Changing Scenario of SDLC and Role of SRS in the success of Software

Miss. Vrushali V. Jadhav
Lecturer, jadhav.vrushali@rediffmail.com

Ms. Archana R. Ugale
Lecturer, archu_aag@rediffmail.com

Dr. Nitin D. Nayak
Director, nayak78@yahoo.co.in
Bharati Vidyapeeth University
Institute of Management and Rural Development Administration, Sangli,(MS)

ABSTRACT
Software Requirement Specification (SRS) is a document which enlists all requirements of users from particular software. Although producing of SRS depends on Process Model, it is produced at the end of analysis phase. It contains four components: functional requirements, performance requirements, design constraints, and user interfaces. In functional requirements, main requirement from software are written. Performance requirement component specifies various static and dynamic performance expectations from software. Design Constraints component is for writing report format, H/W and other S/W constrains. Last component i.e. Interface gives User Interface requirement, H/W interface requirement and S/W interface requirement. These four components give all possible requirements of software under considerations. SRS plays very crucial role in the overall success of software project.

Traditionally, SRS was being made by system analyst or software engineer. When any developer or Developer Company was getting order from any client, developer used to start write from Technical Feasibility, System Analysis and other phases or steps. But now client contains employees who can create SRS very well. As these employees know system expectation very well they can write SRS which is correct, traceable and consistent. So now clients provide SRS. Thus traditional SDLC is divided into two sites, one is Client and other is Developer.

The paper discusses this new changing scenario of software development life cycle and importance of SRS in software development process. The paper concentrates on companies like Reliance, Blue Point, Tech Mahindra, Dolphin

KEYWORDS
SRS, Client Side Software Analysis, User Interface

INTRODUCTION
Software Development Life Cycle (SDLC), a series of different phases to make a good, qualified software. Since 1960’s, SDLC is conducted using different process models which has been formulated from familiar “Waterfall” model.

These classic software process models usually include some version or subset of the following activities: [1]
- System Initiation/Planning
- Requirement Analysis and Specification
- Functional Specification or Prototyping
- Partition and Selection (Build vs. Buy vs. Reuse)
- Architectural Design and Configuration Specification
- Detailed Component Design Specification
- Component Implementation and Debugging
- Software Integration and Testing
- Documentation Revision and System Delivery
- Deployment and Installation
- Software Maintenance

The apparent purpose of early software life cycle models was to provide a conceptual scheme for rationally managing the development of software systems. Such a scheme could therefore serve as a basis for planning, organizing, staffing, coordinating, budgeting, and directing software development activities. [1]
Software process models often represent a networked sequence of activities, objects, transformations, and events that embody strategies for accomplishing software evolution. Such models can be used to develop more precise and formalized descriptions of software life cycle activities and people involved in it. We can categorize the people involved in software development as client, end users and developers. Client is who gives order for software. End-users are that persons who will actually operate on software and developer are a team who will create that system.
Within all software life cycle activities, writing of Software Requirement Specification (SRS) is an integral part. The SRS contains all functional and non functional requirements from the software under consideration. There are many good definitions of Software Requirements Specifications that will provide a good basis upon which we can define a great specification.

ROLE OF SRS
There is no any statistical data for questions like “What is the role of SRS in software development life cycle? How much it is
important?” Answers to these questions may vary depending on scope of projects, type of projects, experience of persons etc. One questionnaire was made regarding this issue which contained questions like-

- Do you use SRS?
- Rank importance of SRS in Software development of your company
- Do you think SRS play an important role in the success of software? Why?

The questionnaire was distributed to 75 different IT professionals from companies like Reliance, Kanbay, TachMahindra, BluePoint, Dolphine etc. From this, 50 questionnaires were filled and returned back. The statistical analysis of these questionnaires is as follows.

- Do you use SRS?

![Figure-1](Do you use SRS?)

Figure-1- Response to question ‘Do you use SRS?’

Figure-1 clearly indicates that 90% of total samples are using SRS for development and only 10% are not using it.

- Rank importance of SRS in Software development of your company

For this question, we gave ranks from 1 to 10, being 1 as not important and 10 as critically important.

![Figure-2](Rank importance of SRS)

Figure-2- Response to question ‘Rank importance of SRS?’

Figure-2 shows responses for the question. Most of the persons have given high rank to importance of SRS. Thus, more than 70% of samples give much importance to SRS.

- Do you think SRS plays an important role in the success of software? Why?

Although the reasons given by samples vary, hundred percent of the sample have accepted the importance of SRS in success of software. The reasons contain-

1. It gives us the requirements of clients and after discussion with clients we can suggest them some changes and we can explain them how these changes are helpful in their software.
2. If it is clearly specified and if there are unchanging requirement then, it is more useful to complete project within time limit.
3. It follows all design procedures & specifications of software. Also it is best way to communicate with user, which is important for understanding needs of users so that we can develop good software.
4. It helps developer to prepare the system in right time and up to mark. It also helps end user to understand how the system is created and how does it works.

All process models suggest that the team which is working for analysis makes this SRS by taking requirement from user, freezes SRS and then starts for further design. While making SRS, user should be involved so that requirements can be clearly defined. This principle is followed in every process model while producing SRS and it is stamped to the project database after inspections and verification.

Figure-3 shows the importance of combination of People, Process and Technology for making any software [2]. People mean Developer, Users and Clients. Process means the network of activities that are expected to be conducted in software development. Technology is the tools used for creating software. In fact, the “people factor” is so important that the Software Engineering Institute has developed a people management capability maturity model (PM-CMM). It is “to enhance the readiness of software organizations to undertake increasingly complex applications by helping to attract, grow, motivate, deploy and retain the talent needed to improve their software development capability”[3].

CHANGING SCENARIO

- This was the general scenario of software development up to 2000. The activities of SDLC and sequence of conducting these activities depends on customer, time to market, object technology,
Changing Scenario of SDLC and Role of SRS in the success of Software

networking, shift to economy. Figure 4 shows the factors affecting activities of SDLC.

Figure-4 Factors affecting activities of SDLC.

It is essential that software engineering tool and techniques be used with an eye toward flexibility [4].

Now things are changing. Customer is the king of market. What client needs has to be there in the software or must be fulfilled. Also client is more aware of product and services available in market. More trace is given on Quality, Cost and Delivery. Another thing is that client or any firm which wants to have any application software or utility software is a small scale industry or a private limited company. It has enough number of employees who are knowledgeable frequent users of computer and in many cases they are computer graduates. So the idea of making SRS at client’s site is now emerging. The users can recognize their requirements very well and as they are computer graduate they can define requirements in developer required format. So both intentions are fulfilled and fewer efforts will be required for making good SRS and good software.

We included questions like
- Who makes SRS?
- In traditional SDLC, SRS is made by developers. But now as client has employees from computer field, can client provide SRS and specify to the point views of software under consideration?
- If client is providing SRS, is there any need to change that SRS or you follow it as it is?
- If client can make SRS, do you think the traditional SDLC is changing? Can you give your Views?

Many samples supported the idea of creating SRS by clients. Some employees from TechMahindra answered that now a - day clients give order with SRS only. The reason they have given is that “now client contains computer graduate employees who can specify requirements in developer required form”. It becomes very much useful for further development. SRS is the last document that is produced at the end of analysis phase and as client is writing SRS, so we can say that analysis phase is conducted at client site. Taking this concept as base, we can divide SDLC in two sites, client site and developer site.

A TRY TO DIVIDE SDLC IN TWO SITES
Here we have tried to rearrange the phases of SDLC in client and developer. Figure-5 is graphical representation of it.

Figure-5 indicates the bifurcation of phases in two sites. Thus, System Initiation phase and Requirement Analysis Phase is conducted at client site. We have called it as Client Side Software Analysis. In these phases client finds out proposed solution, analyzes the system and determines business requirement. Another change in traditional SDLC is making of User Interface Design. This activity varies project wise. We can’t say that making of UI is compulsory in this approach. If client has hard constraints regarding UI, he can specify UI in SRS itself. The important work for client is “Producing SRS”. This SRS is given to developer. After getting SRS developer starts to find a process model which fits Software. Iterative Process Model or Agile Software Process model is supposed to be suitable. So in this approach, selecting Software Process Model has moved to design phase. Whatever process model is followed, it is only for System Design, System Construction, Acceptance and Maintenance.

PROJECT AND PRODUCT BASED COMPANY
When we collected data regarding this approach from IT field, many of them divided client companies in to two types. One is Project Based Company and another is Product Based Company.

In Project Base Company SRS provided by customer has to be followed and fulfilled.

In Product Base Company, before and after delivering the product to the customer, if there is some changes in the requirement, client gives CR-SRS. then it depends on company policy whether to accept it or not.

If SRS is related to GUI, then it has to be followed as it is. If it is related to functionality of project or product, then one can change it depending on development process.

ADVANTAGES OF THE APPROACH
1. As end user is knowledgeable frequent user, he can make SRS which contains all requirements. So to the point requirement can be specified.
2. It decreases the time, efforts and money required for project initiation.
3. It decreases the time efforts and money needed for analysis because client is supposed to conduct this phase.
4. Up to some extent user can do maintenance also
LIMITATIONS OF THE APPROACH
1. The effect of this approach is possible only if users are knowledgeable frequent users.
2. There may be need to change SRS after getting it from user so that it should be in developer compatible format.
3. It may make problem of communication among user and developer.

CONCLUSION
Software Engineering has evolved a lot since 1990s. Software companies always try to identify new ways to dramatically improve and transform software processes [5]. Depending on market conditions and situation changes in the traditional ways of SDLC and Software Process Models has done. As customer is king of market, some changes which are flexible have made in the activities of SDLC. We are not claiming here that complete SDLC has changed. But the sequence of activities, producer of SRS has changed. In new scenario, client is completing overall analysis phase, so in future new process model can evolve which will accommodate this new changes.

REFERENCES
3. Roger Pressman, Software Engineering, A Practitioner’s Approach’, Ed. 5th, pp 56-57