Chapter 10

SRS BUILDER 1.0: An Upper Type CASE Tool For Requirement Specification

10.1 Introduction

Although, hardware costs are decreasing drastically, still the computers are not used extensively by the business organization because of the huge software cost. In recent years, Computer Aided Software Engineering (CASE) tools have emerged as one of the most important innovations in software development to manage the complexity of software development projects reducing its product cost. Using CASE tools over their SDLC process may reduce the developing cost significantly.

Although, almost all develop countries do use certain CASE tools, but its extensive use is still a dream because of their product cost. Although, big software giant's do use their own developed or commercially available CASE tools in development software, but their quality, applicability, cost, availability remains as big question. The huge cost of such commercially available CASE tools made these outreach of the small software development companies. In this study, we are going to demonstrate newly developed requirement specification tool name SRS BUILDER version 1.0.

The rest of the paper is organized as follows: We started by defining software engineering, Computer Aided Software Engineering and CASE tools. Then, we have laid down the different types of CASE tools and advantage of using CASE tools with its limitation. In the later section we have discussed about SRS BUILDER 1.0 with its sample design.

Based on the publication in the "*Proc. of The* 4^{th} *National Conference on Computing for National Development(INDIACOM-2010)*", February 25–26, 2010, ISSN: 0973–7529, ISBN: 978-81-904526-9-4

10.2 Computer Aided Software Engineering (CSAE) and CASE Tools

In the following we are going to define some terminologies used in software engineering.
Software Engineering (SE): Before defining CASE, reminding the definition of software engineering is justifiable. Although, the age of Software Engineering is quite old, but prior to its born, people do used to develop software. But because of some problems (discussion of these problems are beyond the scope of paper) faced later with those systems, software engineering emerged as a new subject.
The IEEE defines software engineering as (19) :
(i) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
(ii) The study of approaches as in (i).
More significantly, software engineering may be defined as the systematic approach to develop quality software within both budget and time constraints.
Computer Aided Software Engineering (CASE): CASE is an acronym that stands for Computer Aided Software Engineering. Roughly, this is all about using computers at different phases of the SDLC process during the development and maintenance of software to assistance the development team. CASE provides the software engineer with the ability to automate manual activities and to improve engineering associated to software development. Basically, it is all about using software to develop software.
CASE tools: CASE tools are the set of tools that permit collaborative software development and maintenance. These tools are concerned with automated tools that aid in the definition and implementation of software systems.
Formal definition of CASE:

- *Individual tools to aid the software developer or project manager during one or more phases of software development (or maintenance).*
- *A combination of software tools and structured development methodologies.*

Types of CASE tools: Depending on the activities performed, CASE tools are primarily divided in to three categories. They are as follows:

1. **Upper CASE tools:** Primarily focuses on the System analysis and design phase of the SDLC.

2. **Lower CASE tools:** Focuses on system implementation phase of SDLC.
3. **Integrated CASE tools:** It helps in providing linkages between the lower and Upper CASE tools.

10.3 Advantages of Using CASE Tools

With the growing importance of CASE tools, more steps in the SDLC are being automated. However, a complete automated software facility for all steps is still a dream. Presently available CASE tools cover only a certain modules of the general CASE facility. The benefits of using CASE tools are as follows:

1. Increased Productivity.
2. Product Quality improvement.
3. Development Cost Reduction.
4. Effort Reduction: Different studies carried out to measure the impact of CASE put the effort reduction about 30–40% (126).
5. Reduction in Development Time.
6. Reduce the drudgery and working style in a software engineer's work.
7. Create good quality documentation. Our, proposed tool basically focuses on this particular aspect of computer aided software engineering.
8. Create maintainable system.
9. Providing a uniform platform for software/system developers to present information and knowledge compactly for ease of communication (181, 182, 183).

10.4 Usage of CASE Tools: A Research Report

CASE (Computer Aided Software Engineering) tools are supposed to increase productivity, improve the software product quality and make Information Systems development a more enjoyable task (184). However, they have been failing to deliver the benefits they promise (185). The results of the research carried out by Kemerer (186) regarding the usage of CASE tools after introduction are shown in the Figure 10.1.

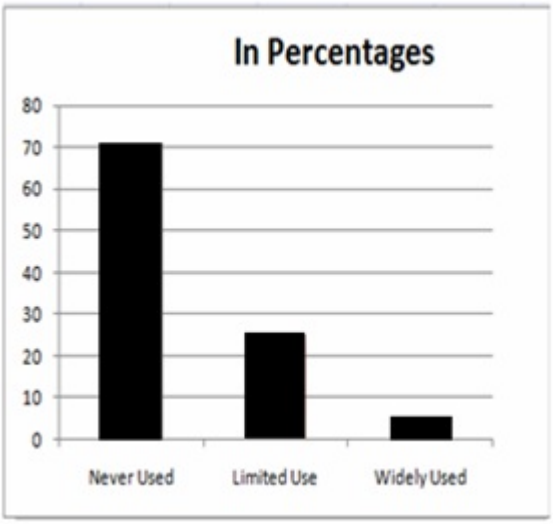


Figure 10.1: Use of CASE Tools in Organizations (in %)

This results about CASE tool usage raise a significant question, why the percentage of widely used case tools are too low i.e. 5% only?
The answer to the question is the limitations with the CASE tools discussed in the next section.

10.5 Limitation of CASE Tools

CASE tools are still in limited use because of the following limitations:

1. CASE tools don't support automatic development of functionally relevant system.
2. It force system analyst to follow a prescribed methodology.
3. It may change the system analysis and design process.
4. Poorly supported expected functionality from them.
5. Huge cost of the CASE tools.
6. Lack of Concern in CASE tool usage.
7. Bad quality of the CASE tools.

8. CASE tools are complex because they offer a large array of options and support for software development activities.

9. Cheap Labor cost: In developing countries like India and other underdeveloped countries, cheaply available human resources remain as one of the biggest reasons for not using the costly CASE tools.

10.6 Motivation for Undertaking the Research Project

In addition to usage in industry by practitioners, CASE tools are employed by educators to teach students software development skills. Evaluating and selecting a CASE tool for a specific systems development course is quite difficult.
While teaching the paper software engineering at university (NBU), it was found that the students are very much confused about CASE tools. The significant reasons for the same are basically widely available complex poor quality CASE tools which are quite expensive to purchase although. Some modules are good in some if not in all. The students do not feel interesting to use them. Then we under took the project to develop a new customized CASE tool for requirement specification supporting IEEE specification as per the students need that will help the student to improve their skill and have a clear vision about it. We start with the development of CASE tool to specify the system requirements to generate the System Requirement Specification (SRS) document. The outcome of our attempt is the so named, *SRS BUILDER 1.0*– the CASE tools to generate the SRS document. Moreover, we are planning to distribute the newly developed CASE tool to the educational institutions on demand almost free of cost with a nominal transportation cost for the sake of student community.

10.7 SRS BUILDER 1.0: The New Requirement Specification Tool

As from the SLC models, we know that after the requirements are gathered by the system analyst, the analyzed and finalized requirements need to be specify in the SRS document. The SRS document is then provided to the software development team for next phase of the development.

- **SDLC Model Followed**
  The BRIDGE software development process model (62) has been followed to develop the SRS BUILDER 1.0.

- **Product Quality Features**
  - Support to IEEE specification for SRS writing format along with the flexible other customized specifications as per your organizational need.
  - Good Graphical User Interface
  - Easy to use
  - Easy Installation
  - Incorporated User Documentation
  - Easily Available
  - Validated
- **System Specifications**
  - Front End: Visual Basic 6.0
  - Back End: MySQL
  - Operating System: Windows XP, Windows Vista.
  - Memory Requirement: 15 KB

10.8 Function Hierarchy Diagram of SRS BUILDER 1.0

The Function Hierarchy Diagram (FHD) of the SRS BUILDER 1.0 is shown below in Figure 10. 2:

10.9 Example: Sample SRS Organization Generated by SRS BUILDER 1.0

A typical SRS generated by the SRS BUILDER 1.0 for the ATM system is shown below in Figure 10.3 to Figure 10.6. This is not a complete and correct SRS for the intended system, but a sample used only to show the SRS organization generated by the tool. The font size and spacing has been changed to accommodate in the paper keeping the context organization unchanged.

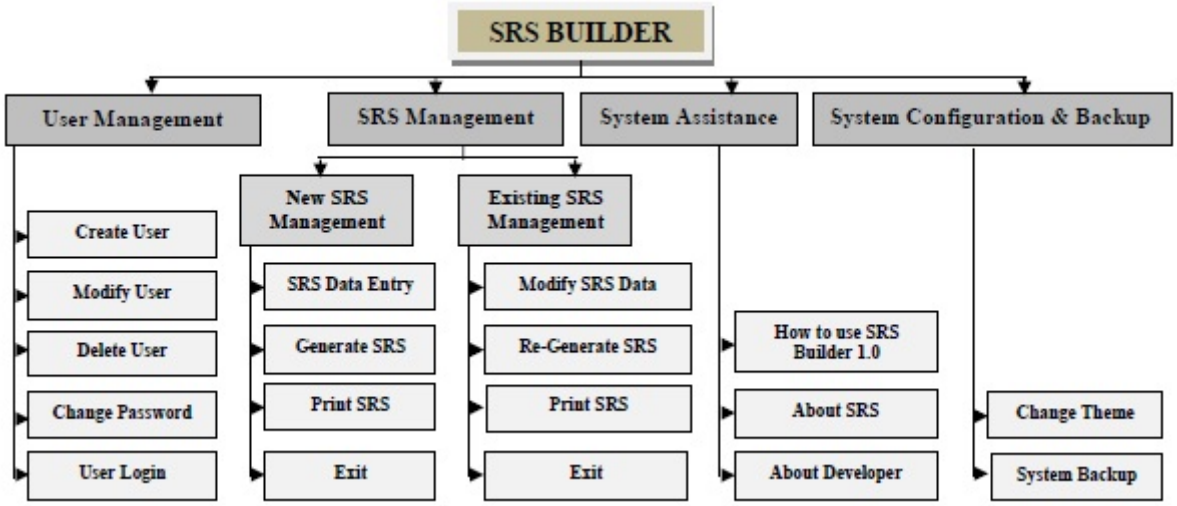


Figure 10.2: FHD (Function Hierarchy Diagram) of SRS Builder 1.0

Project Title: SRS ATM

Project Id: 1

SOFTWARE REQUIREMENT SPECIFICATION

Introduction

Purpose:

This document describes the software require……

Scope:

The software supports a computerized banking …….

Definition:

• Account:

A single account at a bank against which transaction……..

Intended Audience:

The intended audience of this SRS consists of:

• Software designers….

Reference: NA

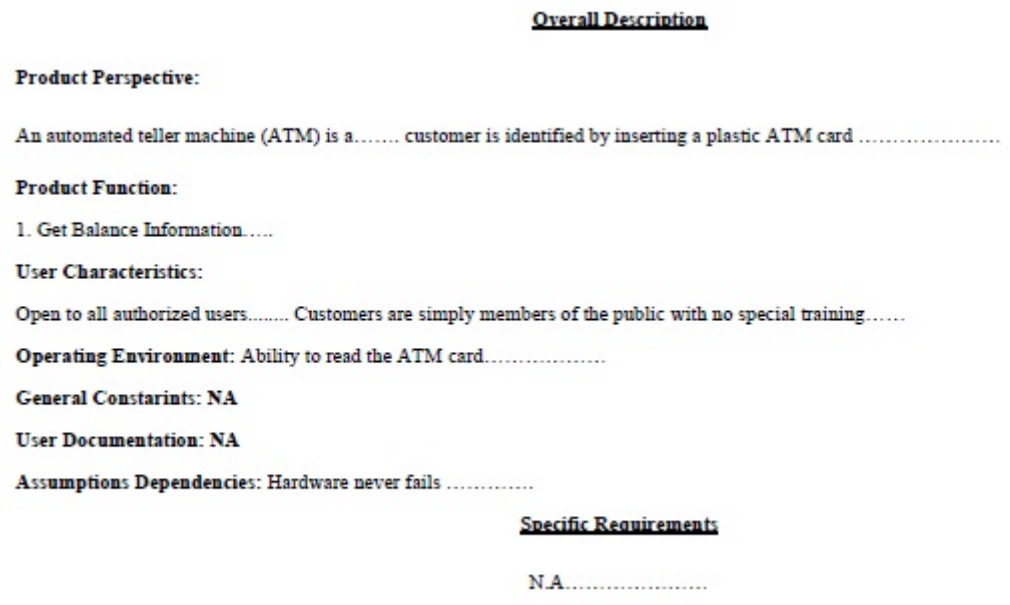
Overview: NA

Document Conventions: NA

Figure 10.3: A sample SRS Structure Generated Using SRS Builder 1.0 (to be continued...)

10.10 Conclusion

We may conclude by pointing out that, this CASE tool will play an important role to the software developers and learners to use and understand the utility of the CASE tool in



Overall Description

Product Perspective:

An automated teller machine (ATM) is a……. customer is identified by inserting a plastic ATM card ……

Product Function:

1. Get Balance Information…..

User Characteristics:

Open to all authorized users..... Customers are simply members of the public with no special training……

Operating Environment: Ability to read the ATM card……

General Constarints: NA

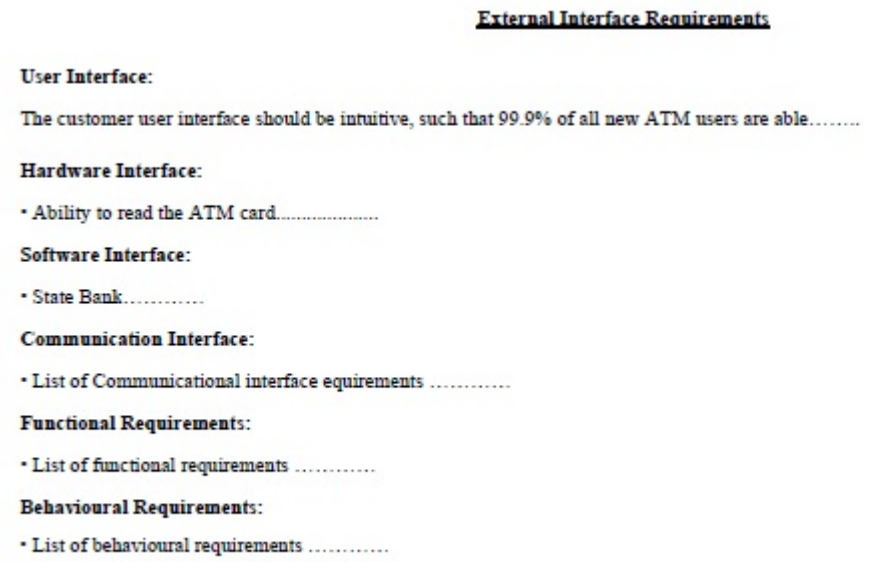
User Documentation: NA

Assumptions Dependencies: Hardware never fails ……

Specific Requirements

N.A……

Figure 10.4: A sample SRS Generated Using SRS Builder 1.0 (to be contd...)



External Interface Requirements

User Interface:

The customer user interface should be intuitive, such that 99.9% of all new ATM users are able……..

Hardware Interface:

• Ability to read the ATM card……

Software Interface:

• State Bank……

Communication Interface:

• List of Communicational interface equirements ……

Functional Requirements:

• List of functional requirements ……

Behavioural Requirements:

• List of behavioural requirements ……

Figure 10.5: A sample SRS Generated Using SRS Builder 1.0 (to be contd...)

todays'complex software projects. Also, as we mentioned earlier, interested educational institutions and organizations may contact the author for the CASE tool for their usage.

Other Non-functional Requirements

Performance Requirements:

• It must be able to perform in adverse conditions like high/low temperature etc. .....

Safety Requirements:

• Must be safe kept in physical aspects, say in a cabin .....

Security Requirements:

• Users accessibility is censured in all the ways .....

Software Quality: NA

Other Requirements: NA

SYSTEM REQUIREMENTS SPECIFICATION for ATM Withdrawal

Submitted by:

Program Manager/Functional Project Officer	Date

Coordination:

Director, Applications Architecture	Date
Director, Engineering	Date
Test Director	Date

Approved by:

Functional Manager	Date

Figure 10.6: A sample SRS Generated Using SRS Builder 1.0