



Performance of Ten Software Development Process Models with Principles

S.Thulasi Krishna M.E., (Ph.D),

*Research scholar
Rayalaseema University
Kurnool (dist.,)
Andhra Pradesh*

Dr. S .Sreekanth, Ph.D

*Professor,
Dept of CSE
SITAMS,Chittoor(dt),
Andhra Pradesh*

Abstract:- The software development life cycle is very important in project development. A software process model is the basic framework which gives a workflow from one stage to the next, for developing any project in efficient manner. Now days we have number of process models available but which one is best model, identification is very difficult. So in this research paper I proposed Ten software development process model histories, phases, principles and performance of software development chart. This chart explores clear architecture, behavior of process model and applicability of software development process model for various projects. This is useful for developing projects in short period.

Keywords:*process model, history, phases, process model principles, performance of software development process chart.*

INTRODUCTION

Software processes performed during software Development and evolution are becoming rather complex and recourse-intensive. They involve people who execute actions with the primary goal to create quality software in accordance with the previously set user requirements and only structured, carefully guided and documented software processes can lead to the stated goal. Constant monitoring and improvement of software processes is therefore of a significant interest for organizational performing software development and maintenance. In order to improve the process an objective description and evolution of the existing process is needed.

A software development process, also known as a software development life cycle (SDLC), is a structure imposed on the development of a software product. It is often considered as a subset of system development life cycle. There are several models for such processes, each describing approaches to a variety of activities that take place during the process. Software Engineering (SE) is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches.

Various processes and methodologies have been developed over the last few decades to improve software quality, with varying degrees of success. However, it is widely agreed that no single approach that will prevent project overruns and failures in all cases. Software projects that are large, complicated, poorly-specified, and involve unfamiliar aspects, are still particularly vulnerable to large, unanticipated problems. A software development process is a structure imposed on the development of a software product. There are several models for such processes, each describing approaches to a variety of tasks or activities that take place during the process. It aims to be the standard that defines all the tasks required for developing and maintaining software. Software Engineering processes are composed of many activities, notably the following:

- Requirements Analysis
- Specification
- Software architecture
- Implementation
- Testing
- Documentation
- Training and Support
- Maintenance

Software development teams, taking into account its goals and the scale of a particular project, and have a number of well-established software development models to choose from. Therefore, even though there are number of models each software Development Company adopts the best-suited model, which facilitates the software development process and boosts the productivity of its team members. There are ten types of Model are: Waterfall model, Incremental model, Prototype model, Spiral model, V-model, Concurrent engineering model, Agile model, Build and fix model.

1. HISTORY OF WATERFALL MODEL WITH PRINCIPLES

History of waterfall model: The formal description of the method is often cited as an article published by Winston. W. Royce in 1970.

Water fall model is a sequential development approach, in which development is seen as flowing steadily downwards through several phases

Phases

Requirement Analysis:

This is the first phase of development where all the requirements Gathered and documented.

Design: In this phase all the system design is analyzed and specified like hardware, system configuration and architecture or the system.

Coding: In this phase all the development works are performed and development components or units handed over to testing team.

Test: Once the development completed, testing phase starts and in this phase we test the each unit or component and make sure the developed components are working as expected. All the testing activities are performed in this phase.

Maintenance: We always keep eye on the product and provide the entire necessary bug or issue fixes if occurs in production or reported by end users. Also time to time we keep updated the product with new updates or patches if developed or available.

The basic principles are:

1. Project is divided into sequential phases, with some overlap and splash back acceptable between phases.
2. Emphasis is on planning, time schedules, target dates budgets and implementation of an entire system at one time.
3. Thighs control is maintained over the life of the project via extensive written documentation, formal reviews, and approval/ signoff by the user and information technology management occurring at the end of most phases before beginning the next phase. Written documentation is an explicit deliverable of each phase.
4. Iterate – create a prototype, then the real system. repeat each phases using new information gained and the entire process at least once delivering the live system.

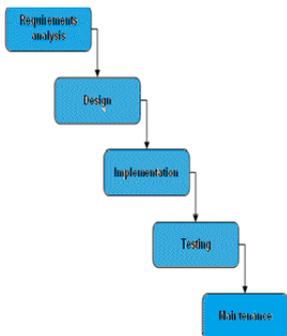
SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L Applicability Small Medium Large	CRF Cost/Risk/ Flexibility
1	WATER FALL MODEL	 <p>Water fall model has divided into six phases: Requirement Analysis Design Coding Testing Maintenance</p>	<ul style="list-style-type: none"> - Simple and easy Understand --linear sequential step by step process - Well understood Milestones - Clearly defined Stages - Provide a template into which Methods for Analysis Design Coding , testing. - Easy to Manage each phase has specific deliverable - Risk of project failure is slow 	<ul style="list-style-type: none"> - Rigid design and inflexible procedure[1]. - Restricting back's and forth movement from a stage to a former one. When new requirements surface accommodating those with existing ones become difficult due to restriction in looping back to prior stages . - poor choice for software development projects where requirements are not well known or understood by development team. -Not good for complex projects that take more time to complete. The project failure is high - Difficult in responding to changes result to high amount of risk and uncertainly. 	small projects. -Un development of database related software Example commercial projects - In development of E- commerce Website (or) portal -In development of network protocol Software	Low Budget High-risk Less Flexibility

Table 1 : Performance of Waterfall Model

2. HISTORY OF INCREMENTAL DEVELOPMENT MODEL WITH PRINCIPLES

History of Incremental Process Model:

Iterative and incremental, 1988, the article: The new product development game “ by Takeuchig nonaka prosed [2].

Different methods are acceptable for combing linear and iterative systems development methodologies, with primary objective of each being to reduce inherent project risk by breaking a project into smaller segments and providing more ease - of - change during the development process.

Phases

Requirement Analysis: Requirement and specification of the software are Collected

Design: some high –end function are designed this stage

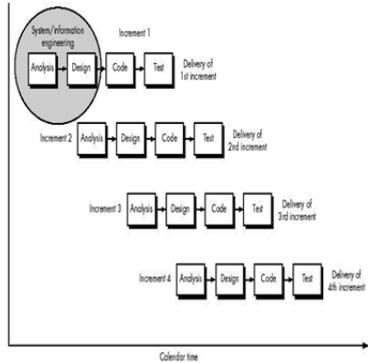
Code: Coding of software is done during this stage

Test: once the system is deployed , it goes through the testing phase

The basic principles are:

1. A series of mini- Waterfall are completed for a small part of a system, before proceeding to the next increment.
2. Overall requirements are defined before proceeding for evolutionary, Mini – waterfall development of individual increments of a system.
3. The initial software concept, requirements analysis, and design of architecture and system are defined via waterfall, followed by iterative prototyping, which culminates in installing the final prototyping a working system.

Table 2. Performance of Incremental Development Model

SL NO	M.N Model Name		A Advantages	D Disadvantages	APY S/M/L Applicability Small Medium Large	CRF Cost/Risk/ Flexibility
2	Incremental Development Model	 <p>Incremental Process Model has 4- Phases</p> <p>Analysis Design Code Testing</p>	<ul style="list-style-type: none"> - Parallel development can be planned -More flexible and less costly requirements -Progress can be measured -Testing and debugging smaller iteration is easy. -Risks are identified and resolved during an iteration and each iteration is an easily managed milestone. -It breaks down the problem into sub problem thus dealing with reduced complexity. 	<ul style="list-style-type: none"> -Each is rigid and do not overlap each other. -Each additional build has to be incorporated into the existing. -Design errors become part of system and difficult to remove. -An overhead in model is rapid context switching between various activities. Each iteration is followed by an evaluation ensuring that user requirements have been met 	Large Project	<p>High Cost</p> <p>Low Risk</p> <p>More Flexibility</p>

3. HISTORY OF PROTOTYPE MODEL WITH PRINCIPLES

History of Prototype model:

The prototype model included in REMS (Regional Economic Modeling System) is for the state of Massachusetts, us in 1989.

Software prototyping is the development approach of acting during Software development, the creation of prototype i.e. incomplete versions of the software program developed.

Phases:

Communication, Quick Plan Modeling, Quick Design:

Construction of Prototype, deployment, delivery and feedback:

The Basic Principles are:

1. Not a Standard, complete development methodology, but rather an approach to handle selected parts of a larger an approach to handle selected parts of a larger, more traditional development methodology (i.e. incremental, spiral, or rapid application development RAD)

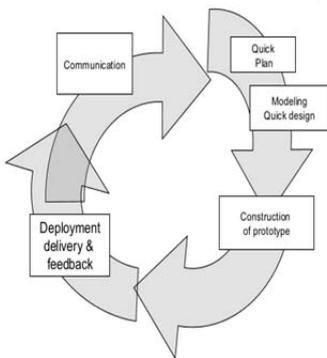
2. Attempt to reduce inherent project risk by breaking a project into smaller segments and providing more ease of change during to development process.
3. User is involved though out the development process, which increases the likelihood of user acceptance of the final implementation.
4. Small – scale mock – ups of system are developed following an iterative modification process until the prototype evolves to meet the user’s requirements.
5. While most prototypes are developed with the expectation that they will be discarded it is possible in some cases to evolve from prototype to working system.

Elaboration, from an overall concept of operation document down to the coding of each individual program.

Each trip around the spiral traverses four basis quadrants:

1. Determine objectives alternatives, and constraints of the iteration.
2. Evaluate alternative, identify and resolves risks.
3. Develop and verify deliverables from the Iteration.
4. Plan the next iteration.

Table 3. Performance of Proto type Model

SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L	CRF Cost/Risk/
3.	Prototype Model	 <p>Prototype Model Has Five Phases:</p> <ol style="list-style-type: none"> 1. Communication 2. Quick Plan 3. Modeling quick Design 4. Construction of prototype 5. Deployment delivery and Feed back 	<p>-Users/ Customers own requirements to work will be faster and efficient if developers will collaborate.</p> <p>-It is created using user feedback</p> <p>-Cost effective (development cost is reduced)</p> <p>-Increased system development speed</p> <p>-Potential risks associated with delivering of the system can be refined</p>	<p>-Includes lack of information about the exact number of iteration and the time period required to upgrade the prototype in order to bring it up to the satisfaction of the user and customer.</p> <p>-Not suitable for large applications</p> <p>-Structure of system can be damaged since may changes could be made.</p> <p>-Integration can be very difficult</p> <p>-The premature prototype lack key consideration like security, fault tolerance, distributed processing and other such key issues[4].</p>	Small To Medium	High Cost Low Risk More Flexibility

4. HISTORY OF SPIRAL MODEL WITH PRINCIPLE

History of Spiral Model:

In 1988 Barry Boehm published a formal software system development “Spiral model”.

Spiral Model: The formal software system development “spiral model”, which combines some key aspects of the waterfall model and rapid prototyping methodologies in an offer to combine advantages top-down and bottom-up concepts. It provided emphasis in a key area many felt had been neglected by another Methodology: deliberate iterative risk analysis particularly suited to large – scale complex systems.

Phases:

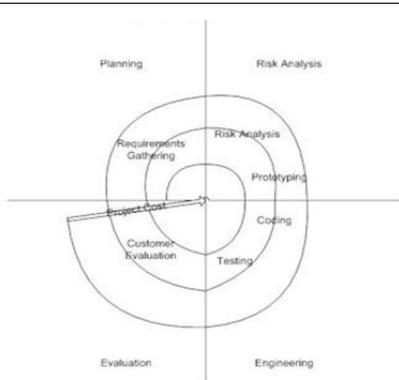
Planning Phase, Risk Analysis, Evaluation phase

The basic principles are:

1. Focus is a risk assessment and on minimizing project risk by braking a project into smaller segments and providing the opportunity to evaluate risks and Weight consideration of project continuation throughout the lifecycle.

2. “Each cycle involves a progression though the same sequence of steps, for each part of the product and each of its levels off or engineering is lesser importance.
3. Project control involves prioritizing development and defining delivery deadlines of “time boxes”.
4. If the project starts to slip, emphasis is a reducing requirement to file the time box, not increasing the deadline.
5. Generally includes point application design, where users are intensely involved in system design, via consensus building in either structured workshops, or electronically facilitated interaction.
6. Active users involved are imperative.
7. Iteratively produces production software as proposed to a throw way prototype.
8. Producer documentation necessary to facilitate future development and maintenance.
9. Standard system analysis and design methods can be fitted into this framework.

Table 4. Performance of Spiral Model

SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L Applicability Small Medium Large	CRF Cost/Risk/Flexibility
4	Spiral Model	 <p>Spiral Model Has Four- Phases:</p> <ol style="list-style-type: none"> 1. Planning 2. Risk Analysis 3. Engineering 4. Evaluation 	<ul style="list-style-type: none"> -Changing requirement can be accommodated -Allow for extensive use of prototype -Requirements can be captured more accurately. -Estimates (budget, schedule ect) becomes more realistic. -It uses a stepwise approaches development. 	<ul style="list-style-type: none"> -Risks analysis requirements high expertise. -Does not work for smaller projects. -Made separately for each applications. -Risks of not meeting budget or schedule. -Complex and difficult to follow strictly. -Applying already development product to needs of a new customer, involves high risk of making the correct product.[5]. 	Small to Medium	<ul style="list-style-type: none"> High Cost Risk Flexibility

5. HISTORY OF RAPID APPLICATION DEVELOPMENT MODEL WITH PRINCIPLES

History of RAD:

The term first used to describe a software development process introduced by James martin in 1991.

RAD is a software development methodology, which favors iterative development and the rapid construction of prototype intend of large amount of upfront planning. The planning “of software development using RAD is interleaved with writing the software itself the lack of extensive pre-planning generally a allows software to be written much faster, and makes it easier to change requirement.

Phases:

Business modeling: The information flow is identified between various business functions.

Data modeling: Information gathered from business modeling is used to define data objects that are needed for the business.

Process modeling: Data objects defined in data modeling are converted to achieve the

Business information flow to achieve some specific business objective. Description are identified and created

for CRUD of data objects.

Application generation: Automated tools are used to convert process models into code and the actual system.

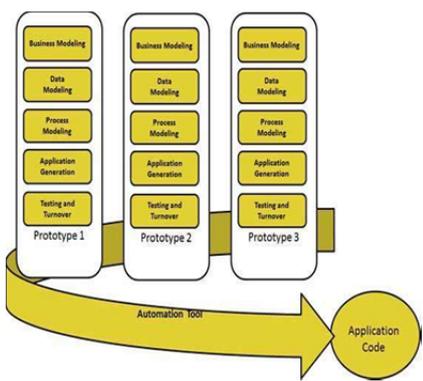
Testing and turnover: Test new components and all the interfaces.

The basic principles of RAD are:

1. Key objectives is for fact development and delivery of high quality system a relatively low investment cost.
2. Attempts to reduce inherent project risks by breaking a project into smaller segments and providing more ease of change during the development process.
3. Aims to produce high quality system quickly, primarily via iterative prototyping (at any stage of development), active users involvement, and computerized development tools. These tools may include (GUI) Graphical users Interface builders Computer Aided Software Engineering (CASE) tools, Database management system (DBMS), fourth generation programming languages, code generation, and object oriented techniques.
4. key emphasis is on fulfilling the business need, while a basic understanding of the fundamental business problem necessary to avoid solving the wrong problems.

Performance of rapid application development model

Table 5. RAD Model

SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L Applicability Small Medium Large	CRF Cost/Risk/Flexibility
5	Rapid Application Development Model	 <p>5- Phases</p> <ul style="list-style-type: none"> Business Modeling Data Modeling Process Modeling Application Modeling Testing And turn Over 	<ul style="list-style-type: none"> -Time to delivery is less. -Changing requirements can be accommodated -Progress can be measured [6]. -Productivity with fewer people in short time. -It makes an overlap reduction in project risk. 	<ul style="list-style-type: none"> -Cost of product is not known. -It is difficult for users to commit the time required for success of the RAD process. -Difficult to reuse the module for future system. -Lack of scalability high cost of commitment by the user. 	Small to Medium	<ul style="list-style-type: none"> Low Cost Technical risks are low [9]. High Flexibility

6. HISTORY OF RATIONAL UNIFIED PROCESS MODEL WITH PRINCIPLES

History of Rational unified process model:

- In 1997, Jacobsen Booch Runbay – Umphase
- Rational unified process model is an iterative software development process frame work
- Created by the rational software cooperation, a division of IBM since 2003 [3].
- Rational unified process model is not a single concrete prescriptive process but rather an adaptable process frame work, intended to be tailored by the development organizing and software teams that will select the elements of the process that are appropriate for their needs .

Phases:

1. **Inception** - The idea for the project is stated. The development team determines if the project is worth pursuing and what resources will be needed.
2. **Elaboration** - The project's architecture and required resources are further evaluated.

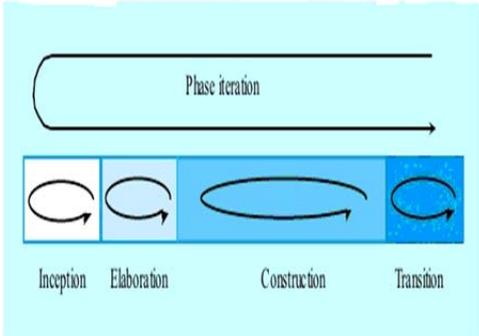
Developers consider possible applications of the software and costs associated with the development.

3. **Construction** - The project is developed and completed. The software is designed, written, and tested.
4. **Transition** - The software is released to the public. Final adjustments or updates are made based on feedback from end user.

Principles are:

1. Develop iteratively, with risk as primary iteration driver
2. Manage requirements
3. Employ a components – based Architecture
4. Model Software visually
5. Continuously verify quality
6. Control changes.

Table 6. Performance of Rational Unified Process Model

SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L Applicability Small Medium Large	CRF Cost/Risk/Flexibility
6	Rational Unified Process Model	 <p>Rational Unified Process Model has Four Phases: Inception Elaboration Contraction Testing</p>	<ul style="list-style-type: none"> - Process details are expressed in general terms, allowing local customization - Heavy emphasis documentation . - Can embrace incremental release. - Evolutionary approach can lead to clean implementation. 	<ul style="list-style-type: none"> - Process details are expressed in general terms, providing minimal guidance and requirement local customization . - complex - Heavy documentation can be expensive. 	Small Project	Low Cost Low Risk Low Flexibility

7. HISTORY OF V-MODEL WITH PRINCIPLES

History of V-Model:

The concept of v-model was developed simultaneously, but independently, in Germany and in the United States in the late 1980's

The German V-model was originally developed by IABG in ottobrunn, near Munich, in co-operation with the feeder office for defense technology and procurement in koblenz, for the federal ministry of ministry of defense. It was taken over by the federal Ministry of the interior for the civilian public authorities' domain in Sumner 1992 [10].

Phases:

Requirements, The high-level design (HLD),The low-level design (LLD), The implementation, Coding .

V-Model principles:

-The following principles are inherent when the v model is applied Large to Small – This principle stated requirements, standards, Testing from a hierarchical perspective.

Performance of V-Model model

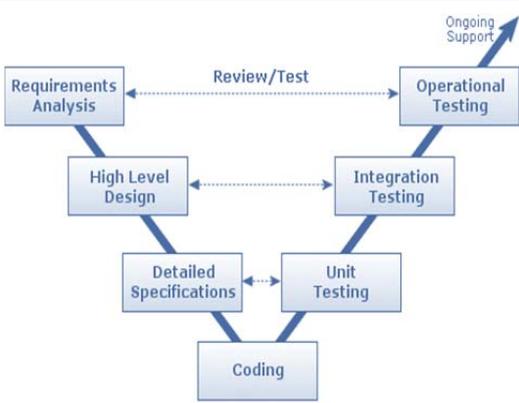
Data/ process integrity ; This principles states that the successful design of any solution requires the incorporation and defined data and process elements must be identified for each and every retirement.

Scalability: This principles states that the “V” concepts has the flexibility to accommodate any IT project irrespective of size, complexity or duration the “V” concept is applicable to a large mainframe development project applying a waterfall approach as it is to a web based development project applying agile techniques.

Cross referencing: This principle states that there must be a direct correlation between every requirement that has been defined with a corresponding and verifiable testing activity and result that substantiates that each and every authorized requirement has been incorporated into the completed application.

Tangible Documentation: This principles states that there must be tangible documentation (electronic/ or hardcopy) created as the project evolves.

Table 7 : V-Model

SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L Applicability Small Medium Large	CRF Cost/Risk/ Flexibility
7	V-Model	 <p>V-model has divided into Some Phases are:</p> <ol style="list-style-type: none"> 1. Requirement Analysis 2. High level Design 3. Detailed Specification 4. Coding 5. Unit Testing 6. Integrated testing 7. Operational testing 	<ul style="list-style-type: none"> - Testing activities like planning testing designing happens well before coding. This saves a lot of time hence higher chance of success over the waterfall model. - Proactive defect tracking that is defects are found at early stage . - Avoids the downward flow of the defects. - Works well for small projects where requirements are easily understood[12]. 	<ul style="list-style-type: none"> - Very rapid and least flexible - Software is developed during the implementation phases, some early prototypes of the software are produced. - If any changes happen in midway, then the test documents along with requirement documents has to be updated. 	Small Project	High Cost High Risk Less Flexibility

8. HISTORY OF CONCURRENT DEVELOPMENT MODEL WITH PRINCIPLES

History of Concurrent Development Model

Sometimes called concurrent Engineering can be repeated semantically as a series of frame work activities, software Engineering and tasks, and their associated states.

The concurrent process model defines a series of events that will trigger transition from a series of events that will tiger transition from state to state for each of the software engineering activities action or tasks.

Phases:

None: none state while initial communication was completed.

Under development: the customer indicates that changes in requirement must be made the modeling activity moves

from the under development state into the awaiting changes state.

Awaiting changes: early in a project the communication activity has completed its first iteration and exists in the awaiting changes state.

Principles are:

The concurrent process model is applicable to all types of software and provides an accurate picture of the current state of a project rather than confining software engineering activities.

Actions and tasks to a sequence of events, it define a network of activities, action, or task on the network exited simultaneous with other activities, action or tasks .

Event generated at one point in the process network trigger transition among the state.

Performance of concurrent development process model

Table 8 : Concurrent Development Model

SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L Applicability Small Medium Large	CRF Cost/Risk
8	Concurrency development Model	<p>Concurrent development Model has divided some phases:</p> <ol style="list-style-type: none"> 1. Under development 2. Awaiting changes 3. Under revision 4. Done 5. Baseline 6. Under revision 7. under review 	<p>-It is can be represented schematically as a series of framework activities.</p> <p>-Concurrency process model that will trigger transition from state to state for each of the software engineering activities and action or task.</p> <p>-The concurrent process model is application to all types of software development and process accurate picture of the current state of a project.</p>	<p>-The SRS must be continually updated to reflect changes.</p> <p>-It requires discipline to avoiding too many new features too late in the project .</p>	Medium Project	<p>Low Cost</p> <p>Low Risk</p> <p>Low Flexibility</p>

9 HISTORY OF AGILE DEVELOPMENT PROCESS MODEL WITH PRINCIPLES

History of Agile Development Process Model

Agile process model development by manifesto in year 2001.

“Agile software development “refers to groups of software development methodologies based on iterative development, where requirements and solutions evolve via collaboration between self-organizing cruces functional teams.

Principles:

1. Our highest priority is to satisfy the customer though early and continuous delivery of valuable software.
2. Welcome changing requirement ever in development. Agile process harness change for the customer’s complete advantage.
3. Delivery working software frequently.

4. Business people and developers must work together daily throughout the project
5. Build project around motivated individual
6. The most effective and effective method of conveying information to and within a development team is face to face communications.
7. Working software in the primary measures of progress.
8. Agile processes promote sustainable development this sponsors, developers and user should be able to maintenance constant place indefinitely.
9. Conation is a function to technical excellence and good design enhance quality.
10. Simplicity the art of maximizing the amount of work not done is essential.
11. The best architecture, requirements and design emerge from self-organizing teams.
12. At regular, integrals the team reflects on how to become more efficient then tunes and adjusts it behavior accordingly.

Table 9: AGILE Development Process Model

Sl.No	M N	AT Architecture	A Advantage	DA Disadvantage	AP S/M/L	CRF
9	AGILE development process Model		<ul style="list-style-type: none"> - Customer satisfaction by rapid continuous delivery of useful software. - Working software is delivery frequently - Face to face conversation is the best form of communication. - Close daily cooperation between business people and developers. - Continuous attention to technical excellence and goal design. 	<ul style="list-style-type: none"> - Some software deliveries, especially the large ones, it is difficult to accesses the effort required at the beginning of the software life cycle . - There is lack of emphasis as necessary designing and documentation - The project can easily get taken off track if the customer representative is not clear what final outcome that they want. 	<p>Large</p> <p>Project</p>	<p>High Cost</p> <p>LowRisk</p> <p>High Flexibility</p>

10. HISTORY OF BUILD AND FIX MODEL WITH PRINCIPLES

History of Build and Fix Model

Build and fix model developed by Boehm1988. In this most simple model of software development, product with is construct with minimal requirements and generally no specifications nor is any attempt at design, and testing most often neglected. This is representation of what is happening in much Software development project.

Phases:

This model includes the following two phases.

Build:

In this phase, the software code is developed and passed on to the nextphase.

Fix:

In this phase, the code developed in the build phase is made error free. Also, in addition to the corrections to the code, the code is modified according to the user's requirements.

Principles:

In this process, developers write code, fix the problems they notice, and repeat. There is no guidance to help developers converge to an appropriate result

Table 10 : Performance of Build and Fix Model

SL NO	M.N Model Name	A.T Architecture	A Advantages	D Disadvantages	APY S/M/L	CRF Cost/Risk/
10	Build and Fix Model	<p> ———> Development - - - -> Maintenance </p> <p>Build and Fix Model Has Three Phases</p> <ol style="list-style-type: none"> 1. Build First Version 2. Modify Until Client is Satisfied 3. Operations 	-Requires less experience to execute or manage other than the ability to program. -Suitable for smaller software. -Requires less project planning.	-No real means is available for assessing the program quality and risks. -Cost of using this process model is high, as it requires rework until user requirement are accomplished. -Informal design of the software as it involves unplanned procedure. -Maintenance of this model is problematic [13].	Small Project	High Cost High Risk High Flexibility

CONCLUSION:

After analysis of ten software process models, through the various factors, it has been found that the original architecture, phases and principles, performance model. Simulation can be used to identify process flaws deficiencies and bottle necks, to estimate the impact of potential changes to the process and to compare alternative process models without putting new process into practice. This paper is very useful to the developers.

REFERENCES:

- [1] Utest : <http://blog.utest.com/its-a-software-crisis/2011/04>
- [2] Model-Based system Architecture-page184 <http://books.google.co.in/books?isbn=1118893646>
- [3]. IBM acquires rations http://en.wikipedia.org/wiki/rational-unified_process
- [4]. B.W. Bohme “ Anchoring the software process”, IEEE,IEEE software, Vol- 13,Issues 4, july1996, PP.73-83. www.testingexcellence.com/incremental-model/.
- [5] The Spiral Model Advantages and disadvantages are available at: www.tutorialpoint.com/sdlc/sdlc-spiral-model.htm
- [6] R.S Preusman, “Software engineering, A practitioner’s approach” 5th edition. Newyork: McGraw-Hill, 2001, pp 34.
- [7] E.I. Mary, B.A.zimmer “The Evolutionary Development model for Software”, Hewlett-Packard journal, Article 4, August 1996, PP, 1-8.
- [8] asked.mark(2003-11-25)."rup in brief" IBM retrie
- [9] Testingfreak.com/rad-model-software-testing-advantage-disadvantage
- [10] v-model advantage disadvantage available at: <http://en.wikipedia.org/wiki/v-model>
- [11]. the"vmodel by comeron waston-project insight [www.projectinsight.net/./the-quot-v-quot-mdel by -cameron-waston](http://www.projectinsight.net/./the-quot-v-quot-mdel-by-/-cameron-waston)
- [12] V-Model advantages and disadvantages: Istqbexamcertification.com/what-is-v-model-advantages-disadvantages
- [13]. Software engineering pages: <http://books.google.com.in/books?isbn=82122423779sangeethasabharwal-2008>

AUTHORS



S.Thulasi Krishna received the B.Tech. Degree in Computer Science and Engineering from Jawaharlal Nehru University, Hyderabad, India in 2005, M.E from Sathyabama University Chennai, and currently pursuing Ph.D in Rayalaseema University, Kurnool. Presently working as an Associate Professor in Kuppam Engineering College. He is also a member of International Association of Engineering.



Dr. S. Sreekanth has obtained Ph.D. Degree from S.V.University, Tirupathi. Presently he is working as a Professor in the Department of Computer Science and Engineering in Sreenivasa Institute of Technology and Management Studies, Chittoor, Andhra Pradesh, with the experience of 20 years. He has published 15 research papers both in national and international journals of computer science.