International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org, editorijaiem@gmail.com
Volume 2, Issue 1, January 2013 ISSN 2319 - 4847

SDLC AND MODEL SELECTION: A STUDY

V. Therese Clara

Asst professor of Computer Science, Madurai Kamaraj University College, Madurai, India

ABSTRACT

In the software industry, the frequency of failure of projects is relatively high. Billions of dollars is eventually lost due to the failed software projects. Lack of proper selection process of software development life cycle (SDLC) models is one of the significant reasons for such failure. Selecting the right software process model fosters a better quality product within a feasible budget and time. The researcher has proposed an approach to select an appropriate SDLC based on different project characteristics.

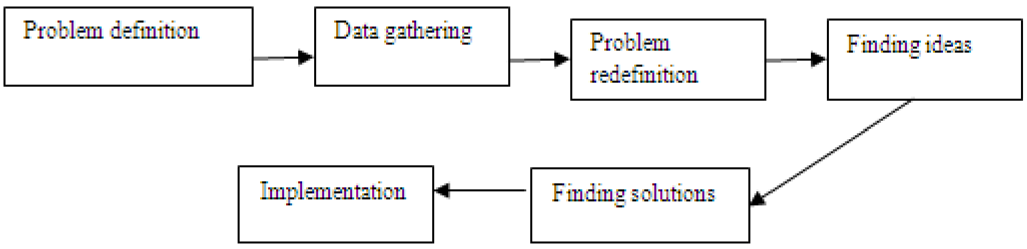
Keywords: SDLC, MODELS, SELECTION, FEASIBILITY.

1. Introduction

The System Development Life Cycle framework equips the system designers and developers with a sequence of activities to be followed during various stages of developing software. Comprehending the basic concepts is imperative to validate the best Software Development Life Cycle (SDLC) methodology. “All software projects go through the phases of requirements gathering, business analysis, system design, implementation and quality assurance testing.” [1] Implementation of any SDLC model depends entirely on the personal choice of the developer. Functionalities of one SDLC model may be better in one situation than in another. This only reinstates the idea that each SDLC model has a great variety in its scope, end user involvement, risk assessment and quality control. Software developers will have to analyze the suitability of the model depending on the functionalities and expectations. “One life cycle model theoretically may suit particular conditions and at the same time other model may also look fitting into the requirements but one should consider trade-off while deciding which model to choose” [2].

2. SDLC

The complexity of creating software was identified as perhaps the most important cause of software problems. The solution to this problem, like to the solution to any problem is to break up the process of solving it into stages. [3]



General problem solving process [3]

The development is subdivided into smaller tasks to facilitate easy management. Subdividing the development process accommodates techniques and skills pertinent to different phases that are to be developed and used. In the event of subdividing the process of software development a life cycle model is developed.

2.1 What is SDLC?

The systems development life cycle (SDLC) refers to the development stage of the system’s life cycle. All systems have a life cyle or a series of stages they naturally undergo. The systems development life cycle (SDLC) is a conceptual model used in project management that describes the stages involved in an information system development project, from an initial feasibility study through maintenance of the completed application. The number and name of the stages vary, but the primary stages are conception, development, maturity and decline.

2.2 Objectives of SDLC

SDLC has the following three primary objectives:

1. To ensure the delivery of high quality systems
2. To provide strong management control

3. To maximize productivity of the systems staff. [4]

2.3 Phases of SDLC

The following are the steps.

Planning – Before the program is created, surveys are conducted to find out the present market needs. In this stage they determine what the developers have to do. Project management plan and other planning documents are developed.

Requirements Analysis - Under this phase analysts analyze the end-user information needs and prepare a solid plan. Functional requirements documents and user requirement documents are developed.

Design - Now the software development team describes desired features, functionalities and operations in detail, including SDLC process and screen layouts, business rules and standards, process diagrams and DFD.

Implementation - Development environment is set up and the design is translated into a program. Programmers carry out some program testing to discover faults in the program and remove these faults in the debugging process.

Maintenance - Software is inherently flexible and can change over its lifetime in response to customer needs. The system is monitored for continued performance in accordance with user requirements and needed system modifications are incorporated.

3. Popular Software Development Models

A software cycle deals with various parts and phases from planning to testing and deploying software. All these activities are carried out in different ways, as per the needs. Each way is known as a Software Development Lifecycle Model (SDLC). [5] A software life cycle model is either a descriptive or prescriptive characterization of how software is or should be developed.
The following are some basic popular models that are adopted by many software development firms

3.1 The Waterfall Model

When requirements are well defined and stable the waterfall model otherwise known as the classical life cycle, with its systematic and sequential approach can be utilized. The sequence begins with communication from the customer regarding specification and progresses through planning, modeling, construction and deployment. If the requirements are fixed and if work proceeds in a linear fashion to complete the project, then the waterfall model is appropriate.

3.2 Prototyping Model

When detailed requirements for functions and features cannot be identified , and when the developer is not sure of the efficiency of an algorithm, the flexibility of an operating system, and the form of human-machine interaction, a prototype concept is utilized. “It is used as a technique that can be implemented within the context of any one of the process models.”[6] The prototype is made after fixing the overall objectives and requirements. The ensuing quick design climaxes in the construction of a prototype. The prototype is checked and refined with the feedback from the end users.

3.3 Rapid Application Development Model

The Rapid Application Development Model (RAD) is an incremental software development process model that emphasizes a very short development cycle. The RAD model is a high speed adaptation of the waterfall model. The rapid development is achieved through component based construction. It results in a fully functional system within a very short period of time if the requirements are well understood and project scope is constrained.

3.4 Component Based Model

The component based development model incorporates many of the characteristics of the spiral model. It is evolutionary in nature [7]. It makes intensive use of existing reusable components. The focus is on integrating the components rather than creating them from the beginning. The project cost and development cycle time can be reduced by incorporating component reuse as part of the organizational culture. The component based model has various steps ranging from requirements specification, component analysis, requirement modification, system design with reuse, development and integration and system validation.

4. Selecting a Software Development Life Cycle

Selecting a Software Development Life Cycle (SDLC) methodology is a challenging task for many organizations. Various software development life cycle models are suitable for specific project related conditions which include organization, requirements stability, risks, budget, duration of project etc. One life cycle model theoretical may suite particular conditions and at the same time other model may also looks fitting into the requirements but one should consider trade-off while deciding which model to choose. There are various methods employed in the industry to adopt a software development model that would be feasible to implement. Some of them maybe based on experience, expertise and even client demands. The crude numerical approach would help us adopt a software development model based on various characteristics of the project as given in [8]

Below mentioned points form the basis of adopting a software model
- Identify the characteristics of the project
- Score each available process model against the characteristics
- The method with the highest score wins

Here is a suitable checklist of characteristics:
- Are the requirements well-established, or ill-defined?
- Are the requirements fixed, or likely to change as the project progresses?
- Is the project small to medium-sized (up to 4 people for 2 years) or large?
- Is the application similar to projects that the developers have experience in, or is it a new area?
- Is the software likely to be is it straightforward or complex (e.g. does it use new hardware)?
- Does the software have a small easy user interface or a large complex user interface?
- Must all the functionality be delivered at once or can it be delivered as partial products?
- Is the product safety critical or not?
- Are the developers largely inexperienced or mainly experienced?
- Does the organizational culture promote individual creativity and responsibility or does it rely on clear roles and procedures?

Placing these in a table, we get:

Project Characteristic	**if true, score 1**	**if true, score 1**
requirements clarity	well-established	ill-defined
requirements change	fixed	changeable
project size	small to medium	large to huge
application	familiar	new
software	straightforward	complex
user interface	simple	complex
functionality	all at once	partial
safety critical	no	yes
developer expertise	largely inexperienced	largely experienced
culture	freedom	Order
User involvement	Minimal	Extensive

5. Concept Application

The concept implied in the above table is applied in a real situation and it is found to be effective.
A local supermarket is all ready to be setup in a city. But, it does not have the necessary software for billing and has no idea how it needs to be setup. The owner of the store decides to have the billing system automated and approaches a leading software firm in the city to get the job done.
The owner briefly explains to the software lead about what he/she needs but falls short of specifying the minute details. He/she explains the process of how billing will be done and cautions the software lead about his/her staff operating the automated system will be college graduates with basic background in operating computers. Before leaving he/she gives

the software team couple of constraints and the automated system should be ready in a short span of time and also the cost should be as minimal as possible.
With the given information in mind the software lead tabulates whatever that was explained and this is how it looks

5.1 Identifying characteristics of project:

Project Characteristic	**if true, score 1**	**if true, score 1**
requirements clarity		ill-defined -1
requirements change		Changeable-1
project size	small to medium-1	
application		New-1
software	Straightforward-1	
user interface	Simple-1	
functionality		Partial-1
safety critical	No-1	
developer expertise		largely experienced-1
User involvement		Extensive-1
Total Score	**4**	**6**

5.2 Characteristics of Iterative model:

Iterative model capabilities	**score**	**score**
requirements clarity	0	1
requirements change	0	1
project size	1	0
application	0	1
software	1	0
user interface	1	0
functionality	0	1
safety critical	0	1
developer expertise	0	1
User involvement	0	1
Total Score	**3**	**7**

The maximum score for the Iterative model is 9 got by multiplying appropriate values form the two tables and the maximum score for the software model characteristics is 10.

5.3 Characteristics of waterfall model:

Waterfall model capabilities	**score**	**score**
requirements clarity	1	0
requirements change	1	0
project size	0	1
application	0	1
software	1	0
user interface	0	1
functionality	1	0
safety critical	0	1
developer expertise	0	1
User involvement	1	0
Total Score	**5**	**5**

The maximum score for the Iterative model is 3 got by multiplying appropriate values form the two tables and the maximum score for the software model characteristics is 10.
Since the Iterative model has more points over the traditional waterfall model based on the numerical approach, it is adopted as the software development model for the above mentioned scenario.

6. CONCLUSION

Going through SDLC, popular software development models one can get an awareness about the existing scenario. Some models based on experience, expertise and client needs vouch for their selection by developers. If one is sure of suitability of software development model characteristics to the project requirements, then it is easier to select an SDLC. The concept application illustrates and substantiates the validity of the chosen software development model. The questionnaire in the tabular format facilitates an easy framework and the interpretation of scores depending on software characteristics and project requirements enables a software developer to choose an appropriate model.

REFERENCES

[1] Klopper, R., Gruner, S., & Kourie, D. (2007),0||Assessment of a framework to compare software development methodologies||, *Proceedings of the 2007 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries,* 56-65. doi: 10.1145/1292491.1292498
[2] Software Methodologies Advantages & disadvantages of various SDLC models.mht
[3] Bennet, McRobb, and Farmer Object Oriented Systems Analysis and Design(Mc raw Hill 2002)p.46
[4] Bender RBT Inc., Systems Development Lifecycle: Objectives and Requirements
[5] Raymond Lewallen - CodeBetter.Com - Stuff you need to Code Better! Published 08-01-2008
[6] Pressman Roger S., Software engineering pp-33.
[7] Nierstraaz, O., s. Gibbs, and D. Tsichritziz, “Component – Oriented Software Development,”pp160-165
[8] www.shu.ac.uk/.../project%20management%20-%20new%20version.doc