ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
Available online at www.ijarbest.com

International Journal of Advanced Research in Biology Ecology Science and Technology (IJARBEST)
Vol. I, Special Issue IV, December 2015 in association with BPC COLLEGE PIRAVOM
National Conference on Recent Trends in Data Mining (NCRTID-2015) - 10th & 11th December 2015

Comparative Study of Various Process Model in Software Development Life Cycle

Ms.Manju Parvathy J
MSc Computer Science
BPC College, Piravom, Kerala, India

Ms.Athira K B
MSc Computer Science
BPC College, Piravom, Kerala, India

Ms. Saumya John
Asst Professor
Dept of Computer Science
BPC College, Piravom, Kerala, India

Abstract— **In the present scenario all software systems are imperfect because they cannot be built with mathematical or physical certainty. Hence in this research paper the comparison of various software development models has been carried out. According to SDLC each and every model have the advantages and disadvantages. So in this research we have to calculate the performance of each model on behalf of some important features. In this paper we have done the comparative study of the following software models namely Waterfall, Prototype, RAD (Rapid Application Development) Incremental, Spiral, Build and Fix and V-shaped.**

Keywords— ***SDLC,Introduction of Process Model, Comparative Study of Process Model with Different parameter, factors affecting to Chose Process Model***

I. INTRODUCTION

Computers are being used in many areas like in banking, education, medicine etc. These areas require specialized software according to the applications they need. Hardware alone is not sufficient to do some useful work.Software and hardware are complementary to each other.Software engineering**[1]**is an engineering discipline whose aim is development of quality product, a product which is reliable, within estimated budget and within a given time framework.

A software development process,also known as a software development life cycle (SDLC),is a structure imposed on the development of 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.

Various processes and methodologies have been developed over the last few decades to improve software quality , with varying degrees of success. Software lifecycle model provides a method for developing a software product. A proper software life cycle model can help an organization not only in building a software product but it also serves as a basis for planning, organizing, staffing, coordinating and directing various other software development activities. Software Engineering processes are composed of many activities, notably the following :

- Requirement 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.In IEEE standard Glossary of Software Engineering Terminology, the software Life Cycle is: “The period of time that starts when a software product is conceived and ends when the product is no longer available for use”.

II. SOFTWARE DEVELOPMENT LIFE CYCLE

A systems development life cycle is composed of a number of clearly defined and distinct work phases which are used by systems engineers and systems developers to plan for, design, build, test, and deliver software systems. Computer systems are complex and often (especially with the recent rise of service-oriented architecture) link multiple traditional systems potentially supplied by different software vendors. To manage this level of complexity, a number of SDLC models or methodologies have been created, such as waterfall, prototype, RAD, incremental, spiral, build and fix, V-shaped.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
Available online at www.ijarbest.com

International Journal of Advanced Research in Biology Ecology Science and Technology (IJARBEST)
Vol. I, Special Issue IV, December 2015 in association with BPC COLLEGE PIRAVOM
National Conference on Recent Trends in Data Mining (NCRTID-2015) - 10th & 11th December 2015

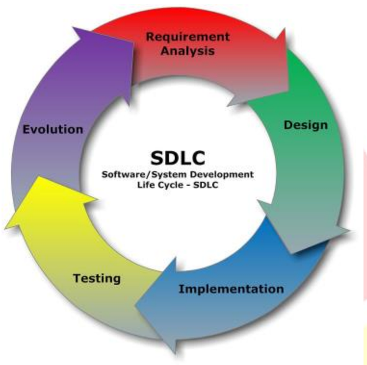

Fig.1. Software Development Life Cycle

III. Software Process Models

Software Process model describes the phases of software cycle and the order in which those phases are executed. Each Phase produces deliverables required by next phase in life cycle.

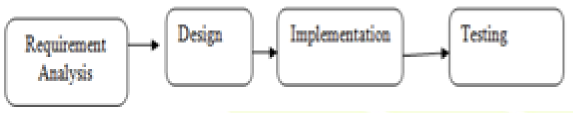

Fig.2. Phases of Software process models.

A. Requirement Analysis

This is based on the requirement of project manager in the business environment.This can be analysed for their validity and possibility of incorporating the requirement in the system to be process. In this phase documentation is done for the next phase.

B. Design

In this phase design of software is prepared from the requirement phase.Design phase helps in defining the overall architecture of system.System design act as input for the next phase.

C. Implementation/Coding

After designing the work can be divided into different module and coding is started.This phase is the longest phase in process model.

D. Testing

After coding phase testing is done against the requirement to make sure that the product is actually solving needs. After successful testing the product is delivered / deployed to the customer for their use. Once when the customers starts using the developed system then the actual problems comes up and needs to be solved from time to time. This process where the care is taken for the developed product is known as maintenance.

IV. General Software Process Models

1) Waterfall model
2) Prototype model
3) Rapid application development model (RAD)
4) Incremental model
5) Spiral model
6) Build and fix model
7) V-shaped model

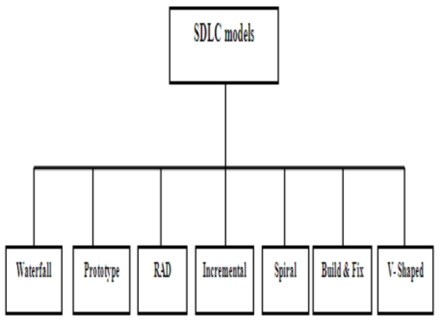

Fig.3. General Software process models.

V. Research Elaborations

A. The Waterfall Model

The waterfall model is the classical model of software engineering. This model is one of the oldest models and is widely used in government projects and in many major companies. This model is named “Waterfall” because its diagrammatic representation looks like a cascade (flow) of Waterfall. As this model emphasizes planning in early stages, it ensures design flaws before they develop. In addition, its intensive document and planning make it work well for projects in which quality control is a major model:

1. System requirements: Establishes the components for building the system, including the hardware requirements, software tools, and other necessary components. Examples include decisions on hardware, such as plug-in boards (number of channels, acquisition speed, and so on), and decisions external pieces of software, such as databases or libraries.
2. Software requirements: Establishes the expectations for software functionality and identifies which system requirements the software affects. Requirements analysis includes determining interaction needed with other

applications and databases, performance requirements, user interface requirements, and so on.

3. Architectural design: Determines the software framework of a system to meet the specific requirements. This design defines the major components and the interaction of those components, but it does not define the structure of each component. The external interfaces and tools used in the project can be determined by the designer.
4. Detailed design: Examines the software components defined in the architectural design stage and produces a specification for how each component is implemented.
5. Coding: Implements the detailed design specification.
6. Testing: Determines whether the software meets the specified requirements and finds any errors present in the code.
7. Maintenance: Addresses problems and enhancement requests after the software releases.

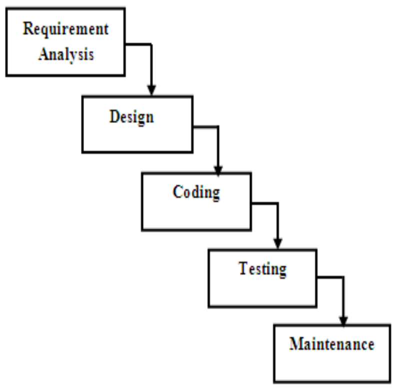

Fig.4. Waterfall models.

Advantages:

- Easy to understand and implement.
- Widely used and known (in theory!).
- Reinforces good habits: define-before- design, design-before-code.
- Identifies deliverables and milestones.
- Document driven.
- Works well on mature products and weak teams.

Disadvantages:

- Idealized, doesn't match reality well.
- Doesn't reflect iterative nature of exploratory early in project.
- Software is delivered late in project, delays discovery
- Difficult and expensive to make changes to documents, "swimming upstream".

B.PURE WATERFALL

The pure waterfall lifecycle consists of several non-overlapping stages, as shown in the following figure. The model begins with establishing system requirements and software requirements and continues with architectural design, detailed design, coding, testing, and maintenance. The waterfall model serves as a baseline for many other lifecycle models.This is the classical system development model. It consists of discontinuous phases:

- Concept.
- Requirements.
- Architectural design.
- Detailed design.
- Coding and development.
- Testing and implementation.

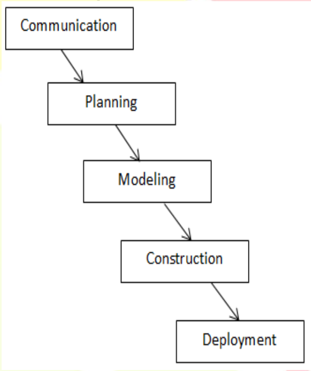

Fig.5. Pure Waterfall models.

C. PROTOTYPE MODEL

The basic idea here is that instead of freezing the requirements before a design or coding can proceed, a throwaway prototype is built to understand the requirements. This prototype is developed based on the currently known requirements. i.e., incomplete versions of the software program being developed.

Not a standalone, complete development methodology, but rather an approach to handling selected parts of a larger, more traditional development methodology.

Attempts to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process. User is involved throughout the development process, which increases the likelihood of user acceptance of the final implementation. Small-scale mock-ups of the system are developed following an iterative modification process until the Prototype evolves to meet the users requirement. 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

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
Available online at www.ijarbest.com

International Journal of Advanced Research in Biology Ecology Science and Technology (IJARBEST)
Vol. I, Special Issue IV, December 2015 in association with BPC COLLEGE PIRAVOM
National Conference on Recent Trends in Data Mining (NCRTID-2015) - 10th & 11th December 2015

system. A basic understanding of the fundamental business problem is necessary to avoid solving the wrong problem.

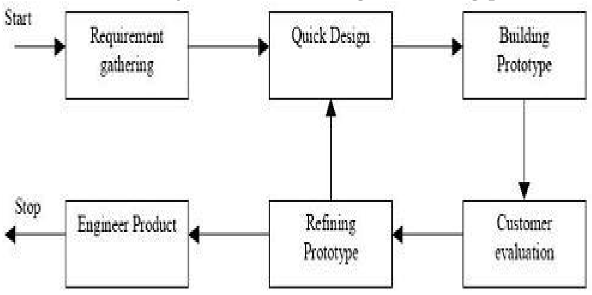

Fig.6. Prototype model.

Advantages

- Early visibility of the prototype gives users an idea of what the final system looks like Encourages active participation among users and producer .
- The customer understand the software development process easily.
- Customer gives the feedback for the next increment of the prototype.
- Cost effective (Development costs reduced).
- Increases system development speed.

Disadvantages

- Possibility of causing systems to be left unfinished.
- The customer sees what appears to be a working version of the software, unaware that the prototype is incomplete.
- Often lack flexibility.
- Not suitable for large applications.
- Project management difficulties.

D. RAD MODEL

RAD, the Rapid Application Development model is a high speed adaption of waterfall model. This model can be implemented if a developer knows the requirements of customer in advance and here the development cycle is extremely small. In this model the components or functions are developed in parallel as if they were mini projects. The development are time boxed, delivered and then bring together into a working prototype. This can quickly give the customer something to see and use and to provide feedback regarding the delivery and their requirements.

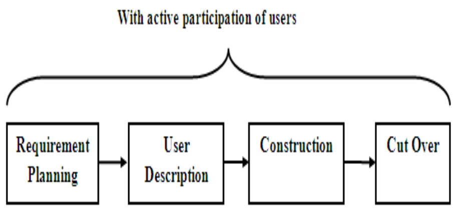

Fig.7. RAD model.

Advantages

- Deliverables are easier to transfer as high level abstractions , scripts and intermediate codes are used.
- Provides greater flexibility ad redesign is done according to the developer.
- Results in reduction of manual coding due to code generators and code reuse.
- Encourages user involvement.
- Possibility of lesser defects due to prototyping in nature.

Disadvantages

- Useful for only larger projects.
- RAD projects fail if there is no commitment by the developers or the users to get software completed on time.
- Not appropriate when technical risks are high. This occurs when the new application utilizes new technology or when new software requires a high degree of interoperability with existing system.
- As the interests of users and developers can diverge from single iteration to next, requirements may not converge in RAD model.

E. INCREMENTAL MODEL

It Combines elements of the waterfall model applied in an iterative fashion each linear sequence produces deliverable "increments" of the software. The first increment is often a core product. The core product is used by the customer. If a customer requires changes in its product, then incremental model**[2]** accommodate changes as required by the customer.The previous models discussed earlier do not take into consideration changes in product. This model is iterative in nature. A reusable product is released at the end of each cycle, with each release providing additional functionality. After each release customer can do some useful work and thereby he can accommodate changes in the product.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
Available online at www.ijarbest.com

International Journal of Advanced Research in Biology Ecology Science and Technology (IJARBEST)
Vol. I, Special Issue IV, December 2015 in association with BPC COLLEGE PIRAVOM
National Conference on Recent Trends in Data Mining (NCRTID-2015) - 10th & 11th December 2015

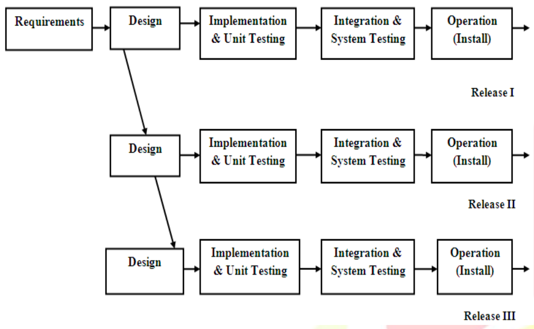
Requirements
Design
Implementation & Unit Testing
Integration & System Testing
Operation (Install)
Release I
Design
Implementation & Unit Testing
Integration & System Testing
Operation (Install)
Release II
Design
Implementation & Unit Testing
Integration & System Testing
Operation (Install)
Release III

Fig.8. Incremental model.

Advantages

- Divides project into smaller parts
- Creates working model early and provides valuable feedback
- More flexible- less costly to change scope and requirements.
- Lowers initial delivery cost.
- Easier to manage risk because risky pieces are identified and handled during its iteration .
- Very useful when more staffing is unavailable.

Disadvantages

- Each phase of an iteration is rigid and do not overlap each other.
- User community needs to be actively involved in the project. This demands on time of the staff and add project delay
- Communication and coordination skills take a center stage
- Informal requests for improvement for each phase may lead to confusion
- It may lead to —scope creep

F. SPIRAL MODEL

The spiral model is similar to the incremental model, with more emphases placed on risk analysis. The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation. A software project repeatedly passes through these phases in iterations (called Spirals in this model).

The spiral model combines advantages of top-down and bottom up concepts. It is a meta-model, a model that can be used by other models.

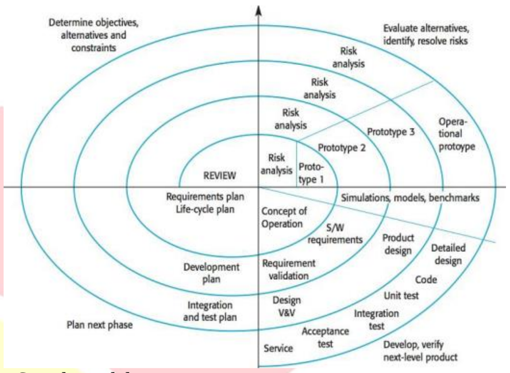

Fig.9. Spiral model.

A spiral model is divided into a number of framework activities, also called *task regions*. Typically, there are between three and six task regions. six task regions:

- **Customer communication**—tasks required to establish effective communication between developer and customer.
- **Planning**—tasks required to define resources, timelines, and other project related information
- **Risk analysis**—tasks required to assess both technical and management risks.
- **Engineering**—tasks required to build one or more representations of the application.
- **Construction and release**—tasks required to construct, test, install, and provide user support (e.g., documentation and training).[3]

Advantages

- Was designed to include the best features form Waterfall and Prototyping Model
- Good for large and mission-critical projects
- Introduces a new component – risk assessment
- Similar to prototyping model, an initial version of system is developed and modified based on input from customer

Disadvantages

- Can be a costly model to use.
- Risk analysis requires highly specific expertise
- Project’s success id highly dependent on risk analysis phase
- Doesn’t work well for smaller projects .

G. BUILD AND FIX MODEL

In this model a software product is built without any specification and without applying any kind of design.

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
Available online at www.ijarbest.com

International Journal of Advanced Research in Biology Ecology Science and Technology (IJARBEST)
Vol. I, Special Issue IV, December 2015 in association with BPC COLLEGE PIRAVOM
National Conference on Recent Trends in Data Mining (NCRTID-2015) - 10th & 11th December 2015

Developer in this model adopts an adhoc**[4]** approach which is not well defined.
This model includes the two phases.

- **Build:** In this phase, the software code is developed and passed on to the next phase for the launch.
- **Fix:** In this phase, the code which has been developed in 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 and recommendations.

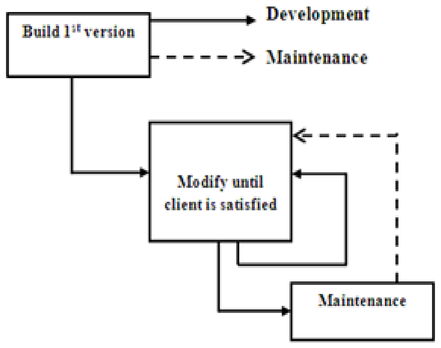

Fig.10. Build and Fix model.

Advantages

- Requires less experience to execute or manage other than the ability to program.
- Suitable for smaller software.
- Requires less project planning.

Disadvantages

- No real means is available of assessing the progress, quality and risks.
- Cost of using this process model is high as it requires rework until users requirements are accomplished.
- Informal design of the software as it involves unplanned procedure.
- Maintenance of these models is problematic.

H. V-shaped model

V-model means Verification and Validation model. This model can be considered as extension of waterfall model. In waterfall model we move in a linear way while in V model process steps are bent upwards the coding phases to form typical V-Shape. The relationship between each phase of the development process (verification) and its associated phase of testing (validation) is shown in V model. In V model emphasis is more on testing as testing is one of the important
part of Software development.

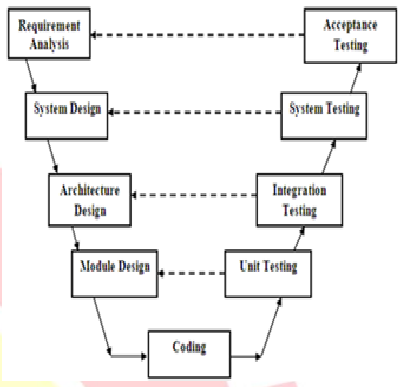

Fig.11. V-shaped model.

Advantages

- Simple and easy to use.
- Testing activities like planning test designing happens well before coding. This saves a lot of time. Hence higher chance of success over the Waterfall model.
- Avoids the downward flow of defects.
- Works well for small projects where requirements are easily understood.

Disadvantages

- Very rigid and least flexible.
- Software is developed during the implementation phase, so no early prototypes of the software are produced.
- If any changes happen in midway, then the test document along with requirements documents has to be updated.
- This model does not provide a clear path for problems found during testing phases.

I. AGILE MODEL

Agile development is claimed to be a creative and responsive effort to address users' needs focused on the requirement to deliver relevant working business applications quicker and cheaper. The application is typically delivered in incremental (or evolutionary or iterative) fashion. The agile development approaches are typically concerned with maintaining user involvement through the application of design teams and special workshops. The delivered increments tend to be small and limited to short delivery periods to ensure rapid completion.

In Agile SDLC, requirement specification can change frequently because they will understand by the customer and software developerAs the meeting of customer with software developer, this will lead to higher chance of success [5]. Developer can control over the cost, if he acquire only needed

ISSN 2395-695X (Print)
ISSN 2395-695X (Online)
Available online at www.ijarbest.com

International Journal of Advanced Research in Biology Ecology Science and Technology (IJARBEST)
Vol. I, Special Issue IV, December 2015 in association with BPC COLLEGE PIRAVOM
National Conference on Recent Trends in Data Mining (NCRTID-2015) - 10th & 11th December 2015

requirement then cost can be covered by developer. This model is very difficult to implement but with low risk involvements.

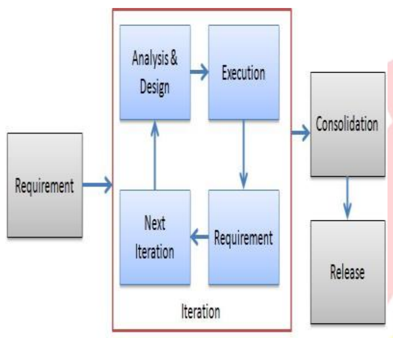

Fig.12. Agile model.

Advantages

- Customer satisfaction by rapid, continuous delivery of useful software.
- People and interactions are emphasized rather than process and tools. Customer, developers and testers constantly interact with each other.
- Working software id delivered frequently(weeks rather than months).
- Face-to-face conversation is the best form of communication.
- Continuous attention to technical excellence and good design.
- Regular adaptations to changing circumstances.
- Even late changes in requirements are welcomed.

Disadvantages

- In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of SDLC.
- There is lack of emphasis on 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.
- Only senior programmers are capable of taking the kind of decision required during the development process. Hence it has not place for newbie programmers, unless combined with experienced resources.

TABLE 1 Comparative Analysis of Models

Feature /Model	Water Fall	Prototype	RAD	Incremental	Spiral	Build and Fix	V-Shaped	Agile
Requirement Specifications	Beginning	Frequently changed	Time box released	Beginning	Beginning	Frequently changed	Beginning	Frequently changed
Cost	Low	High	Low	Low	Expensive	High	Expensive	Very High
Resource Control	Yes	No	Yes	Yes	Yes	No	Yes	No
Simplicity	Simple	Simple	Simple	Intermediate	intermediate	simple	Intermediate	Complex
Risk Analysis	Only at Beginning	No	Low	No Risk Analysis	Yes	No	Yes	Yes
User Involvement	Only at Beginning	High	Only at Beginning	Intermediate	High	No	At the beginning	High
Flexibility	Rigid	Highly Flexible	High	Less flexible	Flexible	Flexible	Little flexible	Highly Flexible
Reusability	Limited	Week	To some extend	Yes	Yes	Limited	To some extend	Use Case

								reuse

VI. Conclusion

This paper discussed what is software process model and various process models, also compare them with different parameter and highlight the factors for choosing them. However the existing model still can be improve and modified based on less cost, time and high efficient.

Each and every model has its own advantages and disadvantages for the development of systems, so each model tries to eliminate the disadvantages of the previous model.

References

[1] Prateek Sharma, Dhananjaya Singh, 'Comparative Study of Various SDLC Models on Different Parameters 'International Journal of Engineering Research Volume No.4, Issue No.4, pp : 188-191

[2] Craig Layman and Victor Basili, "Iterative and Incremental Development: A Brief History", IEEE Computer, 2003.

[3] Roger Pressman, Software Engineering: A Practitioner's Approach, Sixth Edition, McGraw-Hill Publication

[4] A. M. Davis, H. Bersoff, E. R. Comer, "A Strategy for Comparing Alternative Software Development Life Cycle Models", Journal IEEE Transactions on Software Engineering, Vol. 14, Issue 10, 1988.

[5] K. Schwaberand M. Beedle(2001)," Agile Software Development with Scrum", Upper Saddle River, NJ, Prentice –Hall, 1st Edition, Oct 2001

[6] Ashwini Mujumdar, Gayatri Masiwal, P. M. Chawan / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com Vol. 2, Issue 3, May-Jun 2012, pp.2015-2021

[7] Asmita,kamlesh,usha.Review on comparative study of software process models.International Journal of Science, Technology & Management Volume No 04, Special Issue No. 01, March 2015

[8] Prateek Sharma1, Dhananjaya Singh. Comparative Study of Various SDLC Models on Different Parameters International Journal of Engineering Research ISSN:2319-6890)(online),2347-5013(print)Volume No.4, Issue No.4, pp : 188-191.

[9] Mr. Preyash Dholakia1, Mr. Dishek Mankad2 The Comparative Research on Various Software Development Process Model International Journal of Scientific and Research Publications, Volume 3, Issue 3, March 2013 International Journal of Computer Applications (0975 – 8887) Volume 82 – No 18, November 2013

[10] Ms. Shikha maheshwari1 Prof.Dinesh Ch. Jain2 . A Comparative Analysis of Different types of Models in Software Development Life Cycle Volume 2, Issue 5, May 2012 International Journal of Advanced Research in Computer Science and Software Engineering

[11] 1 Dr. B V Ramana Murthy, 2 Prof. Vuppu Padmakar 3 Ms. A. Vasavi Comparative Analysis of Software Development Process Models Volume 4, Issue 6, June 2014 ISSN: 2277 128X International Journal of Advanced Research in Computer Science and Software Engineering.

[12] Ali Makhmali, Hajar Mat Jani," Comparative Study On Encryption Algorithms And Proposing A Data Management Structure" international journal of scientific & technology research

[13] Behrouz Forouzan, "Cryptography and network security"

[14] Gurpreet Singh , Supriya "A Study of Encryption algorithm(RSA, DES , 3DES and AES) for Information Security" ,International Journal of Computer Applications

[15] AL .Jeeva, Dr .V. Palanisamy, K. Kanagaram ," Comparitive analysis of performance efficiency and security measures of some Encryption algorithms" International Journal of Engineering Research and Applications

[16] Mohit Marwaha, Rajeev Bedi, Amritpal Singh, Tejinder Singh, " Comparative analysis of cryptographic algorithms" , International Journal of Advanced Engineering Technology

[17] Hamdan.O.Alanazi, B.B.Zaidan, A.A.Zaidan, Hamid A.Jalab, M.Shabbir and Y. Al-Nabhani, "New Comparative Study Between DES, 3DES and AES within Nine Factors ",Journal of computing.