A Comparative Study of Universally Accepted SDLC Models for Software Development

Madhup Kumar
M.Tech Research Scholar, Jharkhand Rai University, Ranchi, Jharkhand, India

ABSTRACT

A software life cycle is the series of identifiable stages that a software product undergoes during its life time. An s/w life cycle model is a descriptive and diagrammatic representation of s/w life cycle. A life cycle model map the different activities performed on a s/w product from its inspection to retirement. Business organization follow steps-Business process, manufacturing industries-manufacturing process same as for s/w development use s/w process model. The first life cycle of any s/w product is generally feasibility study, RAS, Design, Coding, Testing and Maintenance. A (software/system) life cycle model is a description of the sequence of activities carried out in an SE project, and the relative order of these activities.

Keywords: Software Engineering, SDLC, Comparative Life Cycle Model

I. INTRODUCTION

Advent of computers has voraciously changed our lives. Earlier its usage was limited for personal use or for particular task but now computer is used widely. A software engineering discipline gives us method, process, tools etc for developing computer software.[1] Software engineering is tool for software developers for developing error free software in systematic and in discipline manner as per user requirements. Delivery of the software should be in time and in budget. In past year traditional approach were used for developing software .But that method is not fit for these days it had several limitation [2].Now these days there are several process model by which we can develop our software that is called SDLC (software development life cycle model) each model has some phases and a software undergoes each phase of SDLC model[3]. Software developers are free to adopt any life cycle model it's depend on developer team and nature of the developing software[4]. Every SDLC model has some advantages and some limitations also[5]. In this paper we compare each SDLC model its advantages, limitation and when to use which SDLC model. SDLC model has some phase like FS, RS, Designing, Coding, Testing, Maintenance etc. Objective of this research paper is to find out advantages, drawbacks of each SDLC model and when to use which model.

The remainder of this paper is organized as follows: Sect. II discusses the study of several SDLC models with their usage, advantages and disadvantages. Section III compares each of the methods with reference to their contemporary models. Section IV concludes the paper and discusses future scope.

II. STUDY OF SEVERAL SDLC MODELS

2.1 Classical Waterfall Model

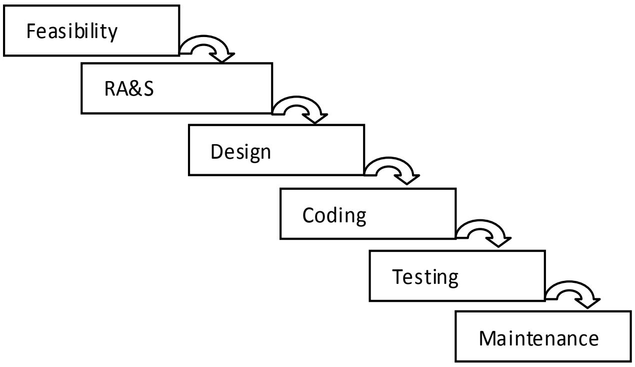


Figure 1. Steps involved in the Classical waterfall Model.

2.1.1 When to use the waterfall model: [9, 10]

- ✓ When the requirements are very well understood and fixed.
- ✓ When Product definition is constant.
- ✓ When Technology is understood.
- ✓ When there are no confusing requirements.
- ✓ When sufficient resources with required expertise are available without restraint.
- ✓ When the project is short

2.1.2 Advantages of waterfall model:

- ✓ User friendly, easy to use and understand
- ✓ It is easy to manage due to the inflexibility of the model
- ✓ Phases do not go beyond.
- ✓ These models manly used for smaller projects where requirements are very well understand.

2.1.3 Disadvantages of waterfall model:

- ✓ Iteration problem (it is very difficult to go back from one phase to another phase.)
- ✓ Late delivery of the software.
- ✓ High amounts of risk and ambiguity.
- ✓ Not feasible for complex and object-oriented projects.
- ✓ Not fit for long and ongoing projects.
- ✓ Not good where high risk chances occur and the requirements are moderate.

2.2 Waterfall Iterative Model [7]

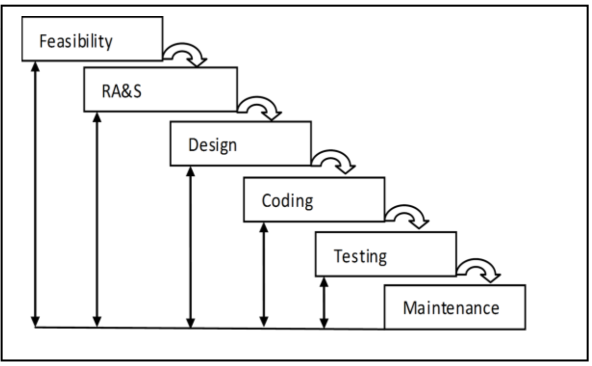


Figure 2. Steps involved in the Iterative waterfall Model

2.2.1 When to use the Iterative model:

All the phases of this model are same as classical waterfall model. Condition for using this model is also same as classical waterfall model.

2.2.2 Advantages of Iterative waterfall model:

Advantages of Iterative waterfall model is same as classical waterfall model including one or more feature added in this model i.e. Iteration is possible

In this model developer can go back form one phase to another i.e. if some problem occur in one phase iteration is possible (go back to one phase to another phase).

Iterative waterfall model overcome the one of the iteration problem of waterfall model. This model is same as classical waterfall model all process of developing project are same as waterfall model.

2.2.3 Disadvantages of Iterative Model of SDLC

- ✓ Iterations possible but may cause confusion as the project proceeds.
- ✓ In this model we freeze software and hardware. But not advisable especially in long-term projects.
- ✓ Requirement gathering is a tough task.
- ✓ Change in any previous stage can cause big problem (Dependency).
- ✓ Iteration can be a costly.

2.3 Prototype Model [5]

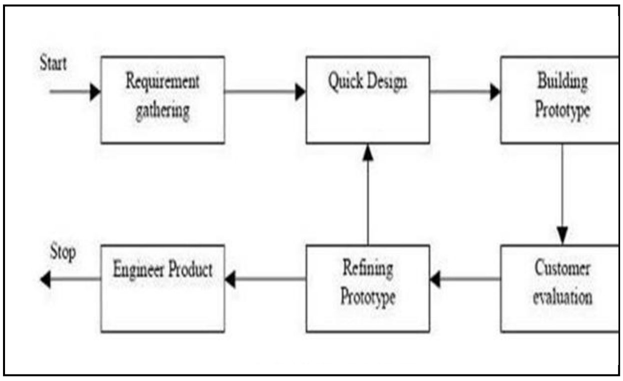


Figure 3. Steps involved in design of Prototype Model

2.3.1 When to use the prototype model:

- ✓ When the end users have lot of interaction with system.
- ✓ For developing online systems, web interfaces, like system are appropriate for this model. A little bit user training is required for this system.
- ✓ Feedback is necessary from user/customer for development of this model the user can use the system before final delivery of the system.

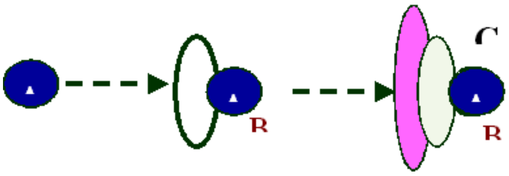
2.3.2 Advantages of Prototype model:

- ✓ User's involvement in the development cycle.
- ✓ Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed.
- ✓ Errors can be detected much earlier.
- ✓ Quicker user feedback is available leading to better solutions.
- ✓ Missing functionality can be identified easily
- ✓ Confusing or difficult functions can be identified Requirements validation, Quick implementation of incomplete, but functional applications.

2.3.3 Disadvantages of Prototype SDLC model:

- ✓ Leads to implementing and then repairing way of building systems.
- ✓ Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.
- ✓ Incomplete application may cause application not to be used as the full system was designed Incomplete or inadequate problem analysis.

2.4 Incremental Model



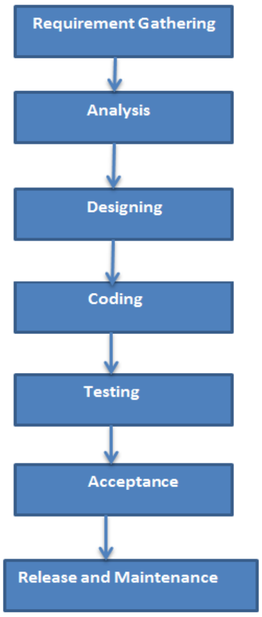


Figure 4. Design flow of the Incremental Model

2.4.1 When to use the Incremental model: [8]

- ✓ This model can be used when the requirements of the complete system are clearly defined and understood.
- ✓ Major requirements must be defined; however, some details can evolve with time.
- ✓ When there is an immediate need to get a product to the market early.
- ✓ A new technology is being used
- ✓ Resources with needed skill set are not available
- ✓ There are some high risk features and goals.

2.4.2 Advantages of Incremental model:[11]

- ✓ Generates working software quickly and early during the software life cycle.
- ✓ This model is more flexible – less costly to change scope and requirements.
- ✓ It is easier to test and debug during a smaller iteration.
- ✓ In this model customer can respond to each built.
- ✓ Lowers initial delivery cost.
- ✓ Easier to manage risk because risky pieces are identified and handled during its iteration.

2.4.3 Disadvantages of Incremental model:

- ✓ Needs good planning and design.
- ✓ Needs a clear and complete definition of the whole system before it can be broken down and built incrementally.
- ✓ Total cost is higher than the waterfall model.

2.5 Spiral Model

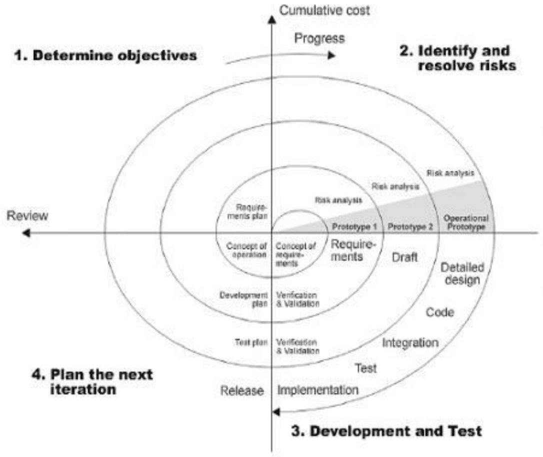


Figure 5. The phases involved in the Spiral Model

2.5.1 When to use Spiral model:

- ✓ When costs and risk evaluation is important
- ✓ For medium to high-risk projects
- ✓ Long-term project commitment unwise because of potential changes to economic priorities
- ✓ Users are unsure of their needs
- ✓ Requirements are complex
- ✓ New product line
- ✓ Significant changes are expected (research and exploration)

2.5.2 Advantages of Spiral model [5]

- ✓ High amount of risk analysis hence, avoidance of Risk is enhanced.
- ✓ Good for large and mission-critical projects.
- ✓ Strong approval and documentation control.
- ✓ Additional Functionality can be added later.
- ✓ Software is produced early in the software life cycle.

2.5.3 Disadvantages of Spiral model:

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

2.6 The V-model

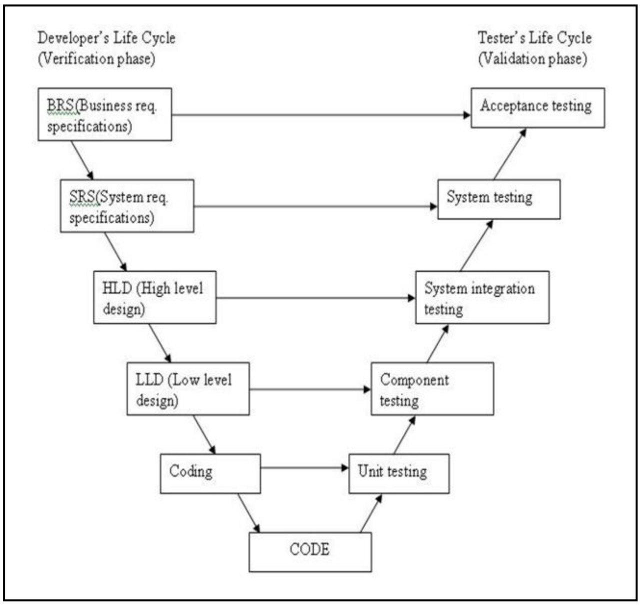

Developer's Life Cycle (Verification phase)
Tester's Life Cycle (Validation phase)
BRS(Business req. specifications)
Acceptance testing
SRS(System req. specifications)
System testing
HLD (High level design)
System integration testing
LLD (Low level design)
Component testing
Coding
Unit testing
CODE

Figure 6. The step-by step phases involved in V model

2.6.1 When to use the V-model: [10]

- ✓ The V-shaped model should be used for small to medium sized projects where requirements are clearly defined and fixed.
- ✓ The V-Shaped model should be chosen when ample technical resources are available with needed technical expertise.
- ✓ High confidence of customer is required for choosing the V-Shaped model approach. Since, no prototypes are produced, there is a very high risk involved in meeting customer expectations.

2.6.2 Advantages of V-model:

- ✓ 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 prevails.

- 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.

2.6.3 Disadvantages of V-model:

- 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 documents along with requirement documents has to be updated.

2.7 RAD model

2.7.1 When to use RAD model:[11]

- RAD should be used when there is a need to create a system that can be modularized in 2-3 months of time.
- It should be used if there's high availability of designers for modeling and the budget is high enough to afford their cost along with the cost of automated code generating tools.
- RAD SDLC model should be chosen only if resources with high business knowledge are available and there is a need to produce the system in a short span of time (2-3 months).

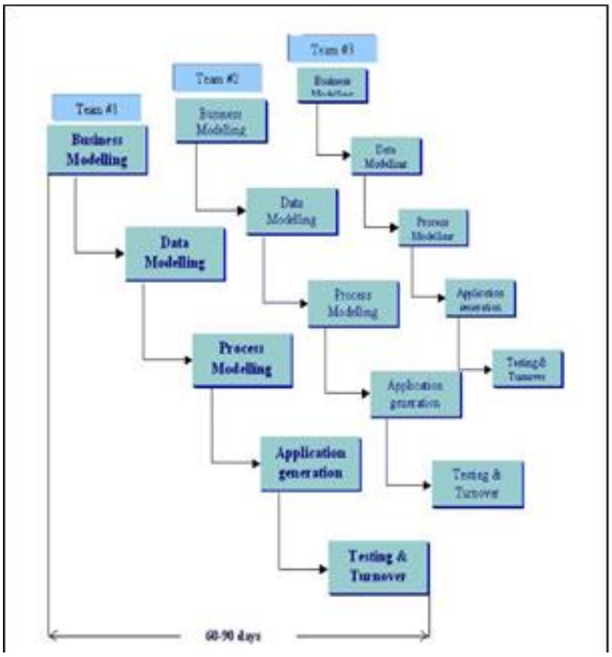


Figure 7. The parallel phases of the RAD Model

2.7.2 Advantages of the RAD model:

- Reduced development time.
- Increases reusability of components
- Quick initial reviews occur
- Encourages customer feedback
- Integration from very beginning solves a lot of integration issue.

2.7.3 Disadvantages of RAD model:

- Depends on strong team and individual performances for identifying business requirements.
- Only system that can be modularized can be built using RAD.
- Requires highly skilled developers/designers.
- High dependency on modeling skills.
- Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.

2.8 Agile model

2.8.1 When to use Agile model:[11]

Agile technology is used when new changes are needed to be implemented. The freedom agile gives to change is very important. New changes can be implemented at very little cost because of the frequency of new increments that are produced.

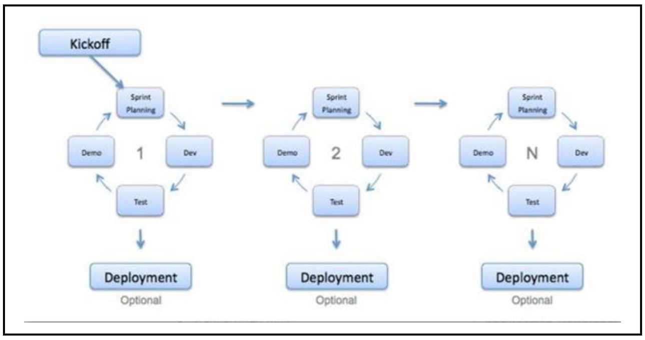


Figure 8 .The phases of the Agile Model

To implement a new feature, the developers need to lose only the work of a few days, or even only hours, to roll back and implement it.

Unlike the waterfall model, in agile model very limited planning is required to get started with the project. Agile assumes that the end users' needs are ever changing in a dynamic business and IT world. Changes can be discussed and features can be newly affected or removed based on feedback. This effectively gives the customer the finished system they want or need.

Both system developers and stakeholders alike, find they also get more freedom of time and options than if the software was developed in a more rigid sequential way. Having options gives them the ability to leave important decisions until more or better data or even entire hosting programs are available; meaning the project can continue to move forward without fear of reaching a sudden standstill.

2.8.2 Advantages of Agile model:

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

2.8.3 Disadvantages of Agile model:[6]

- ✓ In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
- ✓ 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 decisions required during the development process. Hence it has no place for newbie programmers, unless combined with experienced resources.

III. COMPARATIVE STUDY OF SEVERAL SDLC MODELS

3.1 AGILE vs. SPIRAL

TABLE 1. COMPARISON OF AGILE AND SPIRAL MODEL

AGILE MODEL	SPIRAL MODEL
Sustainability and maintenance is risky.	Improved risk management.

Documents and rules are easily employ.	Large number of transitional stages requires extreme documentation.
No planning required.	Required.
Easy to use.	More complex.
Partial working model delivered.	At end of the phases.
Good for small type software.	Not good for small and low risk software.
Depends customer relations.	Does not depend.
All iteration is a split model.	Not a split model
Mixture of iterative and incremental model.	Mixture of iterative and waterfall.

3.2 AGILE vs. ITERATIVE MODEL

TABLE 2. COMPARISON OF AGILE AND ITERATIVE

AGILE MODEL	ITERATIVE MODEL
It contains more risk of sustainability, maintainability and extensibility.	Risks are identified and resolved during iteration.
Agile method breaks the product in small incremental builds. These builds are provided in iteration.	An iterative life cycle model, project is divided into different builds which are proceed iteratively.
It is incremental and iterative model. So it can be used to build more complex project.	It is iterative, all builds are developed, tested Iteratively.
The main focus is on adaptability and customer satisfaction with rapid development of project.	The main focus is on producing new version of software at the end of each iteration to satisfy customer needs.
It can be used to develop more complex projects.	It cannot be used for more complex projects as iterative practice will increase complexity, time requirement.
Quick in development of application	Comparatively slower development.
Various modules of the application can be developed simultaneously.	Only 1 module is developed at a time.
Constant monitoring is required.	Monitoring is required after every iteration.
More flexible with customer requirements.	Less flexible.
Evaluation is done more frequently resulting in better error detection.	Evaluation is after every iteration hence errors if any are detected after a delay.
Progress cannot be clearly measured.	Progress can be clearly measured.
Least resources are required.	More resources required.

Agile development considers number of different methodologies with specific steps to produce a software project example being Extreme Programming. The Spiral Model on the other hand is an example of iterative development. It includes all aspects of the iterative lifecycle - requirements and then a release of the prototype.

In an Incremental model, we build the overall solution in parts but at the end of each phase, feedback may not be provided. The model waits until the final stage for the incremental process to deliver the final product.

In the Agile Development Model, customers, developers and testers constantly interact with each other.

3.3 SPIRAL vs. WATERFALL

TABLE 3. COMPARISON OF SPIRAL AND WATERFALL MODEL

SPIRAL MODEL	WATERFALL MODEL
Not fit for small projects.	Fit for small projects.
Risk management is better.	Risk and uncertainty is high.
Process is complex.	Easily to understand.
The process may become hazy .	All Stages are well defined.
Good model for big and ongoing software.	Not good model for big and ongoing software.
It follow Iterations process .	Sequence procces is followed.
User requirements can be changed.	Requirements are fixed .
Easy to Refinements .	Not so easy.
Repetition of Phases are possible.	Not possible.

WATERFALL vs. PROTOTYPE

3.4 WATERFALL vs. PROTOTYPE MODEL

TABLE 4. COMPARISON OF WATERFALL AND PROTOTYPE

Waterfall Model	Prototype Model
Client can only preview the system only after the final version of the software is developed because there is no feedback loop.	Clients have a preview of the system from the "quick design" and the prototype developed early at the end of the process.
Developers encounter a freezing requirement where they are not allowed to modify the requirements or specification of the previous phase until the next iteration.	Developers can refine or add requirements and specification to the system after the prototype is built.
The complexity of an error increases because of the nature of the model; each phase is sequential of the other.	The complexity of an error is low because the prototype enables the developer to detect any deficiency early at the process.

IV. CONCLUSION

After a comprehensive review, it can be concluded that each of these models developed has its limitations and advantages. All SDLC models are conditional as to when to implement which SDLC model it totally depends on nature of the Software. My future work is to minimize the drawback of SDLC models and to propose a new concise model with some new phase and eliminate the drawbacks of spiral and Iterative waterfall model. The effort would also be to implement a new method form SDLC phase in analogy with the requirement gathering phase to estimate size of the software as per the remarks observed in this research work.

V. REFERENCES

[1]. Ian Sommerville, "Software Engineering", Addison Wesley, ISBN-13: 978-0137035151, 9th edition, 2010.

[2]. Rajib Mall, "Software Engineering principle" PHI, 4th edition, ISBN- 978-81-203-4898-1, pp.129–151, April 2014.

[3]. Nabil Mohammed, Ali Munassar and A. Govardhan, 2010," A Comparison Between Five Models Of Software Engineering" , IJCSI , Vol. 7, Issue 5, ISSN (Online): 1694-0814, pp. 94-101

[4]. Rajendra Ganpatrao Sabale, Dr. A.R. Dani, 2012,"Comparative Study of Prototype Model

For Software Engineering With System Development Life Cycle", (IOSRJEN), ISSN: 2250-3021, pp. 21-24

[5]. Adel Alshamrani and Abdullah Bahattab, 2015, "A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model" ,IJCSI,ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784, pp. 106-111

[6]. Vanshika Rastogi," Software Development Life Cycle Models-Comparison, Consequences",2015,(IJCSIT), pp. 168-172

[7]. Ales ZivKovic, Marjan Hericko, Bostjan Brumen ,Simon Belogvlac, Ivan Rozman, 2016," The Impact of Details in the Class Diagram on software Size Estimation",INFORMATICA, Vol. 16, No. 2,pp. 295–312

[8]. Vishwas Massey," Comparing Various SDLC Models And The New Proposed Model On The Basis Of Available Methodology", 2014, IJARCSSE, ISSN: 2277 128X, pp. 170-177

[9]. T Bhuvaneswari, S Prabaharan, " A Survey on Software Development Life Cycle Models",2013,IJCSMC, ISSN 2320–088X, pp. 262-267

[10]. A. Bhoumik,"SQA and Automation /SDLC-models", http://ajitesh2.blogspot.in/p/sdlc-models.html, Accessed on Feb. 10, 2018 [Online].

[11]. N.Mohammed, A.Munassar, A.Govardhan," A Comparison Between Five Models Of Software Engineering", 2010, IJCSI, ISSN (Online): 1694-0814, Vol. 7, Issue 5, pp. 115-121, September10