the opinion of the project owner is received. This methodology is appropriate for small, medium-sized and large companies [24].

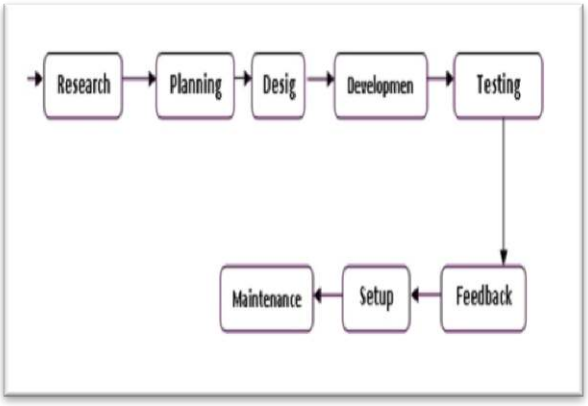


Figure 15. Model-Driven Engineering [25]

In fig.15 shows that the Model-Driven Engineering results for the code tests opinion for project owner after implement the model

Q. Crystal Methods

It is also a member of a family of methodologies that focus on people and give importance rather than tools or process.

Crystal methods include many methodological elements and do not deal with all projects in the same way, but use custom processes and tools according to the nature of the project. A project that requires security or large project needs more elements of methodology and small projects that need some elements of the methodology. In the Crystal methodology, FAO develops and uses only those methodologies required by its work or projects [19]. It is also an iterative approach but it does not apply at all with every iterative. Receive feedback from the project owner after each end of the repeat. It is appropriate for small, medium and large firms. The approach depends on the project's size as shown in fig.16 [26].

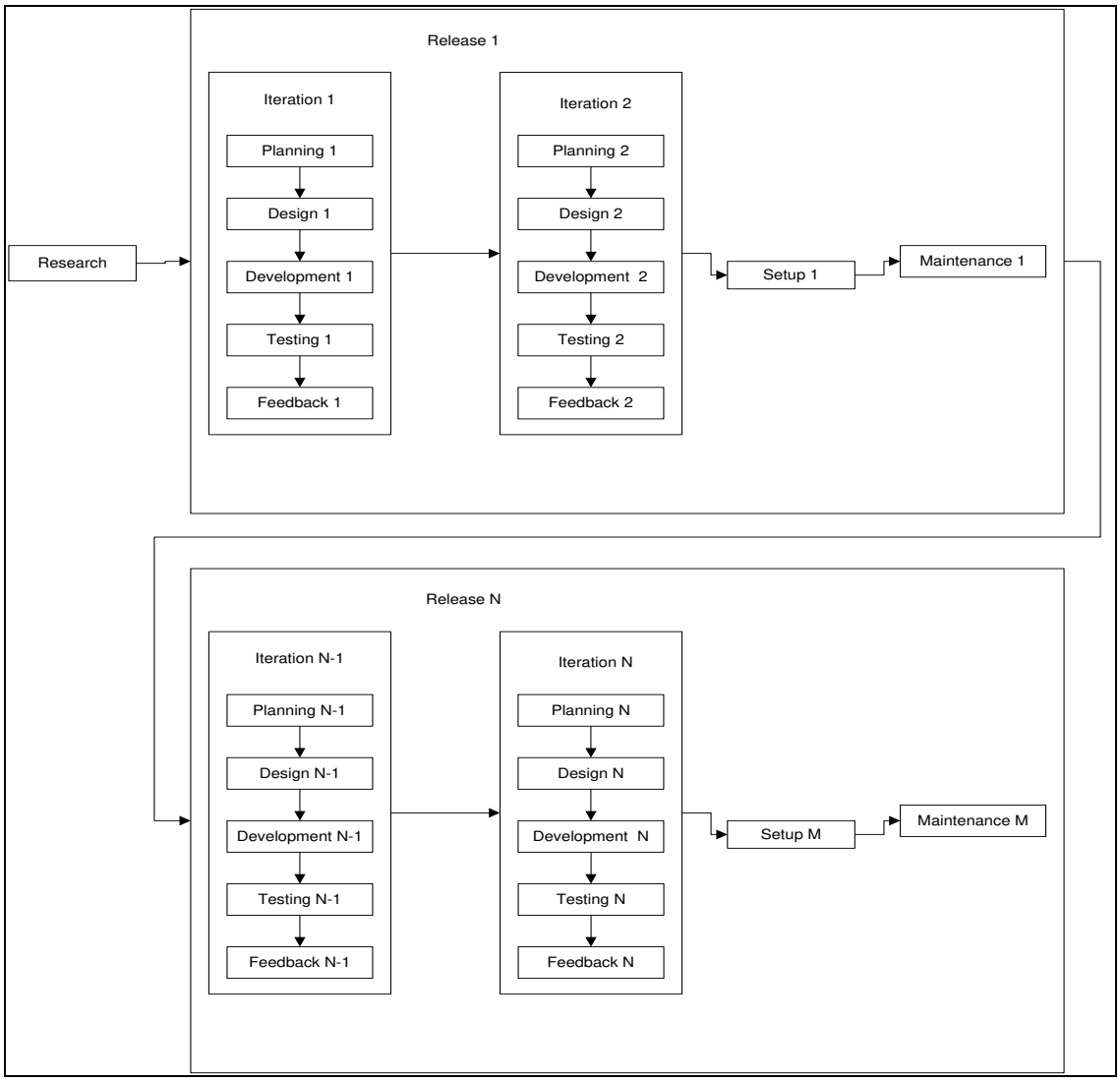


Figure 16. Crystal Methodology [27]

R. Joint Application Development

This methodology focuses on system requirements through the participation of project owners, project team and end-users in a free interaction meeting [28].In the design and development phase, the project manager and the project team participate significantly. This methodology also uses the prototype for real software development. Comments received from project owners at each JAD meeting and after the completion of the prototype. This methodology is suitable for medium and large enterprises as shown in fig.17.

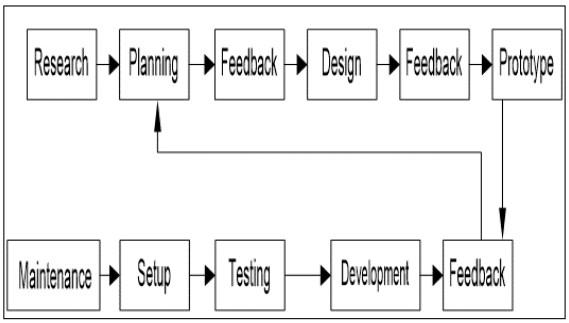


Figure 17. Joint Application Development [29]

S. Adaptive Software Development

This methodology is also based on the repetitive development and continues until the project is successful. It is a methodology constructed in response to an economy that is changing and developing more and more [30].

Accept changes and value presented throughout the project. It also responds to and accepts risks and manages them. Comments are received after each repeat is completed. This methodology is suitable for small, medium and large enterprises [31].

T. Open Source Software Development

It is a "decentralized methodology without a central authority, the owner of the project, without compensation to the project team, without responsibility, however, with a high success rate." [32]

However, only open source software or open source code is publicly available. With an open source license to study, modify and design. Thousands of programmers who work, test and test programs without expecting any direct compensation do not have these programmers face-to-face. All methodological phases are combined and three phases are generated: initiative (Research,

Planning and design), implementation (Testing and Development) and research (Maintenance and Settings). No comments on the development of open source software provided by the project owner. This methodology is suitable for small, medium and large companies [33].

U. Microsoft Solutions Framework

In fig.18 shows that the deliberate and disciplined approach to technical projects based on Microsoft's guiding principles, models, disciplines, concepts and proven practices. Versions of this methodology exist. These applications are light and heavy. It also opens the link and authorizes members of the team, but at the same time, it is clear. After publication, comments are received. It is suitable for small and medium-sized enterprises [34][35].

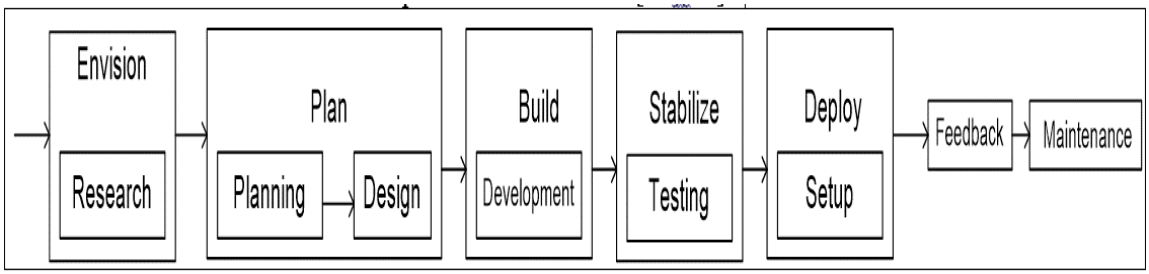


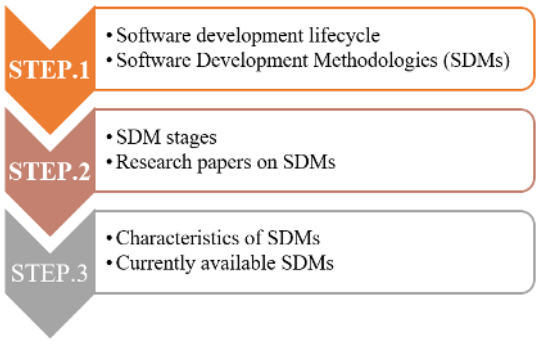
Figure 18. Microsoft Solution Framework [25]

4. RESEARCH METHODS

A. Method of Data Collection

A systematic review of available literature on the Internet for different software development methodologies will be conducted. Following online databases will be explored to search literature. To achieve results author follow the Qualitative Research methodology & gathered the data by using these three Ethnographic Research, Content Analysis & Case Study techniques. By the help of these, the author achieves objectives on the base of observations, analysis, and documents, rather than rely on a single data source.

A literature search will include the following keywords.



B. Sampling Technique/ Dataset Description

All available publications will be reviewed easily and comprehensively to extract widely used software development methodologies. These selected software development methodologies will be compared with the following aspects.

C. Sample size /Sampling Technique

One hundred and twenty-five hundred and fifty (125-150) articles will be reviewed by articles and books reviewed. The most common and most common software development methodologies will be selected, which are 21 based on literature review and will be compared to the above criteria.

D. Instrument/Software of Data Collection

✓ Computer / Online research databases

E. Research Model developed

It is a comparative study of the manual evaluation of many available software development methodologies. This will help developers to choose the most appropriate methodology according to their needs, specifications, objectives, resources and time.

F. Graph showing percentage of helpful Data Gathering Sources

With the help of analysis of different Software development methodologies, the best way to select methodology for any project can be done by these factors kept in mind, size of project, cost, time and you should choose a Waterfall, Spiral, RAD as the best choices and still modify them according to the development environment and available resources.

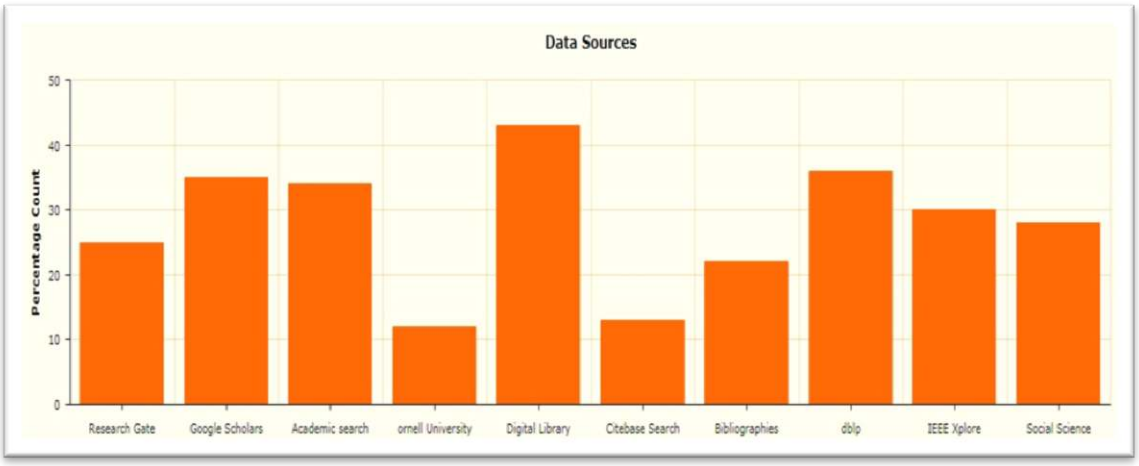


Figure 19. Graph Value of data sources size of the project, cost, time of Waterfall, Spiral and RAD

5. RESULTS

By review of different research papers, the comparative study is done of different methodologies by strengths and weaknesses as well. The below table shows the comparability of multiple SDM which is related to survey and after gathering the data of a survey of multiple software houses, the researcher creates the graph of the project cost, time of waterfall, spiral and RAD value data as these model are usually work on multiple SDM at international level. The comparison of SDM is based on a survey and collect the source of data is an online Google form.

TABLE 2. COMPARATIVE RESULT OF MOST COMMON TWENTY-ONE SOFTWARE DEVELOPMENT METHODOLOGIES

Methodology	Characteristics	Strengths	Weaknesses
Waterfall	1. Full documentation with careful planning 2. Process is linear 3. Every step has its own deliverables	1. Simple and fully described steps 2. Simple to manage even large projects 3.Easy to understand by everyone	1. Code of the project delivered late 2. Does not manage well when new requirements are required 3. Low helping in design and planning errors
Prototyping	1. Na number of the demo version of software products built in it. 2. Project owner fully involved 3. It valued to coding not writing specifications.	1.Perfect identification of application requirements 2. Project owner give early feedback 3. Early find if missing functionality	1.Increase the application's complexity 2.Increased time in programming 3.Increase cost due to generating a prototype
Iterative & Incremental	1. Project owner fully involved. 2. No. of iteration build in it as an initial model. 3.Highlights design over documentation	1. Project owner give feedback continuously 2. Multiple revisions are done in the entire project 3. Coding delivered early in the project	1. Each iteration seems to be inflexible like a small-scale waterfall project.
Spiral	1.Divided into four major phases 2.Attention on objectives and alternatives 3.Highlight risk analysis 4.Calculate multiple alternatives before the planning stage	1. Early project code delivered 2. Due to focus on risks its minimize the risks 3. Make excellent documentation	1. Mostly cost spend on risk handling 2. Without accurate risk analysis, it can't continue
Rapid Application Development	1.Focus on development 2.working & complete in fixed time 3.working is so fast	1. Everything developed very fast 2.Reusable code	1. Documentation is poor because of speedy 2. Development cost increased 3.Working with different modules at the same time
Extreme Programming	1. Project owner decide which task should be started first 2.Speed result release 3.Unit testing 4. Project owner contact continuously like working on-site	1. Everything gets fast 2.Fast releasing of working code 3. Due to repetition bugs are reduced 1. Get feedback continuously from the project owner	1. Documents are lack 2.developers unwilling to do pair programming 2. Programmers are not willing to write tests first before coding 3. Frequently meeting required
V-Model	1. In every development, stage testing is also done 2.Attention on the significance of maintenance	1. Bugs are usually less 2.Simply understandable by everyone	1. It focuses on the starting stage of specification 2. Easily harmed
Scrum	1.Iterative Development 2.Daily bases meeting held know as Scrum 3. The development team is self-organized	1. Products deliver in short time 2. Feedbacks are fast from the project owner 3. Quick requirement changing is done	1.The Need for experienced developers 2.Short of documentation 3.It's hard to estimate the cost at the beginning of the large project

	4. Logs manage tasks. 5. Tasks are done in a time box called Sprints		
Cleanroom	1.Iterative Development 2.The structure is based on the box method 3.Quality control used mathematical models 4.Testing is done by the statistical approach	1.Satisfied lower the bugs rate 2.Excellent software quality products made	1.Cost of development increased 2.Marketing time of software product increased 3.The developer must be highly qualified &experienced
Dynamic System Development Method	1.Iterative Development 2.Moscow prioritization of task 3.It uses a time box approach 4.Feedback did in every stage 5.In the beginning, the standard of quality set 6.Testing was done continuously	1.Effectively focus on business needs 2.Documentation should be complete 3. Involvement of user active	1.To cover multiple tasks it needs large no. of team 2.Highly skilled developers required
Rational Unified Process	1.Iterative Development 2.Risk handling is done in prioritize based 3.Suitable business model 4.Changeable management 5.Good testing performance	1.Accurate documentation 2.Good requirement changeable management 3.Have the power to integrate new code 4.Software components and codes are enabled to reuse	1.Professionals should be highly qualified 2.The process of development is very complex and poor
Lean Software Development	1.Iterative Development 2.Components are discarded those which is not valued in products 3.Increased learning 4.Focus on customers 5.Improvements are continuous	1.By eliminated unvalued things in products, it reduced the time and cost 2.Working code delivered before time 3.The project owner is highly motivated	1.The project depends on individual team member 2.Individual must have strong business analysis skills
Test-Driven Development	1.Highly testing based system 2.Before the start of coding, testing scenarios are developed 3.Short development cycle repeated 4.Best for debugging code	1.Speedy one debugging 2.Code quality is higher 3.Due to the continuing contribution of users with developers, it makes less defected in the end	1.Actual functionality is overlooked because tests are focused on system 2.It requires more code than any other methodology 3.The only developer has done testing 4.Because of unit testing, it increases the code
Behavior-Driven Development	1.Unit testing is done in it 2.Target is business value 3.Development& business works together	1.Maintenance is easy 2.Early discovered the issues of usability 3.The rate of defect reduced 4. New code easily integrated	1.The project owner is unwilling to write scenarios
Feature-Driven Development	1.It also iterative development 2.Features are made by breaking the application 3.Each feature should not take more than two weeks 4.To find progress, it uses milestone	1.Working can be done at the same time by multiple teams 2.Progress and report tracking best in it 3.Easily to understand	1.Codeare has done individually 2.Its iteration is not accurately defined
Model-Driven Engineering	1.As a name, it used domain models 2.Working code automatically transformed by models 3.High-level models are encapsulated of knowledge 4.highlight reused of standardized models	1.Abstractionarehigh degree 2.Productivity can be increased 3.Take less time to market 4.Maintenance cost is reduced	1.Experts are required 2.Only domain experts can read the documentation 3.It's hard to convert the modelling into implementation version
Crystal Methods Methodology	1.Not focus on the process 2.It depends upon people and skill 3.Iteration one in a release 4.Due to project sizes and criticality, it uses different approaches	1.Simply implementation 2.Deliver working code speedily 3.Developers have committed timeslots to return on possible code improvements	1. Critical decisions are structured individually; not the entire team involved in it.
Joint Application Development	1.Highlights the system requirement 2. In design and development, both project owner and end users are involved 3. JAD meeting held 4. Use prototyping	1.Designing done speedily 2.Increase the quality 3.Support teamwork with the customer 4. Maintenance cost reduced	1.Highly confident on the success of the meetings 2.documentation approach is not done in it to follow the system requirement and other steps of development

Adaptive Software Development	1. Iterative Development 2. Keep an eye on the final goal 3. Feature-based 4. Time-based 5. Risk is driven	1.Helpful for change and scope creep 2. Simply to understand 3.Enables innovation	1. Risk handling is lower 2. Assumption and predication used 3.Require solid documentation
Open Source Software Development	1. Iterative Development 2. Teams can be work from around the world 3. Work is done by sharing	1. Costs are low 2.Excellent dedicated developers 3.Testing done by large no. of developer's reviews	1.Less responsibility for submitting code 2.No main management authority 3. Development approach is unstructured
Microsoft Solutions Framework	1. Working with both lightweight and heavyweight implementation 2.Advance communication 3.Authorize the team members and create clear responsibilities and share it	1.Maintain multiple process approaches 2. Risks are handled strongly 3. Simply and easily to change and built 4. Team size reduced	1. Configuration and setup is difficult

Review papers also do a comparative study of methodologies by project size or scales. Below table shows the comparability of methodologies; which methodology works best in which size.

TABLE 3.METHODOLOGIES REGARDING PROJECT SIZES

Methodologies	Best for Project Size of
Waterfall	Small
Prototyping	Large
Iterative and Incremental	Medium and Large
Spiral	Medium and Large
Rapid Application Development	Small, Medium and Large
Extreme Programming	Small, Medium and Large
V-Model	Small and Medium
Scrum	Small, Medium and Large
Cleanroom	Small, Medium and Large
Dynamic Systems Development Methodology	Medium and Large
Rational Unified Process	Small, Medium and Large
Lean Software Development	Small, Medium and Large
Test-Driven Development	Medium and Large
Behavior-Driven Development	Small, Medium and Large
Feature-Driven Development	Small, Medium and Large
Model-Driven Development	Small, Medium and Large
Crystal Methods	Small, Medium and Large
Joint Application Development	Medium and Large
Adaptive Software Development	Small, Medium and Large
Open Source Software Development	Small, Medium and Large
Microsoft Solutions Framework	Small, Medium and Large

Common practices and standards differences found through papers study and other research resources will notify that there are many lacks in Pakistani Development Environment which can be made the effect on developed product and also harmful for the life of Software houses.

TABLE .4. COMPARISON CHART OF COMMON PRACTICES OF INTERNATIONAL & PAKISTANI DEVELOPMENT ENVIRONMENT STANDARDS

International Standards / Practices	Pakistani Standards / Practices
Understand requirements, set frames and involve different key roles in Requirements Gathering phase. E.g. Development Lead and QA	Do not understand requirements fully and start developing structure, not pay focus to set frames and involve different key roles in Requirements Gathering phase just higher authorities attend the meetings and made commitments.
Prepare Documentation & User manuals properly as according to the SDLC directions.	Not pay focus to prepare Documentation & User manuals properly as according to the SDLC directions.
Adopt new Technologies & held Training, Seminars, Workshops, Conferences held to and be familiar with new system changes.	Do the things in a traditional way and not easily adopt new changes, if adopt not have enough dedication, motivation, and directions to focus and be familiar with changes.
Follow Coding Standards & add proper comments for easy to understand.	Follow Coding Standards but not fully in average cases & not add proper comments for easy to understand.

Employees are remains in their domain and not interrupt each other pay fully focuses to generate a useful product.	Employees fast shuffling made workload on other resources which made interruption and disturb focuses which made an effect on results and overall on the efficiency of the product.
Critical situations, Major problems or delays solutions will found by Sessions and discussions with experienced resources and not do bypass from the situations.	Critical situations, Major problems or delays will replace by alternate and bypass in average case to meet with the deadlines due to lack of resources.
Followed Proper chain work.	Did not follow proper chain work in average cases.
Changes are welcome according to SDLC rules.	Changes and rework face resistance due to the environment and team fast shuffling.

6. CONCLUSION & RECOMMENDATIONS

A. Conclusion

There are two main philosophies: light and heavyweight(1) Heavyweight methodologies are suitable for projects, which do not want to change their requirements and allow a detailed complexity of the project. These methodologies are easy to understand and implement. It makes easy to understand because of complete documentation. Project manager easily tracks the project because reporting is done on time. The project owner only participates in the research and planning stage. (2) Light methods are suitable for projects whose requirements are not clearly defined and which can be changed by internal or external factors. Easily deliver the working code, self-organized team, and adaptive planning. The project owner is highly involved in the project to give fast feedbacks. This paper defines the comparability of methodologies as strength and weaknesses, which are also appropriate for small, medium and large- scale projects.

B. Recommendations

On the base of above deep analysis of software development models and by the judge the nature of project first, we can summarize software development models into categories and then chose the best model for the project from the suitable category. Analysis base for top categories and their selected models are as under.

- Flow-Based Model
  - Waterfall model
  - Iterative waterfall model
- Structured Based Model
  - Spiral model
  - V model
- Iteration Based Model
  - Prototype model
  - Evolutionary model
  - RAD model

The methodology should be selected by viewing the size of the project, cost and time. Always try to hire experts to complete the project. By reading this paper, it clearly shows that the methodology can be made according to the

project nature or need. If any new innovation is required as per project may be project manager creates the new methodology or view the all methodologies to accomplish the project.

ACKNOWLEDGEMENT

Authors are grateful to the Department of software engineering at Universiti Teknologi Malaysia-UTM.

REFERENCES

[1] Pagliari, C. (2017), "Design and evaluation in eHealth", Journal of Medical Internet Research, Vol. 9 No. 2, pp. 1-14.

[2]Dahl, Y., Farshchian, B., Vilarinho, T., Helbostad, J.L., Nawaz, A., Nygård, A.J. and Wik, P.B. (2016), "Stakeholder attitudes toward and values embedded in a sensor-enhanced personal emergency response system", Interacting with Computers, Vol. 28 No. 5, pp. 598-611.

[3]MojtabaVaismoradiPhD, MScN, BScN, HanneleTurunenPhD, RN, TereseBondasPhD, RN, 11 March (2013), Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study. https://doi.org/10.1111/nhs.12048

[4] Torgeirdingsøyr,sridharnerurcvenugopalbalijepally,nilsbredemoe, (2012), a decade of agile methodologies: towards explaining agile software development. https://doi.org/10.1016/j.jss.2012.02.03

[5] Matthew Kearse Richard Moir Amy Wilson Steven Stones-Havasmatthew Cheung Shane Sturrock Simon Buxton Alex Coopersidney Markowitz Chris Duran, June 2012

[6] Geneious Basic: An integrated and extendable desktop software platform for the organization and analysis of sequence data, *bioinformatics*, volume 28, issue 12, 15, pages 1647–1649. https://doi.org/10.1093/bioinformatics/bts199

[7] Drummond, A.J. et al. (2010) Geneious v5.5. Available at http://www.geneious.com.

[8] Goecks J., et al. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences, Genome Biol. , 2010, vol. 11 pg. R86.

[9] P. Abrahamsson, K. Conboy, X. Wang'Lotsdone, more to do': the current state of agile systems development research European Journal of Information Systems, 18 (2009), pp. 281-284.

[10] S.T. Acuna, M. Gomez, N. JuristoHow do personality, team processes and task characteristics relate to job satisfaction and software quality? Information and Software Technology, 51 (2009), pp. 627-639.

[11] Agarwal, R. Shankar, M.K. TiwariModeling the metrics of the lean, agile and leagile supply chain: an ANP-based approach European Journal of Operational Research, 173 (2006), pp. 211-225.

[12]E. Arisholm, H. Gallis, T. Dyba, D.I.K. SjobergEvaluatng pair programming with respect to system complexity and programmer expertise IEEE Transactions on Software Engineering, 33 (2007), pp. 65-86.

[13] T. DybåSpecial section on best papers from XP2010 Information and Software Technology, 53 (2011), pp. 507-508.

[14] Falessi, G. Cantone, S.A. Sarcia, G. Calavaro, P. Subiaco, C. D'Amore Peaceful coexistence: agile developer perspectives on software architecture, IEEE Software, 27 (2010), pp. 23-25.

[15] N.B. Moe, T. Dingsøyr, T. DybåA teamwork model for understanding an agile team: a case study of a Scrum project, Information and Software Technology, 52 (2010), pp. 480-491.

[16] N. Salleh, E. Mendes, J. GrundyEmpirical studies of pair programming for CS/SE teaching in higher education: a systematic literature review IEEE Transactions on Software Engineering, 37 (2011), pp. 509-525.

[17] Grünloh, C., Walldius, Å., Hartmann, G. and Gulliksen, J. (2015), "Using online reviews as narratives to Evoke designer's empathy", in Abascal, J. et al. (Eds), INTERACT, Springer, Switzerland, pp. 298-315.

[18] P. Sfetsos, I. Stamelos, L. Angelis, I. DeligiannisAn experimental investigation of personality types impact on pair effectiveness in pair programming, Empirical Software Engineering, 14 (2009), pp. 187-226.

[19] HallewellHaslwanter, J.D. and Fitzpatrick, G. (2016), "Why do few assistive technology systems make it to market? The case of the HandyHelper project", Universal Access in the Information Society, Vol. 16 No. 3, pp. 755-73. doi: 10.1007/s10209-016-0499-3.

[20] Fereday, J., & Muir-Cochrane, E. (2006). The role of performance feedback in the self-assessment of competence: A research study with nursing clinicians. Collegian, 13(1), 10-15.

[21] Masters B.C., et al. Species Delimitation - a Geneious plugin for the exploration of species boundaries, Mol. Ecol. Resour. , 2011, vol. 11 (pg. 154-157).

[22] C. Ebert, P. Abrahamsson, N. Oza, "Lean Software Development", IEE Software, vol. 29, no. 5, pg. 22-25, 2012.

[23] M. Soeken, R. Wille, R. Drechsler, "Assisted Behavior Driven Development using Natural Language Processing", Proceedings of the 50th International Conference on Objects, Models, Components, Patterns, TOOLS 2012, 29-31 May 2012, Prague, Czech Republic, Publisher: Springer Berlin Heidelberg, 2012, pg. 269-287. doi: 10.1007/978-3-642-30561-0_19

[24] Conboy, K., Coyle, S., Wang, X., Pikkarainen, M.: People over process: key people challenges in agile development. IEEE Softw. **99**, 47–57 (2010).

[25] J. Bezivin, Model Driven Engineering: An Emerging Technical Space, Internation Summer School, Summer School on. Generative and Transformational Techniques in Software Engineering, 4-8 Jul. 2005, Braga, Portugal, Publisher: Springer Berlin Heidelberg, pg. 36-64. doi: 10.1007/11877028_2.

[26] Turner, K.J. and McGee-Lennon, M.R. (2013), "Advances in telecare over the past 10 years", Smart Homecare Technology and TeleHealth, Vol. 1, pp. 21-34.

[27] D. J. Anderson, Feature-Driven Development, Microsoft Corporation, Oct. 2004.

[28] Peek, S.T.M., Wouters, E.J.M., van Hoof, J., Luijkx, K.G., Boeije, H.R. and Vrijhoef, H.J M. (2014), "Factors influencing acceptance of technology for aging in place: a systematic review", International Journal of Medical Informatics, Vol. 83 No. 4, pp. 235-48.

[29] J. A. Livermore, "Factors that Impact Implementing an Agile Software Development Methodology", Proceedings of IEEE SoutheastCon, SECON 2007, 22-25 Mar. 2007, Richmond, USA, Publisher: IEEE, 2007, pg. 82-86. doi: 10.1109/SECON.2007.342860,

[30]Sallinen, M., Hentonen, O. and Karki, A. (2015), "Technology and active agency of older adults living in service house environment", Disability and Rehabilitation: Assistive Technology, Vol. 10 No. 1, pp. 27-31.

[31] E. W. Duggana, C. S. Thachenkaryb, "Integrating Nominal Group Technique and Joint Application Development for Improved Systems Requirements Determination", Information & Management, vol. 41, no. 4, pg. 399-411, 2004. DOI: 10.1016/S0378-7206(03)00080-6

[32] Ambler, S.: Agile adoption strategies: survey results. http://www.ambysoft.com (2013)

[33] E. M. Maximilien, L. Williams, "Assessing test-driven development at IBM", Proceedings 25th International Conference on Software Engineering, ICSE 2003, 3-10 May 2003, Portland, USA, Publisher: IEEE, 2003, pg. 564-569. doi: 10.1109/ICSE.2003.1201238

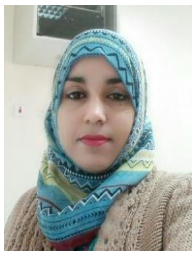
[34] S. Mathur, S. Malik, "Advancements in the V-Model", Internation Journal of Computer Applications, vol. 1, no. 12, pg. 29-34, 2010. doi: 10.5120/266425

Ms Soobia Saeed is working as an Assistant Professor, Head of publication Department, and Coordinator of Seminars and Training at Institute of Business & Technology-IBT, Karachi, Pakistan. Currently, she is a PhD Scholar in software engineering, from University Teknologi Malaysia-UTM, Malaysia She did MS in Software Engineering from Institute of Business & Technology- IBT, Karachi, Pakistan, and Masters in Computer Science from Instituto of Business & Technology-IBT, Karachi, Pakistan and Bachelors in Mathematical Science from Federal Urdu University of Art, Science & Technology (FUUAST), and Karachi, Pakistan. She is a farmer research Analytic from University Teknologi Malaysia and supervises ICT & R and D funded Final Year Project (FYP).

Noor Zaman has completed his PhD. in IT from University Technology Petronas (UTP) Malaysia. He has 19 years of teaching and administrative experience internationally. He has an intensive background of academic quality accreditation in higher education besides scientific research activities, he had worked for academic accreditation for more than a decade and earned ABET accreditation twice for three programs at College of computer sciences and IT, King Faisal University Saudi Arabia. He also worked for National Commission for Academic Accreditation and Assessment (NCAAA), Education Evaluation Commission Higher Education Sector (EECHES) formerly NCAAA Saudi Arabia, for institutional level accreditation. He also worked for the National Computing Education Accreditation Council (NCEAC) Pakistan. He has experience in teaching advanced era technological courses including, Mobile Programming (Android), Mobile Computing and. Net Framework programming besides other undergraduate and postgraduate courses, graduation projects and thesis supervision.

Dr Noor Zaman has authored several research papers in ISI indexed and impact factor research journals\international conferences, edited 10 international reputed Computer Science area books, focused on research students, has many journals, IEEE conferences and book chapter publications to his credit. He has successfully completed more than 18 international funded research grants. He is Associate Editor, Regional Editor, Editorial board member, PC member, reviewer, Keynote speaker for several reputed international journals and conferences around the globe. He also chaired international conference sessions and presented session talks internationally. He has strong analytical, problem solving, interpersonal and communication skills. His areas of interest include Wireless Sensor Network (WSN), Internet of Things IoT, Security, Mobile Application Development, Ad hoc Networks, Cloud Computing, Big Data, Mobile Computing, and Software Engineering.

Syed Mehmood Naqvi is a Professor in the School of Applied Computing at Sheridan College, Canada. Formerly, he was Dean of Faculty of Computer Science and Information Technology at Institute of Business and Technology, Pakistan. He received PhD in Computer Application Technology from Beihang University (formerly Beijing University of Aeronautics and Astronautics), Beijing, China in 1999. He did his postdoctoral research in the area of signal processing at the University of Northern British Columbia, Canada. Syed Naqvi has more than twenty years of teaching, research, and administrative experience at various universities, colleges, and institutes in Canada, Pakistan, and the UAE. He has served as an active member of many curricula development and revision committees for undergraduate and graduates computer science and information technology programs. His current areas of research include educational technology, medical image processing, and software development methodologies and models.



Mamoona Humayun has completed her PhD. in Computer Architecture from Harbin Institute of Technology, China. She has 12 years of teaching and administrative experience internationally. She is an active reviewer for a series of journals. She has supervised various Masters and Ph.D. thesis. Her research interests include Global software development, requirement engineering, knowledge management, Cyber Security, and wireless sensor networks.