SITUATIONAL METHOD ENGINEERING FOR REQUIREMENT ENGINEERING PHASE

A MASTER'S THESIS

in

Computer Engineering

Atilim University

by

SEÇİL AYDIN

JULY 2009

SITUATIONAL METHOD ENGINEERING FOR

REQUIREMENT ENGINEERING PHASE

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

ATILIM UNIVERSITY

BY

SEÇİL AYDIN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF COMPUTER ENGINEERING

JULY 2009

Approval of the Graduate School of Natural and Applied Science, Atılım University

Prof. Dr. Abdurrahim ÖZGENOĞLU

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. İbrahim AKMAN

Head of Department

This is to certify that we have read the thesis "Situational Method Engineering for Requirement Engineering Phase" submitted by "Seçil Aydın" and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Deepti MISHRA

Supervisor

Examining Committee Members

Prof. Dr. Ali YAZICI

Assoc. Prof. Dr. Alok MISHRA

Assoc. Prof. Dr. Mohammed REHAN

Asst. Prof. Dr. Erol ÖZÇELİK

Asst. Prof. Dr. Deepti MISHRA

Date: 31/07/2009

I declare and guarantee that all data, knowledge and information in this document has been obtained, processed and presented in accordance with academic rules and ethical conduct. Based on these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Seçil AYDIN

ABSTRACT

SITUATIONAL METHOD ENGINEERING FOR

REQUIREMENT ENGINEERING PHASE

Aydın, Seçil

M.S., Computer Engineering Department

Supervisor: Asst. Prof. Dr. Deepti Mishra

July 2009, 75 pages

This thesis focuses on requirements engineering phase and reviews the existing requirement engineering methods and compares them according to the constraints in the software projects. It has been found that some techniques are better suited to particular project teams and circumstances. Besides, methods are normally general in nature and they can not be used directly without adapting them according to the characteristics of the project. This is the concern of situational method engineering, where the term situational method is used to refer to a method tailored to the needs of a particular development setting. A criterion methodology is established to distinguish requirement engineering methods from each other according to different characteristics of the project. A tool is implemented to store different methods according to this criterion methodology by using Situational Method Engineering. This tool is compared with other tools that exist in the literature. The tool is published and validated by collecting data from the industry. The results gathered from the industry are presented and discussed for the improvement of proposed approach.

Keywords: Requirement engineering, Situational method engineering

ÖZ

GEREKSİNİM MÜHENDİSLİĞİ FAZI İÇİN DURUMSAL YÖNTEM MÜHENDİSLİĞİ

Aydın, Seçil

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Yrd. Doç. Dr. Deepti Mishra

Temmuz 2009, 75 sayfa

Bu calışma, gereksinim mühendisliği fazına odaklanmış, varolan gereksinim mühendisliği fazının yöntemlerini incelemekte ve yazılım projelerindeki kısıtlamaları göz önünde bulundurarak karşılaştırmaktadır. Bazı projeler ve durumlar için bazı yöntemlerin daha uygun olduğu anlaşılmıştır. Bu yöntemler; projenin özelliklerine duruma ve projeye göre biçimlendirilmelidir. Bu durumsal yöntem mühendisliğinin konusudur ve yöntemin geliştirilmekte olan projenin ihtiyaçlarına göre biçimlendirilmesidir. Yöntemleri değişik proje özelliklerine göre birbirinden ayırmak amacıyla, yeni bir kategorilendirme stili yaratılmıştır. Bu kategorilendirme stili ve durumsal yöntem mühendisliği kullanılarak yeni bir program geliştirilmiştir. Bu program varolan programlarla karşılaştırılmıştır. Yayınlanmış ve sektörden gelen geri bildirimler toplanmıştır. Bu geri bildirimler sunulmuş ve ileri sürülmüş olan yöntemin geliştirilmesi için tartışılmıştır.

Anahtar Kelimeler: Gereksinim mühendisliği, Durumsal yöntem mühendisliği

To My Parents, for their endless support

To My Advisor, for her trust

ACKNOWLEDGMENTS

I express sincere appreciation to my supervisor Asst. Prof. Dr. Deepti Mishra for her guidance and insight throughout the research. It has been a great experience for me to work with her.

I should also express my appreciation to my examination committee members Prof. Dr. Ali YAZICI, Assoc. Prof. Dr. Alok MISHRA, Assoc. Prof. Dr. Mohammed REHAN, Asst. Prof. Dr. Erol ÖZÇELİK for their valuable suggestions and comments.

I also would like to thank respondents of the questionnaire for their valuable comments, my friends for their support and encouraging, my colleagues at Bilkent University for their support.

Above all, I am very grateful for the endless love and support of my parents Cevher and Selahattin Aydın and my grandmother Şükran Kılavuz. I feel stronger and happier with their love and support.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ACM	-	Association of Computing Machinery
CASE	-	Computer Aided Software Engineering
CAME	-	Computer Aided Method Engineering
CMM	-	Capability Maturity Model
IEEE	-	Institute of Electrical and Electronics Engineers
ME	-	Method Engineering
RE	-	Requirements Engineering
RS	-	Requirements Specification
SEI	-	Software Engineering Institute
SME	-	Situational Method Engineering
SRMS	-	Situational Requirement Method System

CHAPTER 1

1. INTRODUCTION

1.1 General View and Terminology

Software is instructions (computer programs) that when executed provide desired features, function, and performance; data structures that enable the programs adequately manipulate information and documents that describe the operation and use of the programs [Pressman, 05].

Software engineering is “the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software” [IEEE, 93]. Software engineering is a layered technology. It has three layers as shown in Figure 1.1. Software engineering must rest on an organizational commitment to quality. Software engineering process is the glue that holds the technology layers together and enables rational and timely development of computer software.

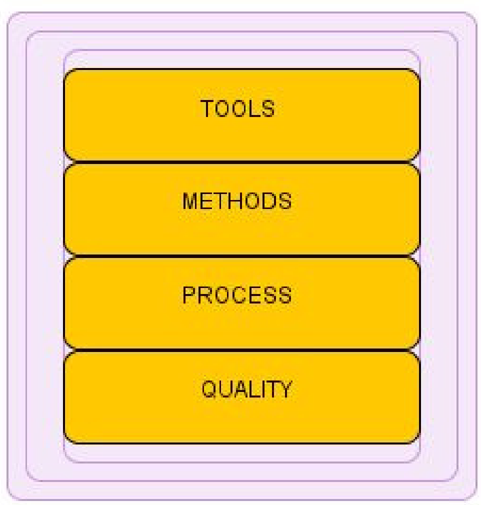


Figure 1.1 Layers of Software Engineering [Ucertify, 08]

Software engineering methods provide the technical instructions for building software. Methods surround a broad array of tasks that include communication, requirement analysis, design modeling, program construction, testing and support. The last layer software engineering tools; provide automated or semi-automated support for the process and the methods [Pressman, 05].

Software development is the set of activities result in software products. It has own process the get the product. There are many software development process models [Pressman, 05], [Hughes&Cottrell, 06], [Pillai,96], [Sommerville, 95], [Ravindran, 07] existing. They describe approaches to tasks and activities that will be used during the process. Software development process has number of activities; it is an iterative learning process and the outcome. These activities are; communication, planning, modeling, construction and maintaining. When communicating, the important task in creating a software product is extracting the requirements or requirements analysis so the formal definition of requirements engineering supports this entirely. *Requirements engineering,* is a software engineering action that begins during the communication activity and continues into modeling activity [Pressman, 05].

The functionality and characteristics of the system to be built must be elicited and documented in a requirements specification. Customers typically have an abstract idea of what they want as an end result, but not what software should do. Incomplete, ambiguous, or even contradictory requirements are recognized by skilled and experienced software engineers at this point so requirements engineering is the most important part of the software development because the product is directly dependent with the requirements we elicit. Defects are the errors that are found in the product after product is delivered to customer. Requirements errors top the delivered defects and contribute approximately one third of the total delivered defects to the defect pile so requirement errors are the most common category of system development errors [Leffingwell&Widrig, 2000]. Table 1 shows the portion of the requirements defect in the software development.

Table 1. Requirements Defect Summary

Defect Origins	*Defect Potentials*	*Removal Efficiency*	**Delivered Defects**
Requirements	1.0	77%	0.23
Design	1.25	85%	0.19
Coding	1.75	95%	0.09
Documentation	0.60	80%	0.12
Bad fixes	0.40	70%	0.12
Total	**5.00**	**85%**	**0.75**

There are specific methods used in different phases of the software development process. *Method* describes a regular and systematic way how to accomplish something. In the domain of Information Systems engineering, Brinkkemper defines a method as "an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rule, structured in a systematic way in development activities with corresponding development product" [Arni-Bloch, 99]. To avoid facing many defect problems, as mentioned before there is a need for formalizing the use of methods for systems development. Method engineering is used for formalizing the methods. Similar to software engineering definition, *method engineering* is defined as the engineering discipline to design, construct and adapt methods, techniques and tools for systems development [Brinkkemper, 96].

Methods that are constructed should be organization-specific or project specific to get more benefit from method construction which is supported by a major component of method engineering: *situational method engineering*. *Situational method engineering* encompasses all aspects of creating a development method for a specific situation [Serour&Sellers, 04], [Mirbel&Ralyte, 05], [Coulin&Zowghi&Sahraoui, 06]. It focuses of the formalization of methods in terms of reusable method components and the definition of assembly techniques allowing to construct new methods by reusing these components.

As mentioned before, projects require tailored methods and tools to support the actual processes. By the help of situational method engineering; organizations can construct tailored methods adapted to organization and the projects at hand. Since incorrect requirements are the biggest cause of software project failure, focus of this work is requirement engineering phase. The problem is how best to construct, select and adapt requirement engineering methods for specific projects. Requirement elicitation is the first step in requirement engineering process. Requirements elicitation is defined as the process of identifying needs and bridging the disparities among the involved communities for the purpose of defining and distilling requirements to meet the constraints of these communities [SEI, 91].

Just a variety of techniques can be used for analyzing and designing software solutions and also a variety of techniques can be used to understand user and stakeholder requirements. Every technique has some advantages and disadvantages. They can not be used as it is. They should be tailored according to the situation, project at hand. Some techniques are better suited to particular project teams and circumstances. Because of the relative strengths and weaknesses of the available approaches, most projects normally require a combination of several techniques in order to produce quality results [Maiden&Rugg, 96]. Clearly requirement elicitation does not occur in a vacuum and is highly dependent on the specific project, organizational and environmental characteristics [Christel&Kang, 92]. Different requirement elicitation techniques and suitability of these techniques in different project situations are studied. With this knowledge, a criterion is formed that classifies different requirement elicitation techniques. These criteria are used during method creation as well as during method selection. Although there exist several studies in the literature which explores SME, but there are not much tool to use for this issue. In this study, a tool is developed to demonstrate the stated approach called Situational Requirement Method System (SRMS). SRMS is developed with PHP. MySQL database is used to store the method related information. Moreover, for the graphical part JpGraph [JpGraph, 09] which is an object oriented graph creating library is used. SRMS tool is a web-based tool because project members can be in different places and therefore tool can be easily accessed by the members of the project.

1.2 Thesis Overview

The remainder of this thesis is organized as follows. In Chapter 2, detailed information about the literature background of the requirement engineering, problems in requirement engineering, method engineering and situational method engineering are given. Main contribution of this study and detailed information about the tool are presented at Chapter 3. Tool validation is also described in Chapter 3. Conclusion and possible future issues are presented in Chapter 4.

CHAPTER 2

2. LITERATURE SURVEY

2.1 Requirements Engineering

2.1.1 Definition

Although many definitions of software requirements have been used throughout the years, the one provided by requirements engineering authors Dorfman and Thayer (1990) is quite workable [Leffingwell&Widrig, 2000]:

- A software capability needed by the user to solve a problem to achieve an objective
- A software capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documentation.

Another definition of requirement is; what the system should do, defined in terms of its external environment; largely what the user wants the system to do [Sommerville, 95].

With the light of these definitions, requirements engineering is defined. From a software process perspective, requirements engineering is a software engineering action that begins during the communication activity and continues into the modeling activity [Pressman, 05].

[IEEE, 90] defines requirements engineering mainly as requirements analysis: (1) The process of studying user needs to arrive at a definition of system, hardware, or software requirements; (2) the process of studying and refining system, hardware or software requirements.

Requirements management defined as a systematic approach to eliciting, organizing and documenting the requirements of the system, and a process that establishes and maintains agreement between the customer and the project team on the changing requirements of the system [Leffingwell&Widrig, 2000]. Its purpose is establishing a good communication between system users and system developers. According to CMM, goals of requirements management are: (1) System requirements allocated to software are controlled to establish a baseline for software engineering and management use. (2) Software plans, products and activities are kept consistent with the system requirements allocated to software [Wang&Lai, 01].

In the literature, there are different types of requirements defined by researchers. [IEEE, 90] defines five requirement types, these are: functional, performance, interface, constraint and quality. Despite this classification, many of the researchers defines two requirement types, which are: functional and non-functional requirements [Ghose, 99], [Sommerville, 95], [Mylopoulos&Chung&Nixon, 92].

A *functional requirement* is a software requirement that specifies a function that a system or its component must be capable of performing. These are software requirements that define behavior of the system, that is, the fundamental process or transformation that software and hardware components of the system perform on inputs to produce outputs [Thayer&Dorfman, 97].

A *non-functional requirement* is a software requirement that describes not what the software will do, but how the software will do it. Examples include software performance requirements, software external interface requirements, software design constraints and software quality attributes. Non-functional requirements are usually difficult to test; therefore, they are usually evaluated subjectively [Thayer&Dorfman, 97].

2.1.2 Problems in Requirements Engineering

Writing good, correct and complete and measurable system and software requirements specifications are a major problem nowadays. The major problems are listed in the following [Thayer&Dorfman, 97]:

- The desire of managers to truncate requirements activity because they believe that the major effort in any software development is programming and testing.
- The inability of engineers to write correct software requirements specification.
- The lack of cooperation by customers when it comes to verifying that the software requirements are correct, and the lack of understanding the structure and purpose of the software requirements specifications.
- The problems associated with identifying which tool and/or methodology to use in developing and representing a software requirements specification.
- The lack of knowledge that system requirements are essential to the development of good software requirements, or the unwillingness to act in accordance with that knowledge.
- The desire of senior management in many large corporations to place personnel with little software knowledge or experience in positions of responsibility over what is essentially a major software project.

It is obvious that these are the common problems in requirements engineering but only knowing these problems will not solve all the problems in the projects. Software engineers and senior management should be proactive early in the project with the help of old project experience.

2.1.3 Requirements Elicitation and Techniques

Requirements elicitation defined as the process of identifying needs and bridging the disparities among the involved communities for the purpose of defining and distilling requirements to meet the constraints of these communities [SEI, 91]. Gaining a better understanding of user needs moves us from the domain of bits and bytes, where many developers are more comfortable, into the domain of real people and real-world problems [Leffingwell&Widrig, 2000].

Answer of "why requirements elicitation is difficult?" question comes with the following problems [Christel&Kang, 92]:

- **Problems of scope.** The boundary of the system is ill-defined or the customers/users specify unnecessary technical detail may confuse, rather than clarify, overall system objectives.
- **Problems of understanding.** The customers/users are not completely sure of what is needed; have a poor understanding of the capabilities and limitations of their computing environment; don't have a full understanding of the problem domain; have trouble communicating needs to the system engineer; omit information that is believed to be obvious; specify requirements that conflict with the needs of the other customers/users; or specify requirements that are ambiguous or unstable.
- **Problems of volatility.** The requirements change over time.

Just a variety of techniques can be used for analyzing and designing software solutions, a variety of techniques can be used to understand user and stakeholder requirements. Some techniques are better suited to particular project teams and circumstances. There are many techniques in the literature; some of them are listed as follows [Pressman, 05], [Thayer&Dorfman, 97], [Leffingwell&Widrig, 2000]:

- Interviewing and questionnaires
- Requirements workshops
- Brainstorming and idea reduction
- Storyboards
- Scenarios, use cases and goal-based approaches
- Role playing
- Prototyping

Interviewing is the process of analysts asking questions and making customers to describe their thoughts, opinions, concerns and needs [Hickey&Davis&Kaiser, 03].

Questionnaires are composed of closed and open ended questions. They are distributed to all customers. When responses returned, they are analyzed to understand general trends and customers' opinions.

Workshops provide integrity between the customers and the development team. A series of planned activities are conducted to produce the requirements deliverables needed. *Brainstorming* is the process of gathering customers in a meeting, asking some questions and encouraging customers to express their ideas about those issues.

Storyboarding is done by drawing a set of drawings for depicting user activities expected from produced system. This technique is similar to prototyping technique, but it is only done on a paper. Customers and development team both contribute these drawings till real requirements agreed. *Use Cases* are drawings of system actors' activities. They will also contain the textual descriptions for writing requirements in a common language both customers and development team can understand.

Role playing is experiencing user's world from users' perspective. Based on a scenario users' activities are analyzed for writing good requirements. *Prototyping* is partially implementation of the system. After prototype production, it is given to the customers to make a comparison between their expectations and the prototype.

Every method has some limitations. Advantages and disadvantages of these methods will be explained. Despite their limitations, using one of the methods only is not suggested [Goguen&Linde, 93]. Combination of the various methods can be usefully applied to particular problems.

2.2 Method Engineering

"There is no detailed information systems methodology which is best in all situations" was stated by Kumar and Welke. A solution is introduced to this problem called method(ology) engineering [Kumar&Welke, 92]. This methodology supports the construction of methodologies for information systems development from fragments or chunks. The relationship between information system development (ISD) and the method engineering is shown in Figure 2.1. A *chunk* is encapsulated both as a process part and a product part [Henderson, 03]. A *fragment* is one or the other. Fragments and chunks are stored in a repository, called a *methodbase*. They are coming from industry best practice and compliant with a given metamodel.

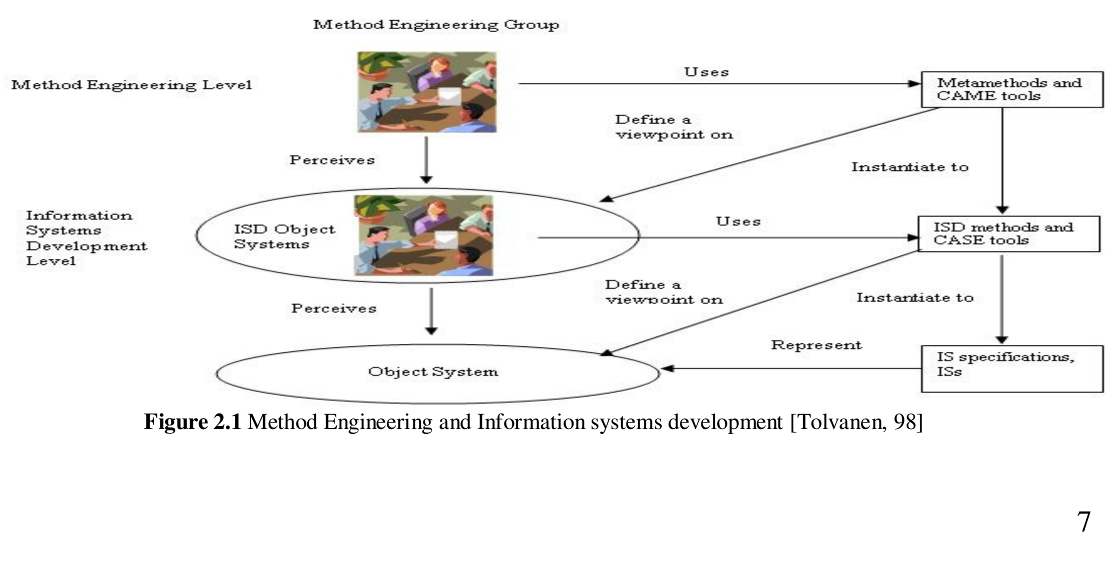


Figure 2.1 Method Engineering and Information systems development [Tolvanen, 98]

Method engineering provides construction of a methodology which is fitting an organization's requirements. It is different than Software Process Improvement but their aim is providing more effective and effective information systems and software engineering practices [Agerfalk&Ralyte, 06]. The focus of method engineering is; supporting incremental, model-based definition of project-specific methods for information systems and software process guidance and incorporation of experience to facilitate improved guidance for future efforts. Moreover, Brinkkemper (1996) defined method engineering as design, construct and adapt methods, techniques and tools for the development of information systems.

Colette Rolland (2007), summarizes method engineering artifacts as method engineering products and method engineering processes. Method engineering products are method engineering meta-models, meta-modeling languages to represent method parts [Rolland, 07]. Moreover, method engineering processes are method engineering strategies, approaches, workflows to combine, integrate, and assemble method parts and guiding method engineering process as shown in Figure 2.2.

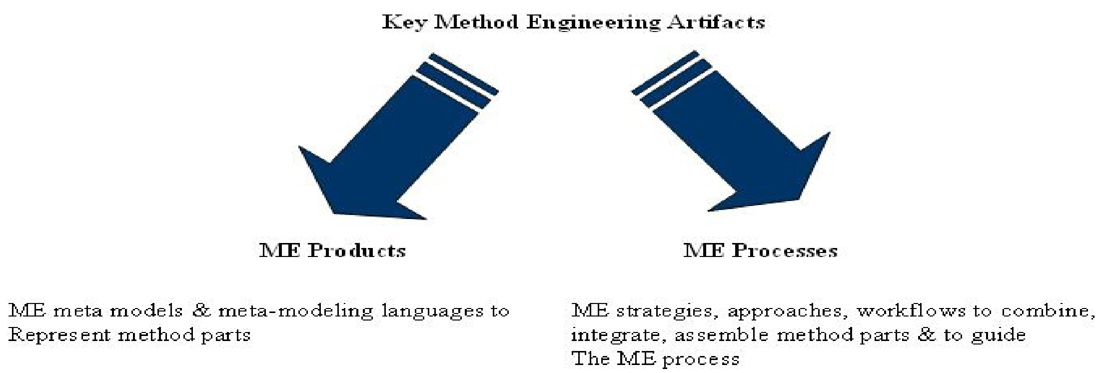


Figure 2.2 Method Engineering Artifacts [Rolland, 07]

2.2.1 Method Engineering Products and Processes

Generally, a product model is explicit description of the abstract systems, spaces and physical elements of a physical facility. Physical model element attributes include dimensions and materials. A process model is explicit description of the tasks of a project with their precedence and information and rework dependencies. Task attributes include work volume, task complexity and required skills. A method is composed of a product model and a process model as shown in Figure 2.3.

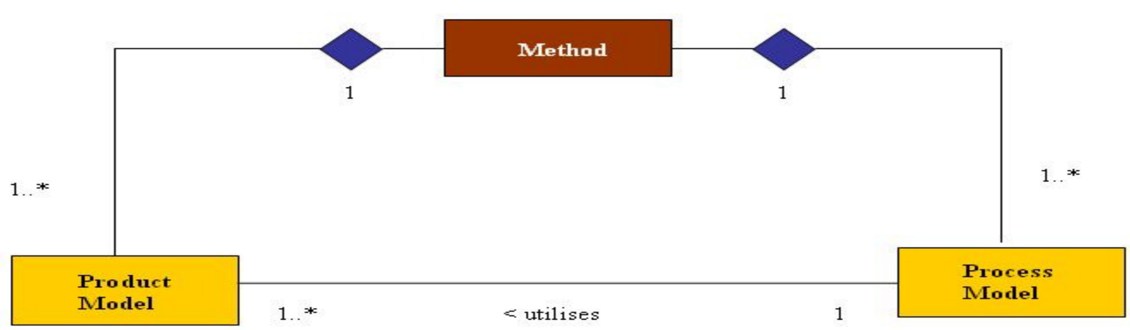


Figure 2.3 Method Parts [Rolland, 07]

Meta-modeling is the appropriate way of describing methods. Saeki (2003) describes methods meta-model of product and process part as shown in Figure 2.4.

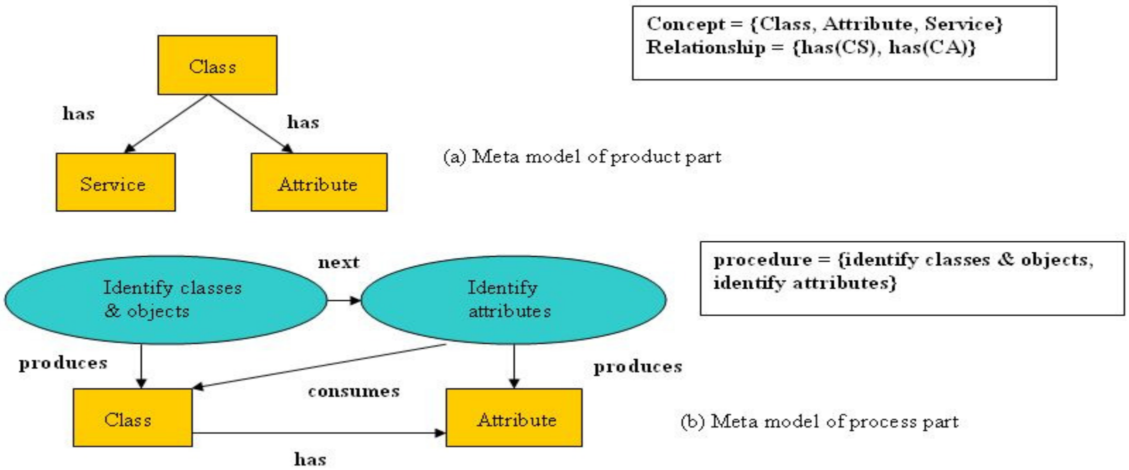


Figure 2.4 Meta-Modeling [Rolland, 07]

The method engineering process has three steps. They are method selection, method construction and tool adaptation. These steps are shown in Figure 2.5.

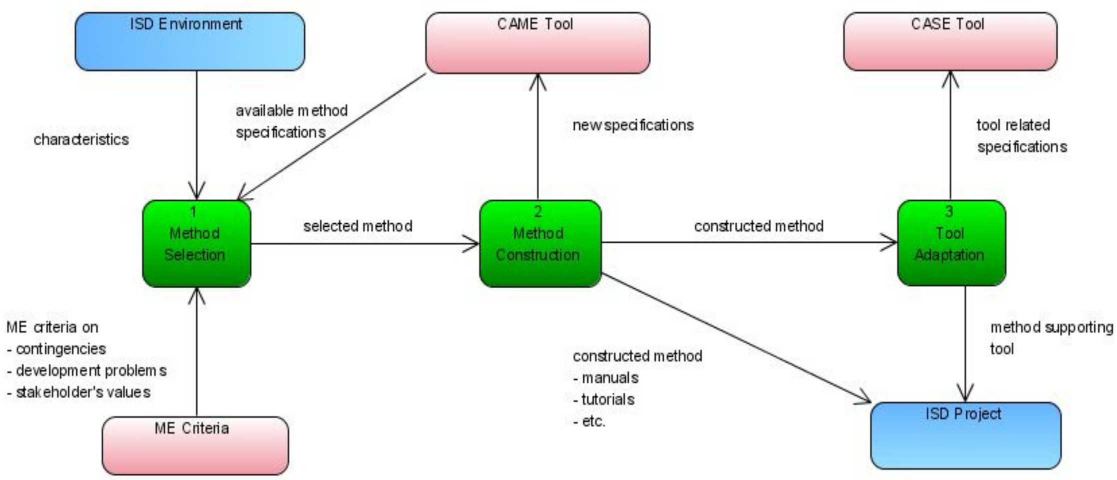


Figure 2.5 Method Engineering and Information Systems Development

The first step which is called method selection process, information systems development environment is analyzed according to method engineering criteria. They include classifications of relevant aspects of methods which should be considered to satisfy the objectives for the method. Contingencies, development problems and stakeholder's values come into prominence as criteria. The method construction with new method components for the selected method or methods which meet method engineering criteria is completed in second step. This provides combining method knowledge from different methods at the same time from different types of method knowledge. Dividing method into method components or fragments provide catching smaller changes even the methods integrated. All criteria can not be matched with existing methods, new method knowledge needs to be defined and some of the method components may need to be removed. The new method configuration is stored into a repository for future selections.

The last step after method construction, this method needs to be adapted into a CASE tool. Tool adaptation deals with customizing tool for the method or choosing a set of tools which cover all the method knowledge. Method construction is limited if this step is not carried out because a tool can ensure that the method is used as intended. An information systems specification without CASE support is similar to method engineering without tool adaptation. There are also some activities which they do not need any adapted tool such as production of manuals, tutorials etc. If this part completed successfully, it provides what developers stored in the repository how system descriptions be represented, retrieved, checked, transformed and how descriptions are managed. Moreover, method development life cycle and possible outputs are shown in Figure 2.6.

Stage	Process	Input	Output
Requirements Engineering	Intention Matching	Goal of the method To-Be	Intentionally similar methods
Design Engineering	Architecture Matching	Architectures of intentionally similar methods	Architecturally similar method
Construction Engineering	Organization Matching	Workflows of architecturally similar methods	Method To-Be

Figure 2.6 Method Development Life Cycle [Rolland, 07]

As mentioned before, methods that are constructed should be organization-specific or project specific to get more benefit from method construction. This is the concern of situational method engineering. Situational method engineering encompasses all aspects of creating a development method for a specific situation. It focuses the formalization of methods in terms of reusable method components and the definition of assembly techniques allowing to construct new methods by reusing these components. Next section will describe situational method engineering in detail.

2.3 Situational Method Engineering (SME)

2.3.1 Definition

Generally situational method engineering is known as quality of products, and it motivates the development of solutions to adapt methods to the project situation at hand [Ralyte&Deneckere&Rolland, 03].

The particular circumstances of each project make it necessary to adapt the methods to deal with the situation at hand. This is the concern of situational method engineering, where the term situational method is used to refer to a method tailored to the needs of a particular development setting. Situational method engineering prescribes the performance of this method customization within the framework of a meta-modeling technique provided with mechanisms to manipulate methods (or fragments of them) for their modification, integration, adaptation or evolution [Dominguez&Zapata, 05].

2.3.2 Situational Method Approaches

2.3.2.1 Generic Process Model

SME process is originated from two main activities. First one is setting the method engineering goal; second one is constructing a method that matches this goal. These are shown in Figure 2.7.

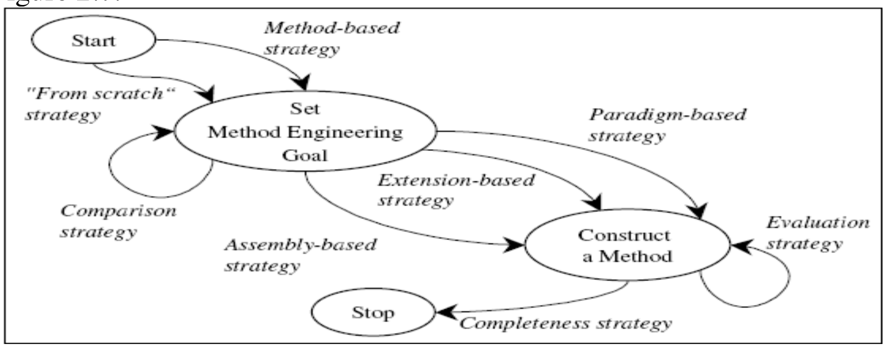


Figure 2.7 The Generic Process Model [Ralyte&Deneckere&Rolland, 03]

The achievement of the first node, namely “Set Method Engineering Goal”, depends on the method situation of the project at hand. In one project situation the method engineer may perhaps consider that a specific method could be applicable but requires some adaptations whereas in other situations, he or she may be convinced that any of the available methods is suitable for the project. In the first case, the method engineering (ME) goal refers to the adjustments of the selected method: enhancement, extension or restriction. The corresponding strategy is called the “Methodbased strategy”. In the second case, it is necessary to construct a completely new method and the corresponding strategy is called the “From scratch strategy”. The achievement of the node “Construct a Method” depends on the applied method construction technique [Ralyte&Deneckere&Rolland, 03].

2.3.2.2 Assembly-Based Method Engineering

Brinkkemper (1996) recognized the following steps for Assembly-Based Method Engineering: (1) characterization of the project, (2) selection of method fragments (that are stored in a method base), and (3) assembly of method fragments. The experience gained in this process is new input for the method base. Saeki (2003) states that the simplest way to construct a new method is first to put meaningful method fragments in a method base, then to select useful method fragments from this method base, and finally adapt and integrate them in a new method. Ralyte, Deneckere and Rolland (2003) have developed the assembly-based process model for situational method engineering as shown in the Figure 2.8. This model describes three steps to develop a new situational method. The steps are: (1) specify method requirements, (2) select method fragments and (3) assemble method fragments [Weer&Souer&Versendaal&Brinkkemper, 05]. Detailed explanation of the approach is that the method engineer must start by eliciting requirements for the method. Next, the method chunks matching these requirements can be retrieved from the method base. And finally, the selected chunks are assembled in order to compose a new method or to enhance an existing one.

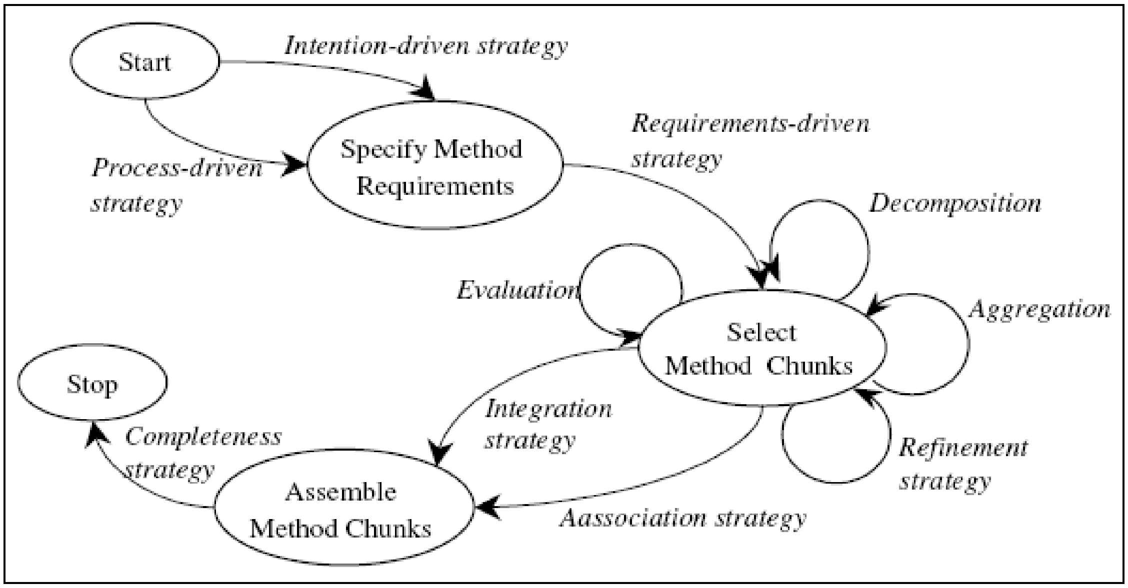


Figure 2.8 The Assembly-Based Process Model [Ralyte&Deneckere&Rolland, 03]

In this study, assembly-based method engineering is used when developing tool of the proposed approach. It begins with selection of two methods and then assembling of methods starting from specifying of method requirements that ends with selection of method steps.

2.3.2.3 Extension-Based Method Engineering

The approach guides method engineer by providing extension patterns that help identifying typical extension situations and provide advices to perform the required extension. Figure 2.9 shows the representation of the extension-based approach [Ralyte&Deneckere&Rolland, 03].

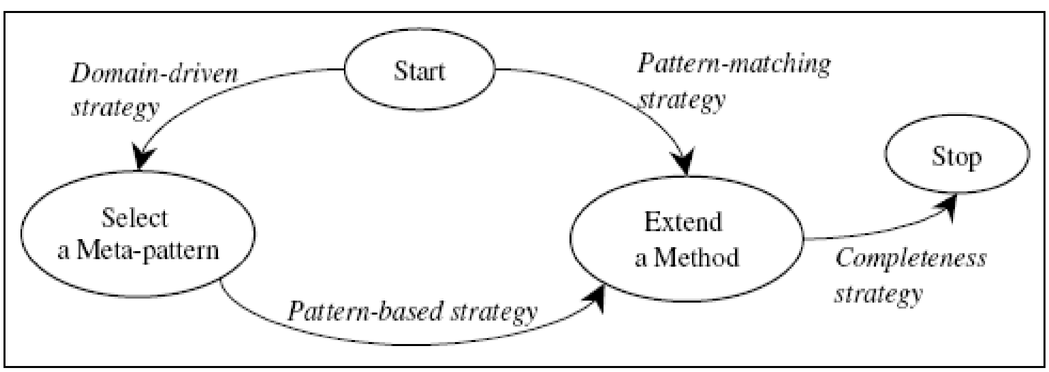


Figure 2.9 The Extension-Based Process Model [Ralyte&Deneckere&Rolland, 03]

It can be seen that approach has two different ways to extend a method: (a) directly through the "Pattern-matching strategy" or (b) by using some generic knowledge related to the domain for which the extension is to be done through the path "Select a Meta-pattern", "Extend a Method" with the "Pattern-based strategy". The former helps to match extension patterns stored in a library to the extension requirements whereas the latter selects first, a meta-pattern corresponding to the extension domain and then, guides the method extension by applying the patterns suggested by the meta-pattern. In this study, extension-based method engineering approach is used when developing tool of the proposed approach. It begins with selection of method and then extending of method requirements and steps.

2.3.2.4 Evolution-Driven Method Engineering

This engineering methods use meta-modeling. Meta-modeling is capturing technique for knowledge. It provides basis for methods in terms of understanding, comparing, evaluating and engineering. As mentioned before, from meta-modeling two terms are coming from which are process model and product model. A product model defines a set of concepts, their properties and their relationships that are needed to express the outcome of a process. A process model is formed by a set of goals, activities and guidelines to support the process goal achievement and action execution so method construction in this meta-modeling technique [Ralyte&Rolland&Ayed, 05], [Ralyte&Deneckere&Rolland, 03] is centered on the definition of these two nodes as shown in Figure 2.10.

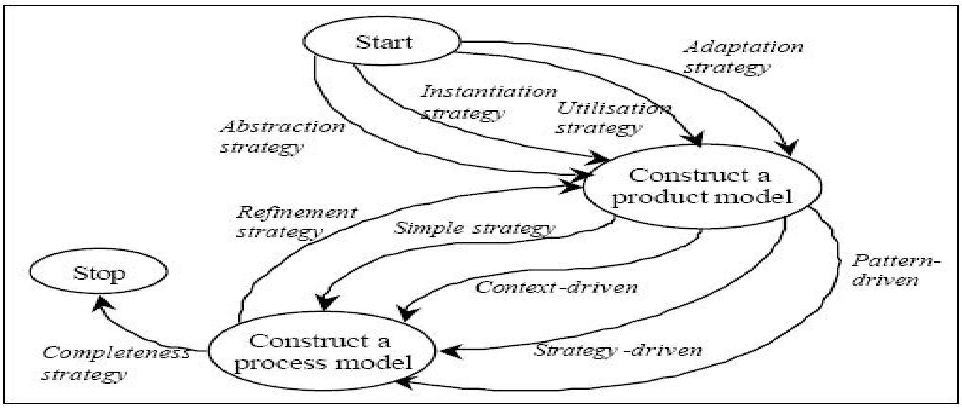


Figure 2.10 The Evolution-Driven Process Model [Ralyte&Deneckere&Rolland, 03]

The construction of the product model depends on:

1) Raising or lowering the level of abstraction of a given model
2) Instantiating a selected meta-model
3) Adapting a meta-model to some specific circumstances
4) Adapting a model

These are the strategies for constructing a product model. They are called; abstraction, instantiation, adaptation and utilization strategies. Process model must conform to the product model. Ralyte, Rolland, Ayed are exampling their approach by an abstraction strategy. This is abstraction of different elements from the paradigm model into elements in new product model and the proposing the obtained elements until the new product model is satisfactory. Paradigm means product and or process model.

Evolution-driven process model has four strategies: simple, context-driven, pattern-driven and strategy-driven for constructing a process model.

- **Simple Strategy:** This strategy is useful for describing a simple process model which can be expressed as a textual description or a set of actions.
- **Context-Driven Strategy:** This strategy is leaning on nature process modeling. According to this modeling technique, a process model can be expressed as a hierarchy of contexts. This context consist of <situation, intention> pair. The part of the product undergoing the process called situation and reflects the goal to be achieved in this situation called intention.

- **Pattern-Driven Strategy:** This strategy takes the form of a catalogue of patterns. Each pattern identifies a generic problem which is possibly occur in the product construction and suggests a solution generically applicable every time the problem appears. This solution is expresses as set of steps allowing to resolve the corresponding problem.
- **Strategy-Driven Process Model:** This is also called the Map. It permits to combine several process models into one complex process model.

2.3.2.4.1 Abstraction-Based Product Model Construction

Abstraction strategy is defining a new product model representing the level of abstraction higher than its paradigm model. This type of construction supports the construction of a product model as an abstraction of other model as shown in Figure 2.11.

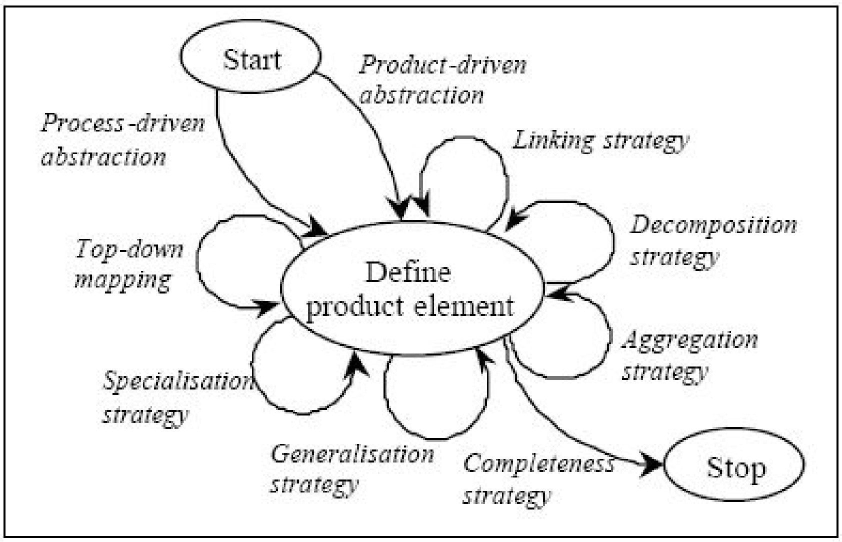


Figure 2.11 Abstraction-Based Product Model Construction [Ralyte&Deneckere&Rolland, 03]

Abstraction-based product model's main activity is defining product element. This activity is supported by product-driven abstraction and process-driven abstraction which are starting the construction process. The product-driven abstraction consists of analyzing the paradigm product model, identifying elements which can be represented by more abstract elements in the new model and defining these abstract elements. The process-driven abstraction purposes analyzing the paradigm process model and abstraction some of its activities into the upper level ones. This construction model has to match with the concepts obtained from this technique. The top-down mapping strategy can be applied for checking it. Generalization, specialization, aggregation and decomposition strategies are used for purifying the model under construction. On the other hand, linking strategy helps to connect different elements of this model.

2.3.2.4.2 Pattern-Based Process Model Construction

The pattern-based process model construction strategy is based on the pattern concept. A pattern is defined as a problem which occurs again and again in the environment and describes the core of the solution to that problem in a way that this solution can be used million times over without making it the same way twice so pattern's central idea is associating problem to its solution in a well identified context. This model consists of two main nodes which are called identify a pattern and construct a pattern as shown in Figure 2.12.

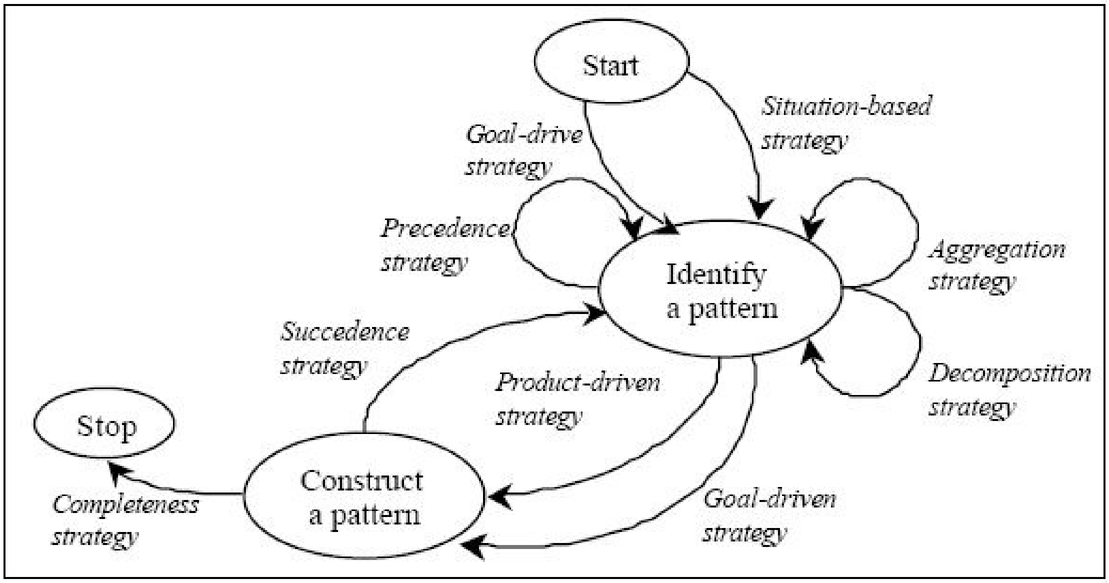


Figure 2.12 Pattern-Based Process Model Construction [Ralyte&Deneckere&Rolland, 03]

Pattern identification is based on the discovery of a typical situation or a generic problem in the method context. These cases are supported by goal-driven strategy and situation-based strategy. Aggregation strategy provides combining several patterns into a compound one to propose solutions for complex problems. On the other hand, decomposition strategy is dealing with the identification of sub-problems that can be considered as generic problems. Precedence strategy supports the identification of a new pattern situation advices us to consider that there must be another pattern creating this situation.

Formalizing the situation and the goal, to define the solution to problem or a set of steps to execute, to define template and give some examples to application represented by constructing a pattern node shown in Figure 2.12. Goal-driven and product-driven strategies support this activity. Product-driven strategy is based on the transformation of the product elements from the pattern situation into the product element. Goal-driven strategy makes the pattern goal reduction into a set of atomic actions to realize to achieve this goal. Succedence strategy computes the results produced by the applied pattern.

2.3.2.5 Combination of Assembly-based and Roadmap-driven Approaches

Mirbel and Ralyte (2005) propose a new approach for situational method engineering called combination of assembly-based and roadmap-driven approaches. To get more benefit and using situational method engineering more efficiently, this approach is formed. Firs step is this approach is providing support to build a new method which is most suitable for the current information systems development project situation. On the other hand, the second step is trying to choose the most adapted path called roadmap to satisfy the requirements of a particular project engineer within the project specific method as shown in Figure 2.13.

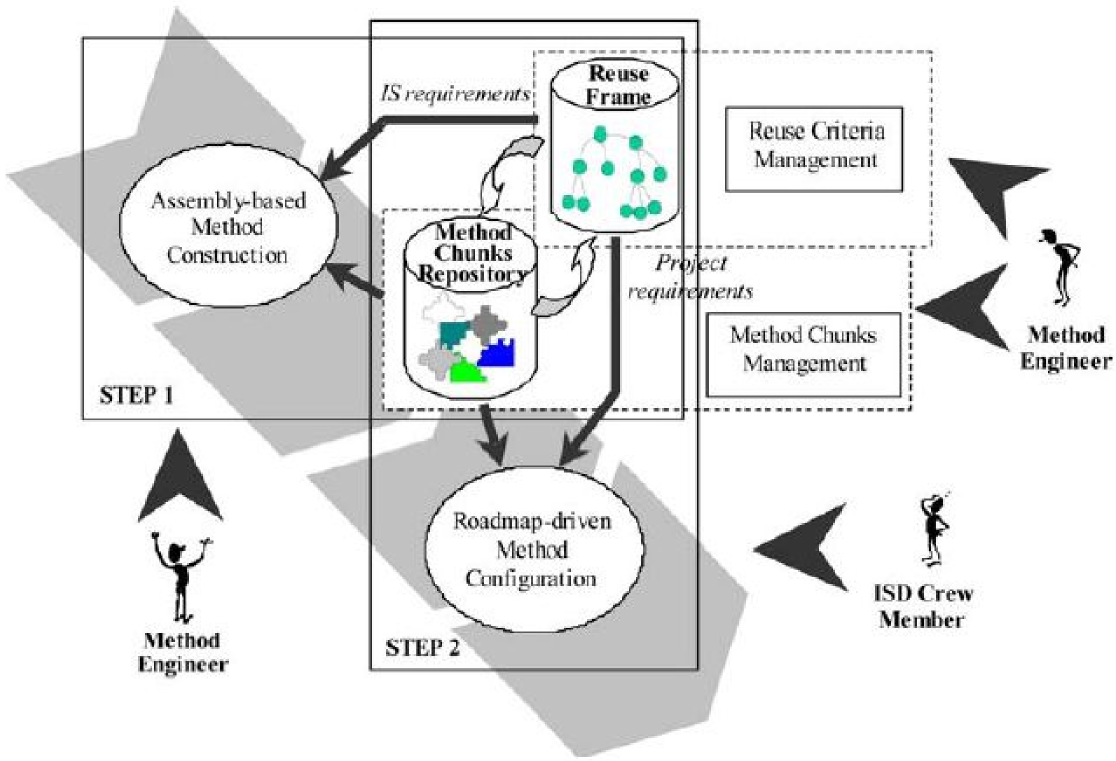


Figure 2.13 Situational Method Engineering combining assembly-based and roadmap-driven approaches[Mirbel&Ralyte, 05]

Assembly-based method construction aim is building instant methods to match with project at hand [Mirbel&Ralyte, 05]. This method consists of selecting reusable method components (method chunks) according to the current project requirements from the method chunks repository and assembling them. The main phases of method engineering are the same in assembly-based method engineering. Method engineer must start with eliciting requirements for the project method. After that, the method chunks matching with these requirements should be retrieved from the method chunks repository. Selected chunks are assembled to compose a new method or enhancing an existing method in the last phase.

Roadmap-driven method configuration's main theme is customizing the project-specific method instantly to match with the possible profile of the information systems development crew member's job within the project at hand. This method includes the selection of method chunks satisfying the information systems development crew member's needs within the frame of the project specific method previously was built for answering project requirements. This approach's is based on three main steps. First one is the specification of the methodological problem of the information systems development crew member. Second one is the retrieval of the method chunks matching with the problem from the project specific method that was built through first step of the approach and from the method chunks repository. Third one is the completion of the solution by additional method chunks are retrieved from the repository. Although both approaches are defined independently, integration between them with method chunk sharing. This chunk which is also called repository is offering basis for each of the approaches.

2.4 Relationship between Situational Method Engineering and Requirements Engineering

Method engineering and software process improvement are different approaches but they are related approaches by aiming more effective and efficient information systems and software engineering activities. Main duty of method engineering is supporting the proper software process guidance and incorporation of experience for facilitating improved guidance for the future efforts [Agerfalk&Ralyte, 06]. As mentioned before, requirement engineering is multi disciplinary activity that a variety of techniques and tools are used through the development in many application areas. Method engineering is important for requirement engineering process to be successful.

In the literature there are many articles that show the relationship between situational method engineering and requirements engineering. Chad Coulin, Didar Zowghi and Abd-El-Kader Sahraoui represents how method engineering principles can be used to create a lightweight situational approach to early phases of requirements elicitation. Johan Hoorn, Mark Breuker and Evelien Kok brings attention to requirement analysis rift. This rift suggests that stakeholders are directed to accept requirements even when they do not match their personal goals. This also shows the key to successful requirements engineering is grounding requirements in goals and paying attention to different viewpoints on requirements, goals and their relationships. Regine Laleau, Sylvie Vignes, Yves Ledru, Michel Lemoine, Didier Bert, Veronique Donzeau-Gouge, Catherine Dubois and Fabien Peureux show the usage of situational method engineering principles to engineer a requirements engineering method suited for this timely and in many ways and critical domain. Toshihiko Tsumaki and Tetsuo Tamai presents practical framework for characterizing different requirements engineering techniques as a basis for selecting those appropriate in a particular situation. This framework acknowledges the changes to an ongoing project as well as difficulties experienced in gathering requirements must reflect on the selection requirements engineering techniques. Inge van de Weerd, Sjaak Brinkkemper, Jurriaan Souer and Johan Versendaal are setting out to improve the development of web-based content management systems through method engineering. This article's study presents a novel approach to developing web-based systems and evaluates its usefulness in two case studies. Finally, Mauri Leppanen presents a overview of current work in method engineering and assesses the usefulness of the various approaches published. This conceptual evaluation of methods for engineering situational information systems development methods will provide much useful advice for anyone wanting to get result of method engineering as a way of improving their requirements engineering and software processes.

CHAPTER 3

3. SITUATIONAL REQUIREMENT METHOD SYSTEM

3.1 Overview

The importance of knowledge management is immense in software development industry. Software industry is knowledge intensive and requires extensive management of this knowledge [Chandani& Neeraja& Sreedevi, 07]. Knowledge management ensures that there is an effective sharing and exploitation of accumulated, collective knowledge. Every company in software development should set software process activities for the processes it adopts. According to each project and the needs of the company, they need different set of activities [Pressman, 05]. As mentioned before, method engineering (ME) is a discipline to study engineering techniques for constructing, assessing, evaluating, managing methods and to study educational techniques for teaching and training method users [Saeki, 03].

Methods are normally general in nature and they can not be used directly without adapting them according to the characteristics of the project. Because the engineering situation of each information system development (ISD) project is different, engineering methods need to be adapted, transformed or enhanced to satisfy the specific project situation [Mirbel&Ralyte, 05]. In addition to method tailoring, it is necessary to fit the project situation, a customization of the engineering method for each engineer participating in the project is also required [Mirbel&Ralyte, 05]. This is the concern of situational method engineering, where the term situational method is used to refer to a method tailored to the needs of a particular development setting.

Also, once a method is constructed according to the project situation and later applied successfully, it should be available for the future use in some repository so that others can learn from past experiences. It is important for the software engineers to learn from their own past experience as well as experience of others. Therefore, there is a need to enhance the cooperation among software developers and software development organizations must have a mechanism to support the sharing of knowledge held by different software developers to remain competitive.

An approach to enhance cooperation and knowledge sharing among software developers is proposed with the help of a web-based tool. Methods that are used successfully in past projects, along with project characteristics, can be stored and shared with the help of this tool. It uses Assembly-Based and Extension-Based Method Engineering Approaches as mentioned in Section 2. It also provides support for the reuse of existing methods to create new methods. This approach is applied for requirement phase in this work but it can be extended for any phase of software development.

There are variety of techniques that can be used to understand user and stakeholder requirements. Because of the relative strengths and weaknesses of the available approaches, most projects require a combination of several techniques in order to produce quality results [Maiden&Ruggy, 96].

3.2 Requirement Engineering Techniques' Advantages, Disadvantages and Limitations

Requirement elicitation is the important part of the requirement engineering phase. Although many methodologies exist, the common aim with them is to assist analysts in understanding needs but the common belief is one technique is applicable to all situations. It is obvious that one technique can't possibly sufficient for all conditions [Hickey&Davis, 03] so number of techniques is available with advantages and disadvantages. Table 2 shows these advantages, disadvantages and limitations of these techniques. This process seems easy but it actually is not easy as it seems. Traditional techniques are; introspection, questionnaires, interviews, workshops, brainstorming, storyboarding, use cases, role playing, and prototyping.

Introspection technique is very useful but it has the problem that introspection of an expert in a different field, such as Requirement Engineering is unlikely to reflect the experience of actual users [Goguen&Linde, 93]. Experts tend to work from what they remember or imagine of themselves; for user design, this experience can be very far from the questions, assumptions and fears of actual users. Requirements engineers can not introspect what work settings look like, or the conditions under which a new technology will be learned. Limitations of the technique should be known before using this technique. Introspection should be checked by some of the more empirical methods.

Interviewing is a simple direct technique. The use of context free questions by the interviewer helps avoid prejudicing the response. Context free question is a question that does not suggest a particular response. It is good for gathering undiscovered requirements.

Questionnaires are very widely used and have the benefit of appearing scientific because they use statistical analysis but the technique has also disadvantages such as: relevant questions can not be decided in advance, the assumptions behind the questions bias the answers, it is difficult to explore new domains and there is no interaction to explore domains that need to be explored, unclear responses from the user are difficult to follow up on [Leffingwell&Widrig, 2000]. The problem of this approach is that in some situations which are especially exceptional or error behavior are so infrequent that they can not be formulated because they occur often only after the deployment of the system.
[Bruegge&Herrmann&Rauschmayer&Renner, N/A].

According to Leffingwell and Widrig, requirement workshop is designed to encourage consensus on the requirements of the application and to gain rapid agreement on a course of action, all in a very short time frame. Advantage of this technique is, key stakeholders of the project are gathered together for a short, intensive period, typically no more than 1 or 2 days. Main advantages of this technique are: costs are lower than interviews, gives a structure to the capture and analysis of the requirements process, dynamic, interactive, cooperative, involves users and cuts across organization boundaries, helps to identify and prioritize needs and resolve contentious issues, helps to manage user's expectations and attitude towards change.

Table 2 Requirement Elicitation Techniques' Advantages and Disadvantages

Technique	*Advantages*	*Disadvantages*	*Limitations*
Introspection	Very useful for introspecting actual user ideas	Experts in that or another field tend to canalize actual users into their idea	Should be checked by some empirical methods
Questionnaire	Uses statistical analysis	Irrelevant questions, and unclear responses	In some situations which are especially exceptional or error behavior are so infrequent that they can not be formulated because they occur often only after the deployment of the system
Interview	Directly communicate with the customer Simple Technique	Inexperienced customer causes irrelevant requirements	Time is needed to interview with the customer, Context free question
Workshop	Costs are lower than interviews, gives a structure to the capture and analysis of the requirements process, dynamic, interactive, cooperative, involves users and cuts across organization boundaries, helps to identify and prioritize needs and resolve contentious issues, helps to manage user's expectations and attitude towards change	Inexperienced customer causes irrelevant requirements	Negative comments, time management and customers' experience
Brainstorming	The most creative or effective ideas often result from combining seemingly unrelated ideas, involves both idea generation and idea reduction	Lack of customer input causes less idea in the meeting	Criticism, limited imagination
Storyboarding	Extremely inexpensive, user friendly, informal, interactive, provides an early review of the user interfaces of the system, is easy to create and easy to modify. It eliminates the risks and higher costs of prototyping	It continues till requirement agreement	Time that extending the agreement on the requirements
Use Cases	Expresses the sequence of events and a common language for the end users and the technical team	It is not appropriate for the applications which has not got much user interaction	Needs time to draw uses cases, can not be used for every system
Role Playing	Cheap and usually quite quick, for addressing particular causes it is quite effective	Results will not be as expected (method only applied)	Role playing is not suitable for every eng. in team
Prototyping	Touching, feeling and interaction with the system before development has been completed	Quite risky when the produced prototype is not matched with the user expectations	Time is needed for producing a prototype and for getting feedbacks

Disadvantage of this technique is meeting oriented. Negative comments, time management and customers' experience has direct influence on this technique.

Brainstorming is a collection of techniques that are useful when stakeholders are collected. The advantages of this technique are; it encourages participation by all parties present, it allows participants to piggyback on one another's ideas, a facilitator or scribe maintains a written trail of everything discussed, it exhibits high bandwidth, it results in a broad set of possible solutions to whatever problem is posed, it encourages out of box thinking, without being limited by normal constraints. When criticism occurs, this is the disadvantage of this technique.

With storyboarding, the user's reaction can be observed very early in the lifecycle, well before detailed specifications are developed. Storyboarding is extremely inexpensive, user friendly, informal, interactive, provides an early review of the user interfaces of the system, is easy to create and easy to modify. It eliminates the risks and higher costs of prototyping.

Use cases can be used for requirement elicitation to understand the needed behavior of the application. Use cases serve as the UML representation for the requirements of a system. Use cases developed in the elicitation process will be of even further value during analysis and design activities. This method is useful because the uses cases can also play an important role in the testing process. It is more appropriate for applications which have more user interaction. They provide a context for the requirements by expressing the sequence of events and a common language for the end users and the technical team.

Another technique, role playing allows the development team to experience the user's world directly by playing the role of the user. For addressing particular causes, the role playing can be very effective. It is also cheap and usually quite quick. An hour or a half day can be enough for role playing.

Software requirements prototyping is important because it is a partial implementation of a system, built to help developers, users and customers better understand system requirements. Users, customers of the system; can touch, feel and interact with a prototype system in a way that none of the other techniques can provide but this technique is quite risky when the produced prototype is not matched with the user expectations. There will be time problem in that case to complete the requirements and finally project.

Different requirement elicitation techniques can be used for software development projects. Every technique has some advantages and disadvantages. None of the technique is perfect in every circumstance. The most important thing is using the most appropriate technique for appropriate project. In order to do this, situational requirements engineering should be considered because situational requirements engineering combines requirements with their context.

“Clearly requirement elicitation does not occur in a vacuum and is highly dependent on some characteristics” [Christel&Kang, 92]. As mentioned before, every method has advantages and disadvantages. Different requirement elicitation techniques and suitability of these techniques in different project situations are studied. With this knowledge, criterion methodology is formed that classifies different requirement elicitation techniques. The criteria can be used during method creation as well as during method selection. To distinguish methods from each other according to situation at hand, project characteristics in which the method was applied successfully should also be stored with the methods. Before that these characteristics should be identified. The combination of characteristics can be also called as scenario that had been experienced before. This also helps the method distinction so the users of the system separate their methods’ according to the scenario at their hand. These characteristics affects the software engineering process in every phase but the focus of this study is requirement elicitation phase because the two biggest problems in software engineering are the process of efficiently, effectively developing requirements and the tools required for creating truly agile solutions that can change as quickly as your clients mind. IEEE [IEEE, 90] defines requirements engineering as the process of studying user needs to arrive at a definition of whole system requirements. In the next section, criteria for classification of requirement elicitation methods are given. The justification of selecting these criteria is also described.

3.3 Criteria for classification of methods

- Experience of the requirement engineer: Experience of the engineer who is responsible for method entrance affects the method success and reliability because requirement engineer sometimes is taking decisions about requirement elicitation process. When engineer faced with a new requirement elicitation situation, analysts select the technique by these approaches: select an elicitation technique because it is the only one that they know, select their favorite elicitation technique and they select an elicitation technique because they understand intuitively that the technique will be effective in the circumstance [Hickey&Davis&Kaiser, 03]. So, if the engineer has not got enough experience, he/she can take wrong decisions. In order to prevent this, experience of the engineer should be considered. For example: if the experience of the engineer is more than 5 years, introspection method can be used for requirement elicitation.
- Experience level of the engineer in similar projects: Experience level of the engineer in similar projects also affects the method because similar project experience provides statistical data to the engineer. By using this statistical data from similar projects, engineer can elicit requirements with the most effective method so similar project experience and collecting statistical data is important for requirement elicitation. It is also important for all phases of the software development. For example: if the similar project experience of the engineer is more than 5 years, using questionnaire technique for requirement elicitation can be useful. Since the technique is beneficial for similar projects.
- Requirement elicitation period: Requirement elicitation period in software engineering process is important because if the period is less or more, the method to be used should be different. Some of the techniques are effective to use in short time period. In order to elicit requirements in that period for example if the company have an hour

to elicit requirement, role playing can be used for requirement elicitation. For example: if the customer can attend the meetings arranged by the company, workshop technique can be used for requirement elicitation.

- Experience level of the company's customer: This will also affect the method to be selected for the use. If the customer does not have enough knowledge about softwares and software environment, requirements can not be easily elicited from the customer. For example: introspection technique is not effective for less experienced customer if the formed requirements from the engineer are entirely different from the customer's. Prototyping is effective for requirement elicitation if the customer is experienced. It is effective because customer can give valuable feedback to the engineer so the requirements will be more mature and healthy. Workshops are also effective for requirement elicitation when the customer is experienced because the ideas given by the customer is more valuable.
- Possibility of meeting between development team and customer: If the customer and development team can make meetings together, it affects the method to be used in the requirement elicitation. For taking a common decision in requirement elicitation making meetings together is important. Common decision and evaluating lots of ideas is more effective for requirement elicitation because it can lead to mature and stable requirements in the early phase of the development. It will not cause lots of requirement changes. Brainstorming and workshop techniques are effective for this type of situations.
- Project budget: Project budget is important for project development since the business to be transacted is limited with the project budget. It has also effect on the tool and the technique to be used for requirement elicitation. If the project budget is less, role playing can be used for requirement elicitation. It is cheap and also quick. Moreover, storyboarding can be used for requirement elicitation because it is inexpensive and also user friendly, easy.
- Project's user interaction level: If the user interaction in the system is less or more, the technique for requirement elicitation should be different. Requirements of user interfaces should be clear early in the project. For example: if the user interfaces in the project is more uses cases technique can be used for requirement elicitation. It is effective for this type of systems and also useful for testing.
- Project complexity: Not all of the projects conducted from the company are at the same level. Some systems are complex to elicit requirements in one phase. For these type of systems, different techniques should be used for requirement elicitation. Prototyping technique can be used for complex type systems since it is a partial implementation of a system, helps developers, users and customers better understand system requirements with the help of this technique.

3.4 SRMS Tool

3.4.1 Tool Overview

There is a need to use a tool to support the proposed approach. Therefore a tool, Situational Requirement Method System (SRMS), is developed to demonstrate the proposed approach. SRMS is developed with PHP. MySQL database is used to store the method related information. Moreover, for the graphical part JpGraph [JpGraph, 09] which is an object oriented graph creating library is used. SRMS tool is a web-based tool because the project members can be in different places so tool can be easily accessed by the members of the project.

Most of the companies store their experiences in lessons learned databases. It is an obligation to store these databases for process improvement. Process improvement is important for capability maturity model (CMM). Lessons learned database is a structured database that everyone in the project can add their comments, experiences. Lessons learned databases are also useful for the project managers who are working in an industry where the projects are similar. It will save time in the next project which is similar to previous project. SRMS also provides this ability to users. If they are effectively used, both of them ensure the same mistakes are not repeated, in case of budget overruns and customer dissatisfaction. Knowledge is very valuable thing in software industry. People who are working in software industry are frequently changing their jobs, so if the knowledge obtained by that people were not recorded it will be lost. Lessons learned meetings should be done every week to hold subject alive. SRMS also provides these features to the users of the system, sharing their ideas from comment module about methods created in the tool. Since only author of the method can update the method. Users can easily give comments and important points about that method so author of the method can change or consider the comments entered to the system.

In lessons learned databases, each member of the project can access the database and add their experiences related with the project. These raw data is also separated into phases and situations to apply in different projects. SRMS provides this for requirement elicitation phase. The main problem with lessons learned databases is that it contains huge amount of data related with the experiences but it is hard to extract valuable information from this type of database. SRMS provides criteria which are explained in Section 3.3 that are related with project characteristics so by using SRMS, users can easily extract the method used in the requirement elicitation phase that is most suitable for their project situation. They can also create new methods according to her/his experience coming from the project. Moreover, in lessons learned databases the items stored do not have any predefined format that may lead to storing incomplete information which is useless in future. In SRMS, fields that are filled by users are previously defined so users are expected to fill complete information so that it can be used in future.

The following sections describe the SRMS tool structure and the usage in detail.

3.4.2 Use Cases

SRMS has two type of user. One of them is **user** and other one is **admin**. These are the actors of the system. Each actor of the system has some processes called use cases. Following are the typical user interaction with the system as shown in Figure 3.1.

- User adds new method information to the system.
- User adds new method by extending a method stored in the system.
- User adds new method by combining two methods stored in the system.
- User views the methods stored in the system.
- User updates the method(s) which is/are created by that user.
- User takes report about the method(s) stored in the system.
- User views the statistics about the method(s) stored in the system.

- User takes help about usage of the system.
- User views and adds method’s comments.

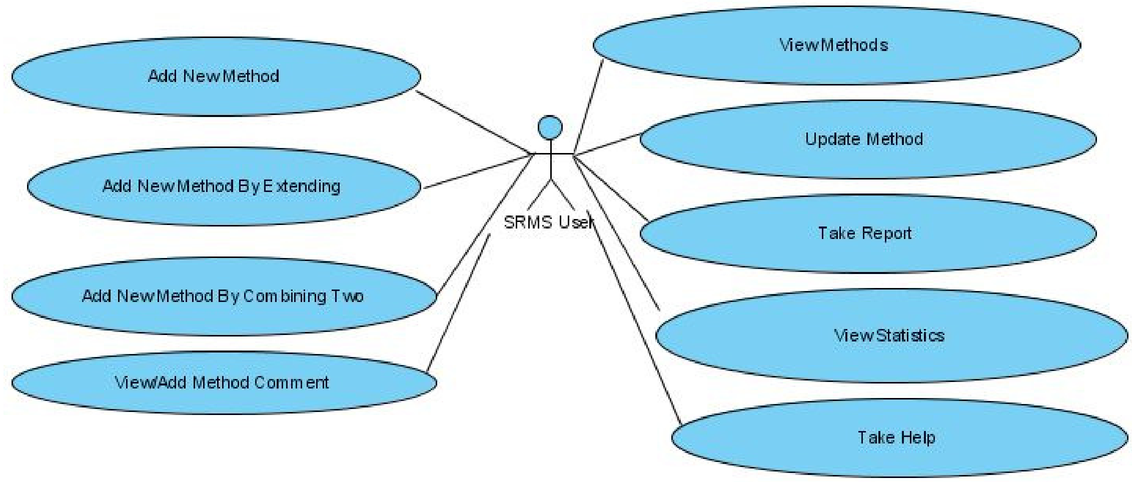


Figure 3.1 Use Case Diagram for SRMS User

Following are the other actor which is called Admin’s interaction with the system as shown in Figure 3.2:

- Admin adds new method information to the system.
- Admin adds new method by extending a method stored in the system.
- Admin adds new method by combining two methods stored in the system.
- Admin views the methods stored in the system.
- Admin updates the method(s) which is/are created by anyone who uses the system.
- Admin takes report about the method(s) stored in the system.
- Admin views the statistics about the method(s) stored in the system.
- Admin deletes the method(s).
- Admin adds new users to the system.
- Admin sets new password to the user.
- Admin deletes the user.
- Admin takes help about usage of the system.
- Admin views and adds method’s comments.
- Admin deletes comments.

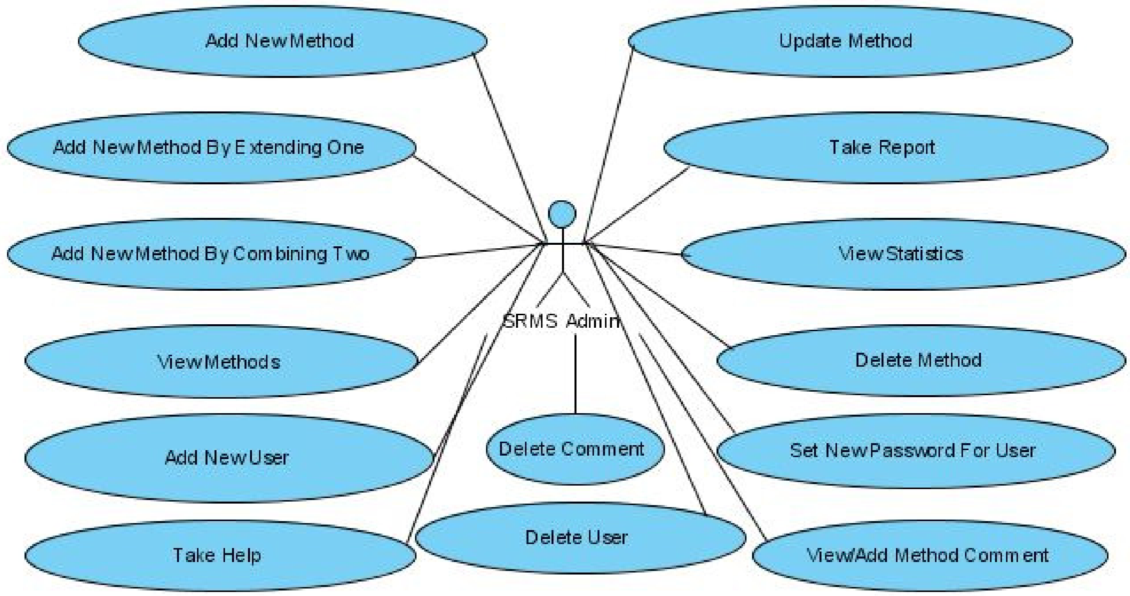


Figure 3.2 Use Case Diagram for SRMS Admin

3.4.3 SRMS Data Storage

SRMS stores the method information in a MYSQL database which is free of use, can be easily retrieved from the Internet. Method, Step, Comments and Users tables and the relationships are as shown in Figure 3.3. Users of the system are stored in the users table. Methods are stored in the method table and the steps of the methods are stored in the step table. One user can have many methods but one method can not belong to more than one user. One method can have many steps related with the method but one step can not belong to more than one method.

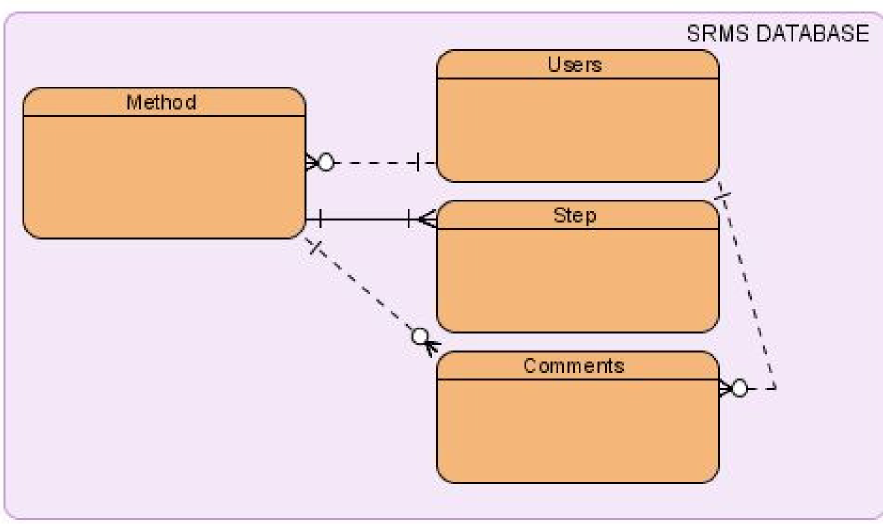


Figure 3.3 SRMS Database and Relations

Method information contains method_id, method_name, experience_year, sim_experience, req_days, cust_experience, meetings, project_budget, user_interaction, project_complexity, added_by, method_type, viewed and used_creation columns to store the method criterias. Comments table contains comment_id, method_id, comment_info and added by columns to store comments. Step table contains step_id, method_id and step_info as shown in Figure 3.4. Users table contains uid, username and password to store the user information of the system. The important thing here is storing the method data does not require a qualified computer.

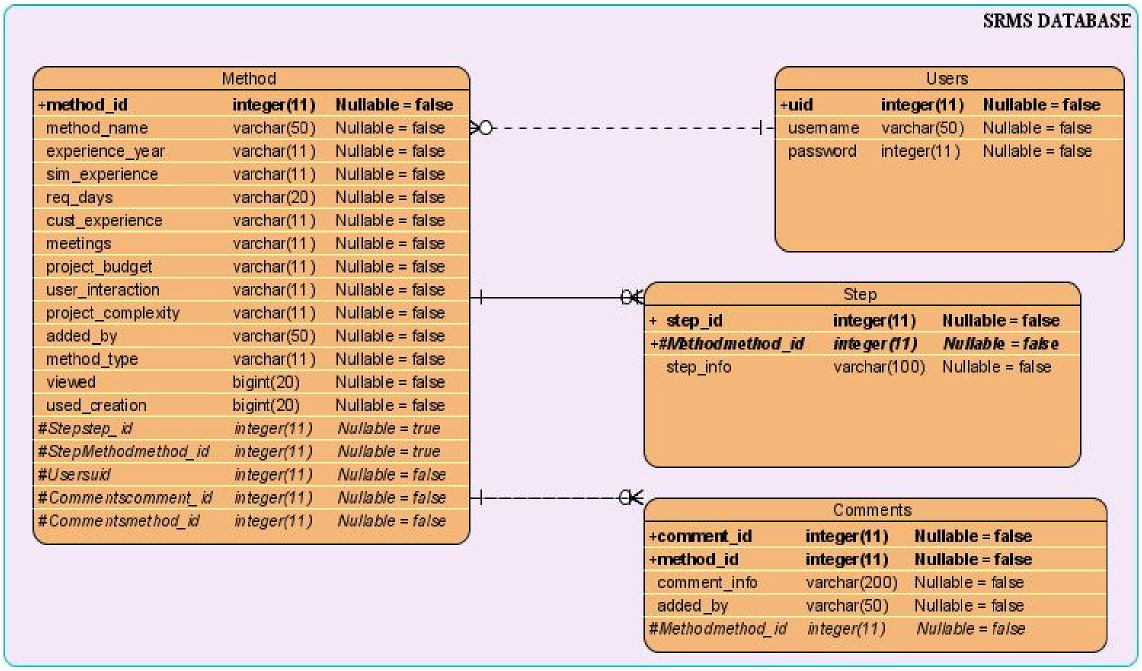


Figure 3.4 Detailed View of SRMS Database

3.4.4 SRMS Actions and Commands

SRMS tool is composed of seven modules for typical user interactions and eleven modules for admin. Capabilities of each type of user are explained in Section 3.4.2 Use Cases in detail. Typical user capabilities are shown in Figure 3.5. Admin has more capabilities compared to typical user. Admin can use the all features of typical user and more features as shown in Figure 3.6.

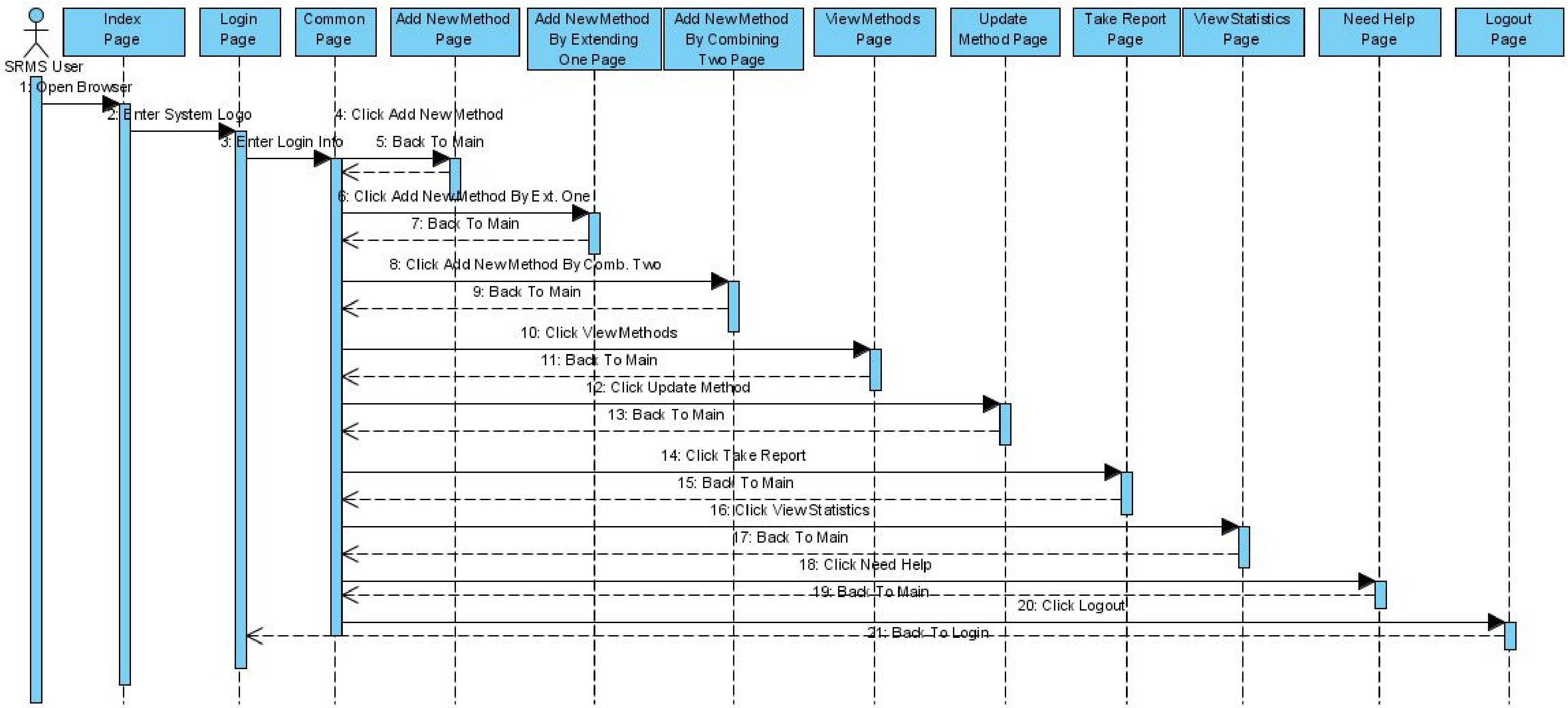


Figure 3.5 Sequence Diagram of Typical User of SRMS

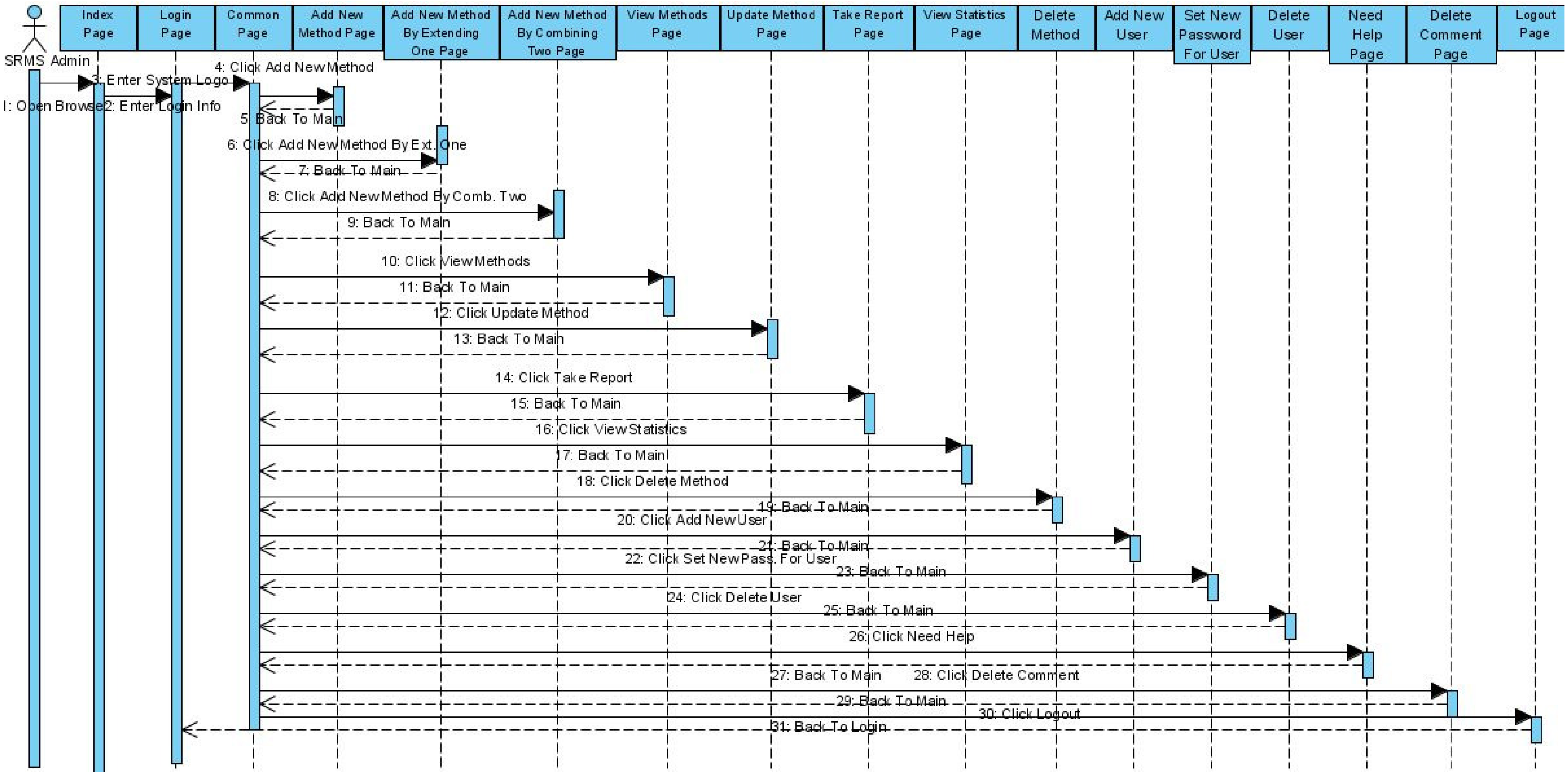


Figure 3.6 Sequence Diagram of Admin of SRMS

3.4.5 SRMS's States

Typical user just opens the browser and types the address of the system, SRMS welcomes user by index page. When user clicks the system logo, he/she redirected to login page to enter to the system. After successful login, main menu is displayed to the user. According to user's selection proper module is displayed to the user. One of the typical user capabilities is Adding New Method to add new method to the system with his/her experience as shown in Figure 3.7.

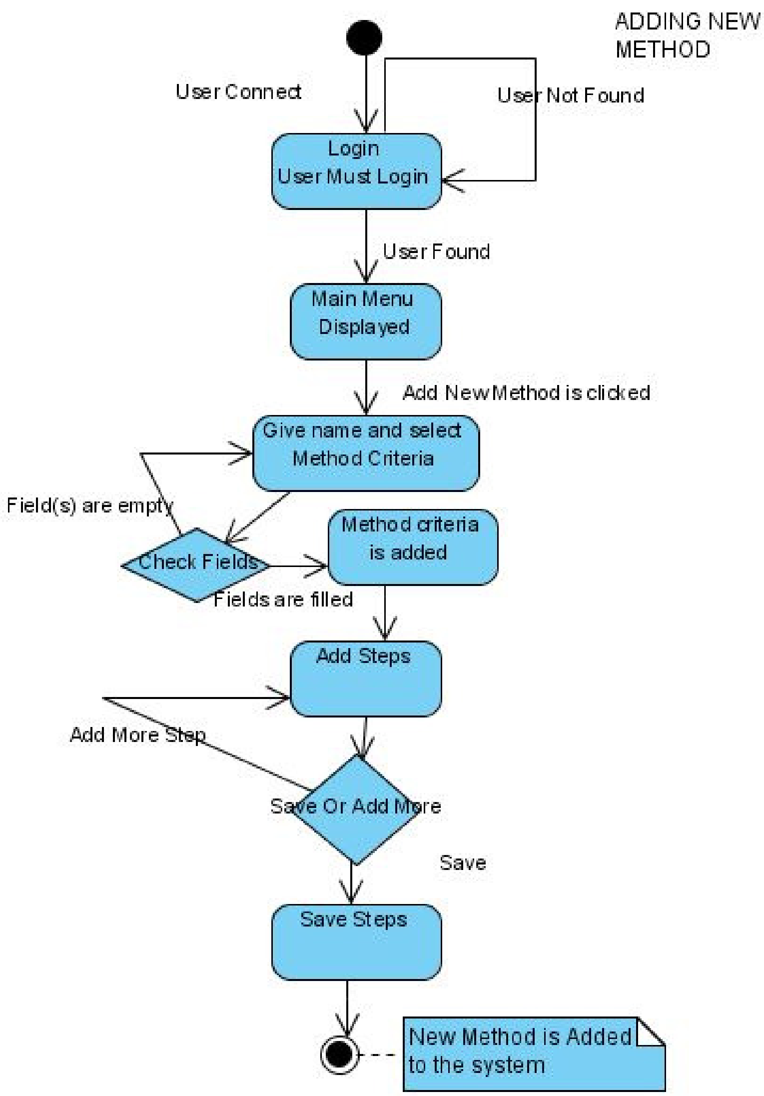


Figure 3.7 State Diagram of Adding New Method

User prefers this capability when the methods stored in the system are not matched with the user's case. If a method stored in the system partially matches the user's case user prefers to use Add New Method By Extending One for adding new method by using an existing method as shown in Figure 3.8.

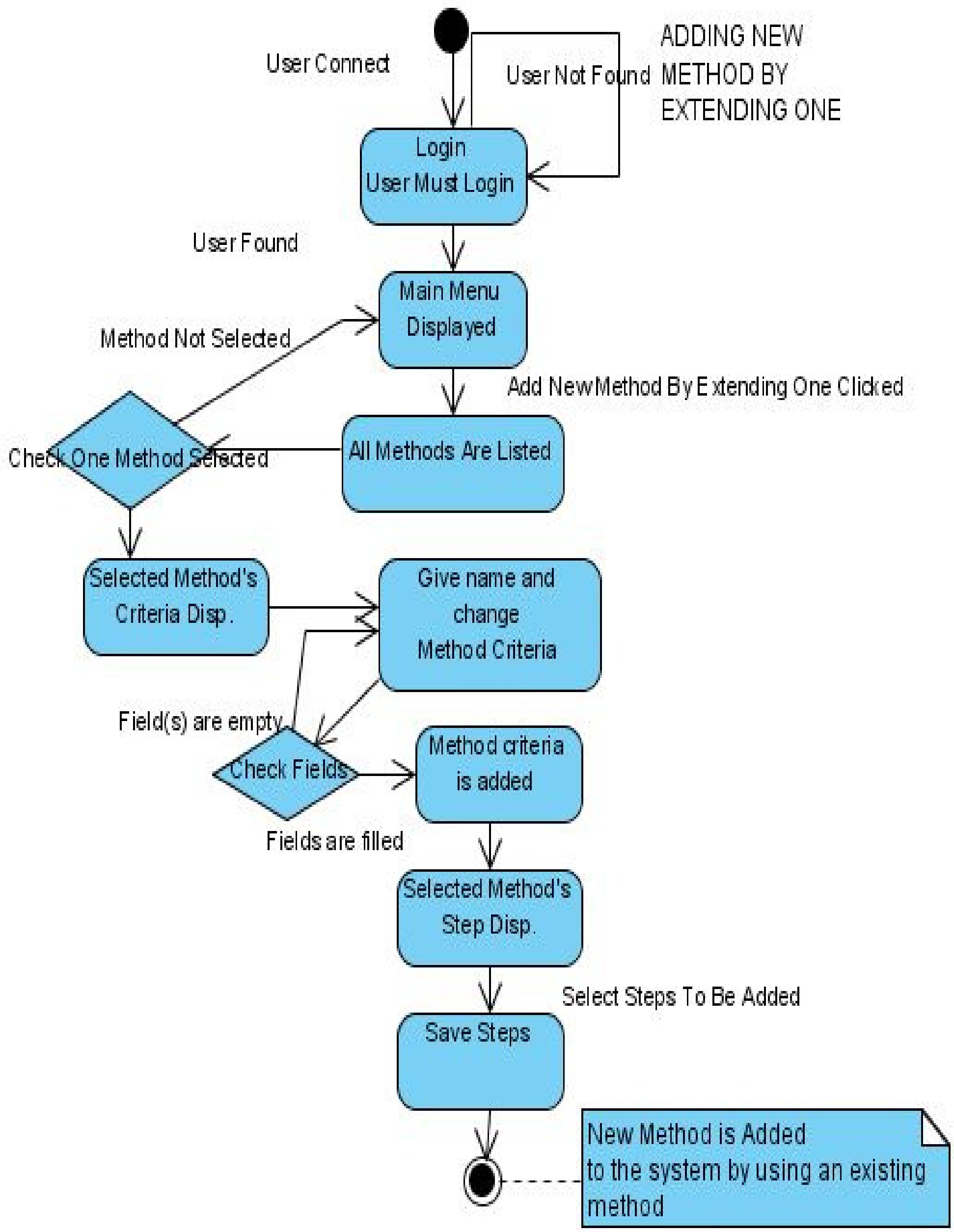


Figure 3.8 State Diagram of Adding New Method By Extending One

User can find two methods matching with user's case; in this circumstance user prefers to use Add New Method By Combining Two capability for adding new method by using two method as shown in Figure 3.9.

There are also other capabilities rather than adding method such as Viewing Methods stored in the system. Moreover, user can add comments about each method in the system by using this module. This capability provides user to examining the methods stored in the system before adding a new one or just using a method matching with user's case shown in Figure 3.10.

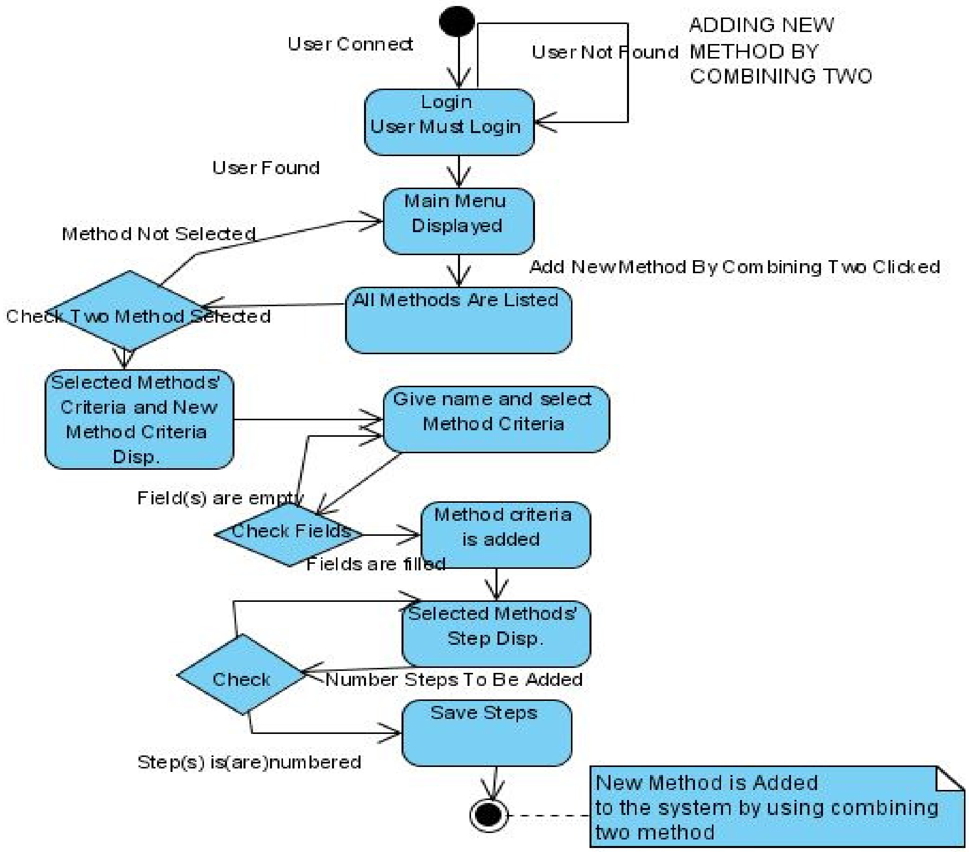


Figure 3.9 State Diagram of Adding New Method By Combining Two

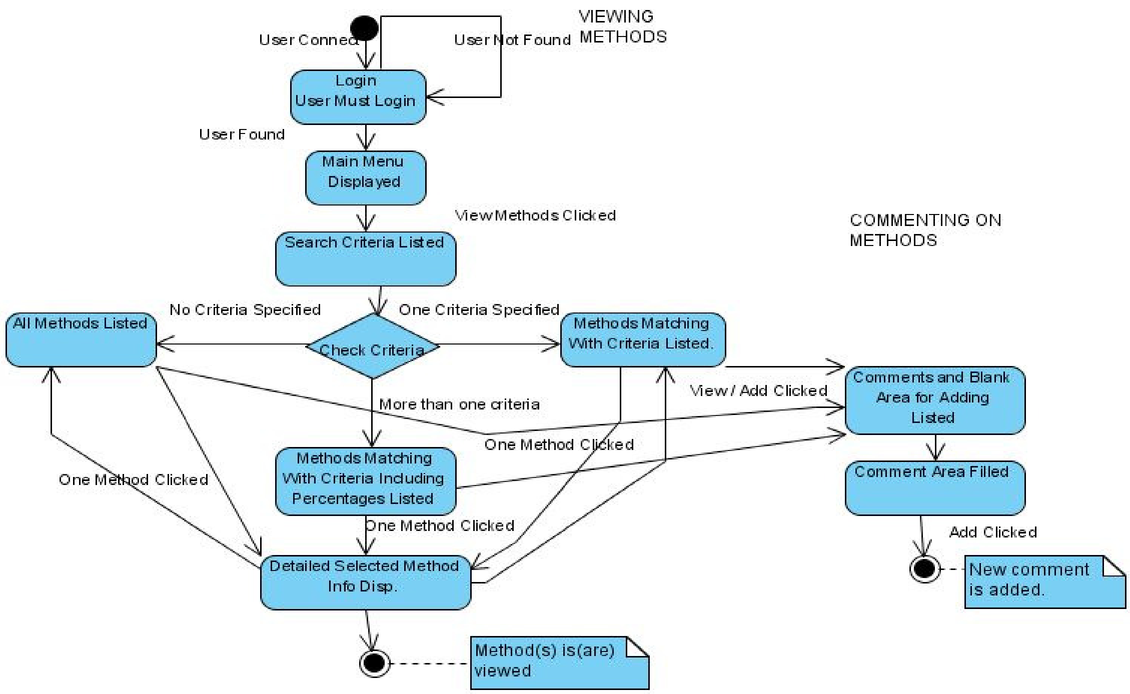


Figure 3.10 State Diagram of View Methods

User can also view the graphical representation of the method related statistics stored in the system. For example: which type of method is created more, which user created more, etc. According to the selected criteria proper graph is displayed to the user. This capability gives idea about the methods in the system as shown in Figure 3.11.

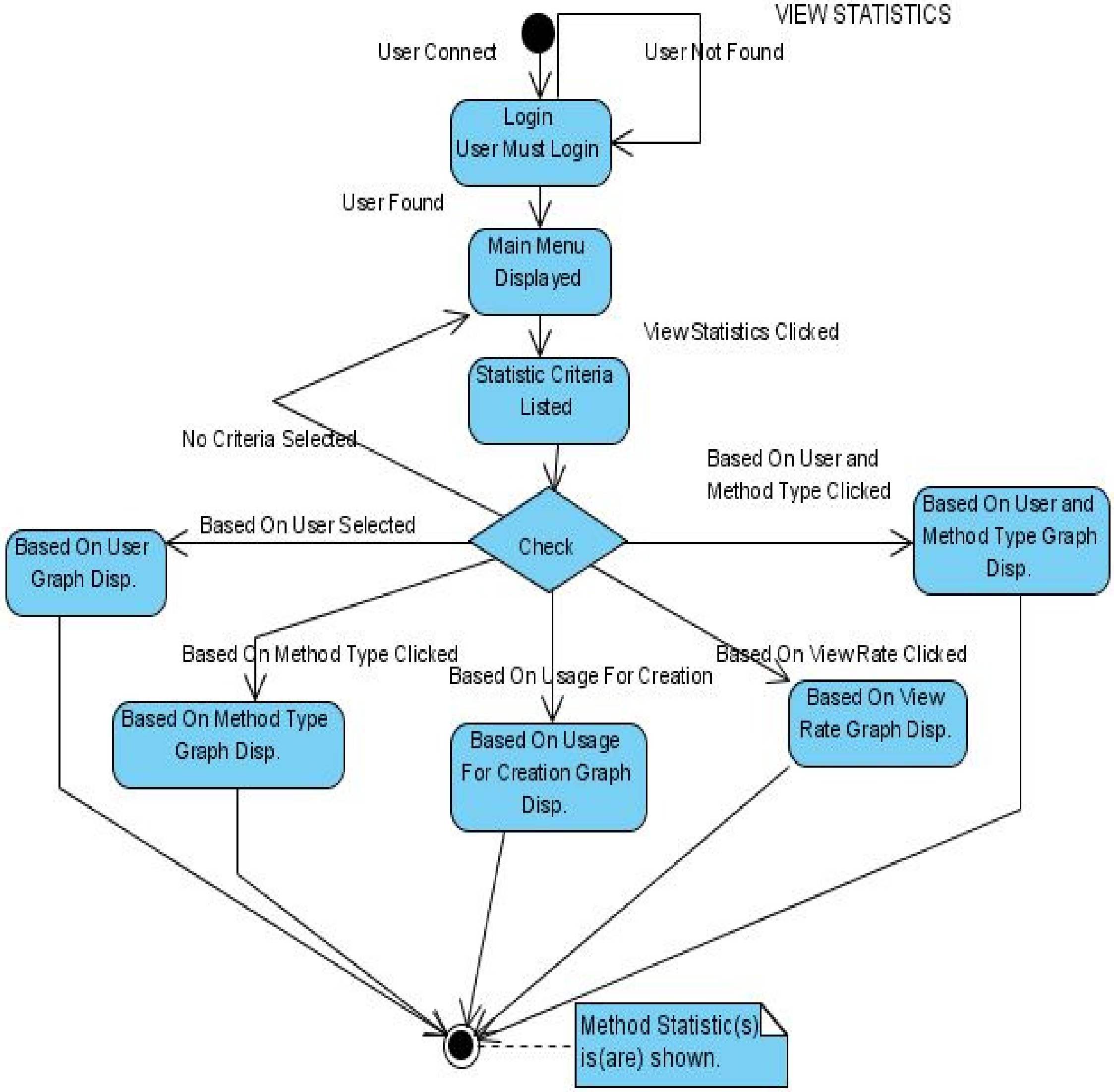


Figure 3.11 State Diagram of View Statistics

Methods stored in the system can be updated by the user. User can use Update Method capability to update the method criteria, steps and add more step(s) to the method. This capability provides user to change the method after creation as shown in Figure 3.12.

Moreover, user can take report about the methods stored in the system by Take Report capability. Method detailed information listed in a report format and user can take print from this page as shown in Figure 3.13.

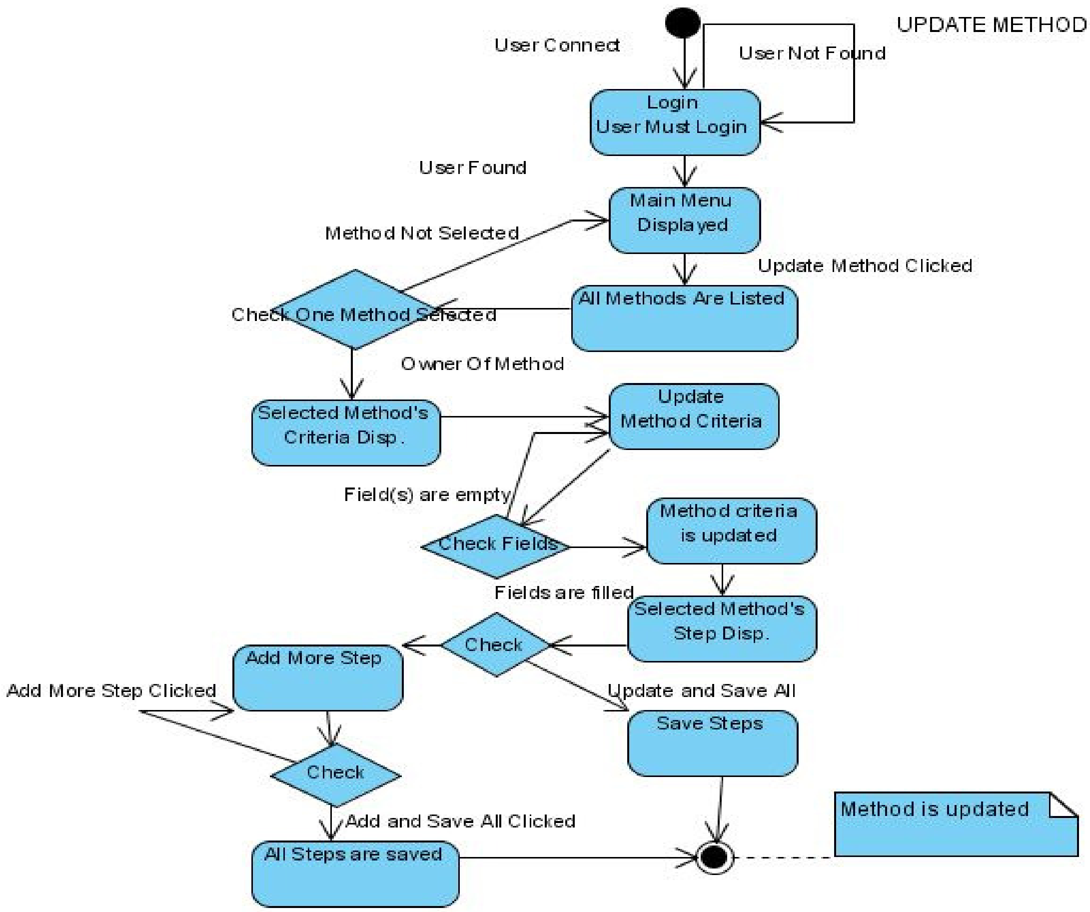


Figure 3.12 State Diagram of Update Method

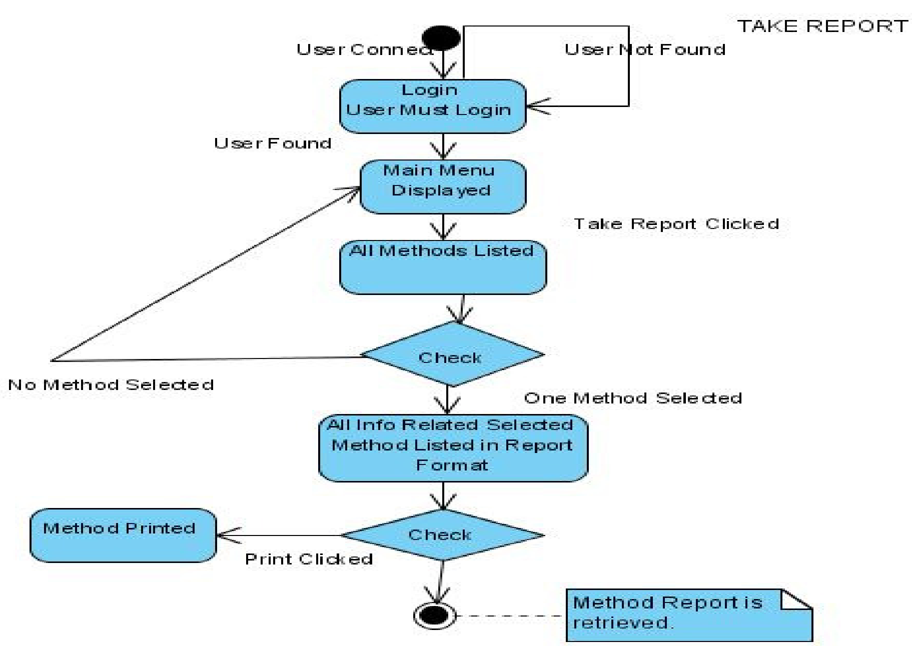


Figure 3.13 State Diagram of Take Report

Obviously, it is a necessity to refer a help when using a system. It is also provided to users of the system. When user needs some help about the system, user needs to use Need Help module. User can see a little information about the system and features in items form. Moreover, detailed features can also be retrieved from this module. Need Help capability is shown in Figure 3.14.

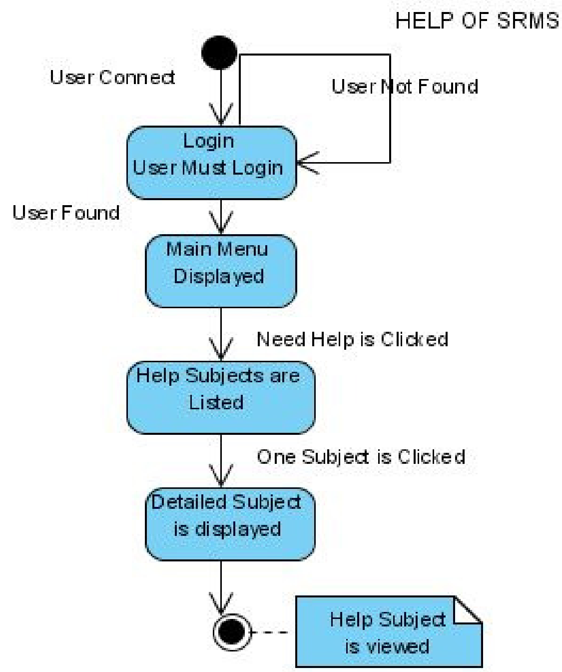


Figure 3.14 State Diagram of Need Help

Admin can do the all user capabilities and has more capabilities user can not do such as deleting a method. Users can not delete a method stored in the system so the loss of knowledge is prevented. Admin Delete Method capability is shown in Figure 3.15.

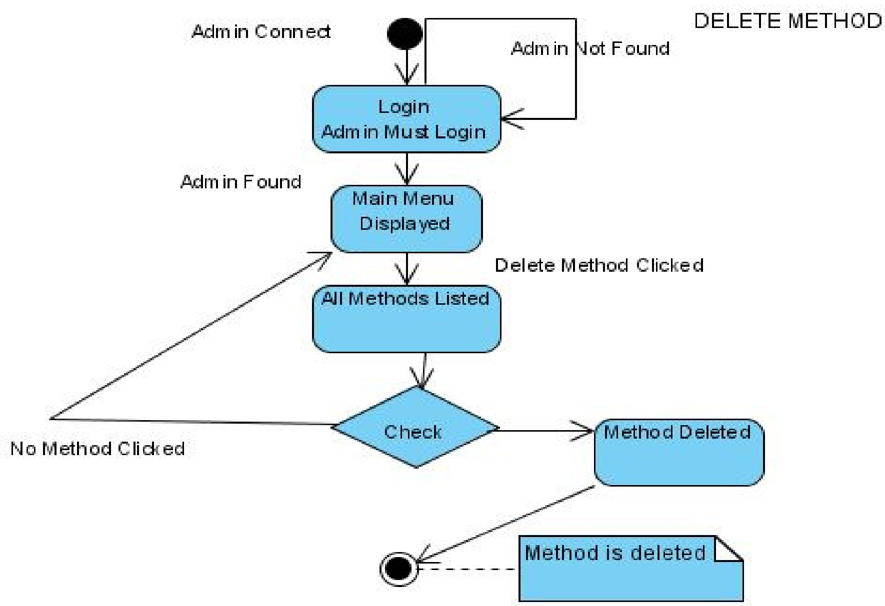


Figure 3.15 State Diagram of Delete Method

Admin of the system can also add new users to the system. Username and password are given by the admin to the user. Admin's Add New User capability is shown in Figure 3.16.

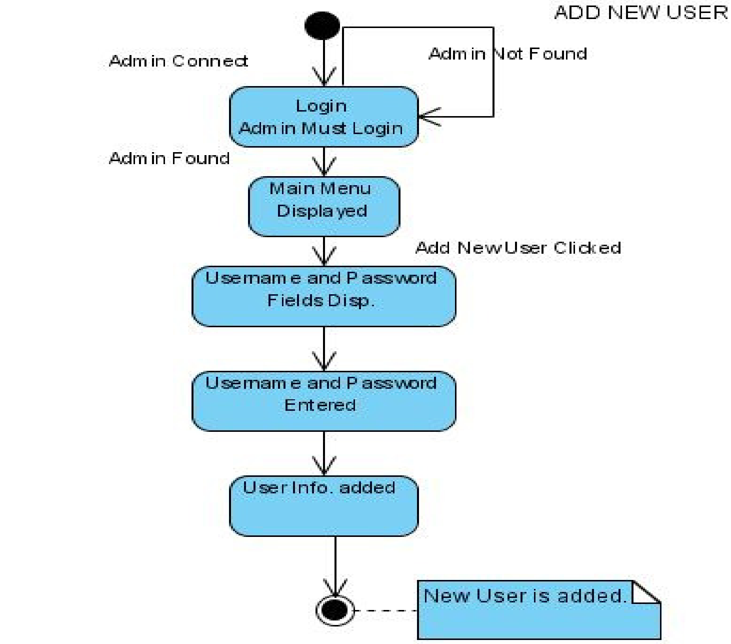


Figure 3.16 State Diagram of Add New User

Besides adding new user to the system, admin have two more capability such as setting new password for the user and deleting a user from the system. After adding a user, user can forget his/her password, to set a new password for this user admin can use Set New Password for User capability as shown in Figure 3.17.

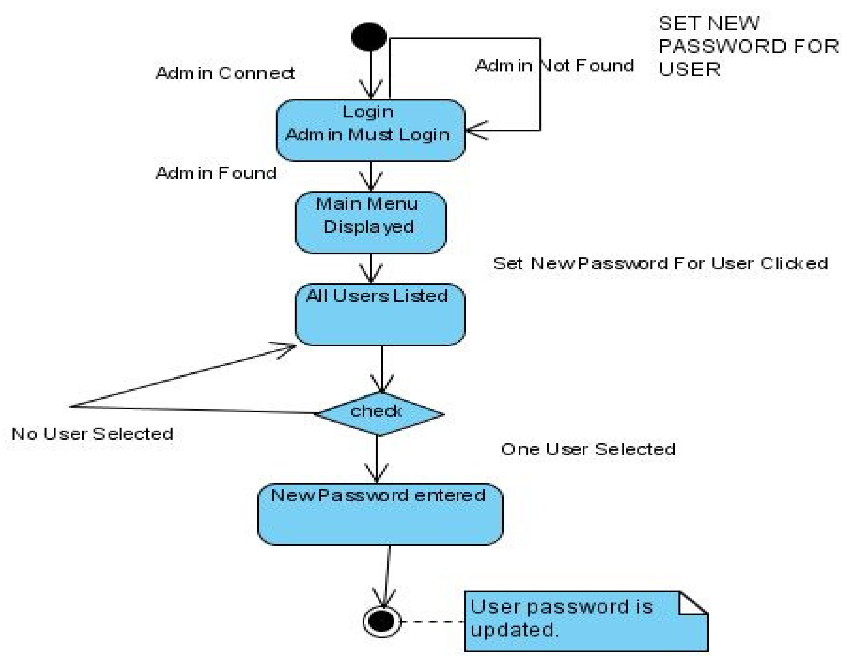


Figure 3.17 State Diagram of Set New Password For User

Moreover, admin can delete a user entirely from the system. To do this, admin can use Delete Method capability as shown in Figure 3.18.

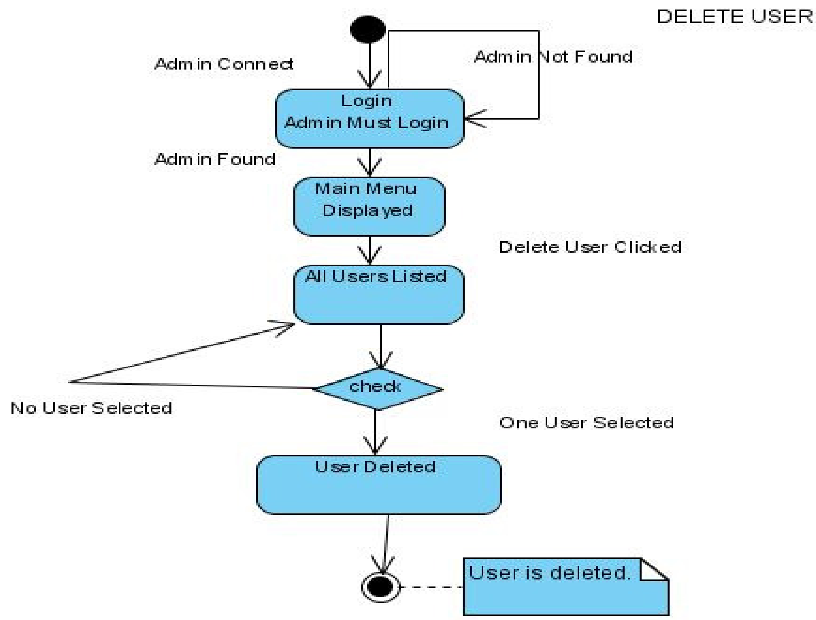


Figure 3.18 State Diagram of Delete User

System can contain irrelevant comments related with the methods stored in the system. Admin can delete these types of comments by using Delete Comment module as shown in Figure 3.19.

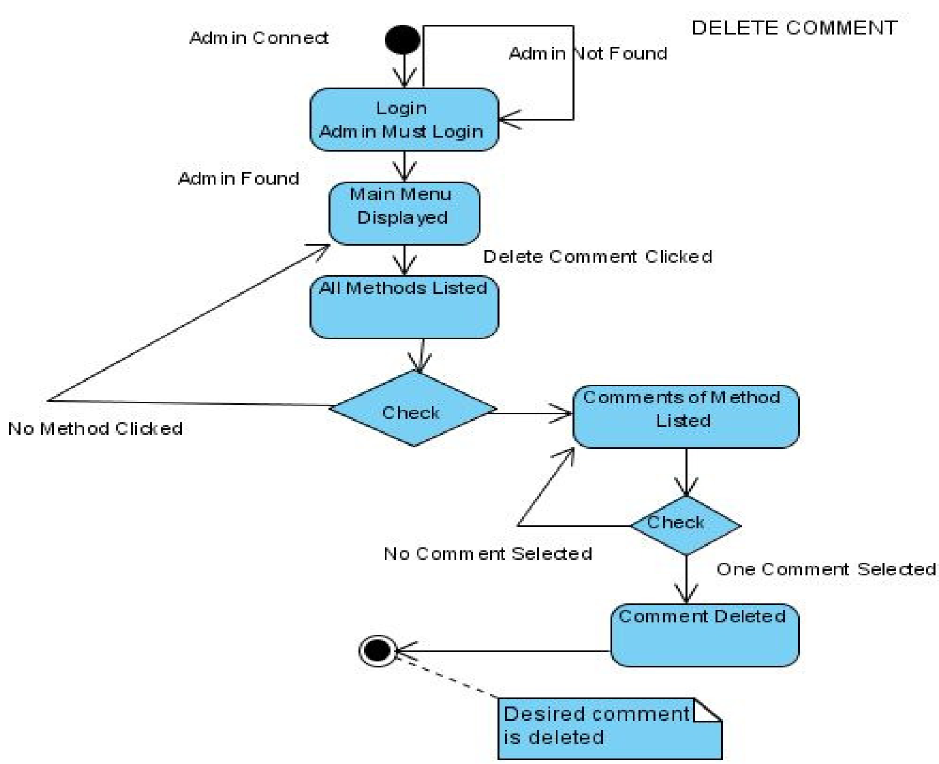


Figure 3.19 State Diagram of Delete Comment

3.4.6 SRMS Usage

SRMS tool is user-friendly and easy to understand. There are no complicated features to manage. When client browser is opened and the system address is entered index page of the system welcomes the user as shown in Figure 3.20.

Figure 3.20 Index Page

User clicks the image on this page and redirected to the login page, user enters his/her username, password and clicks Login as shown in Figure 3.21.

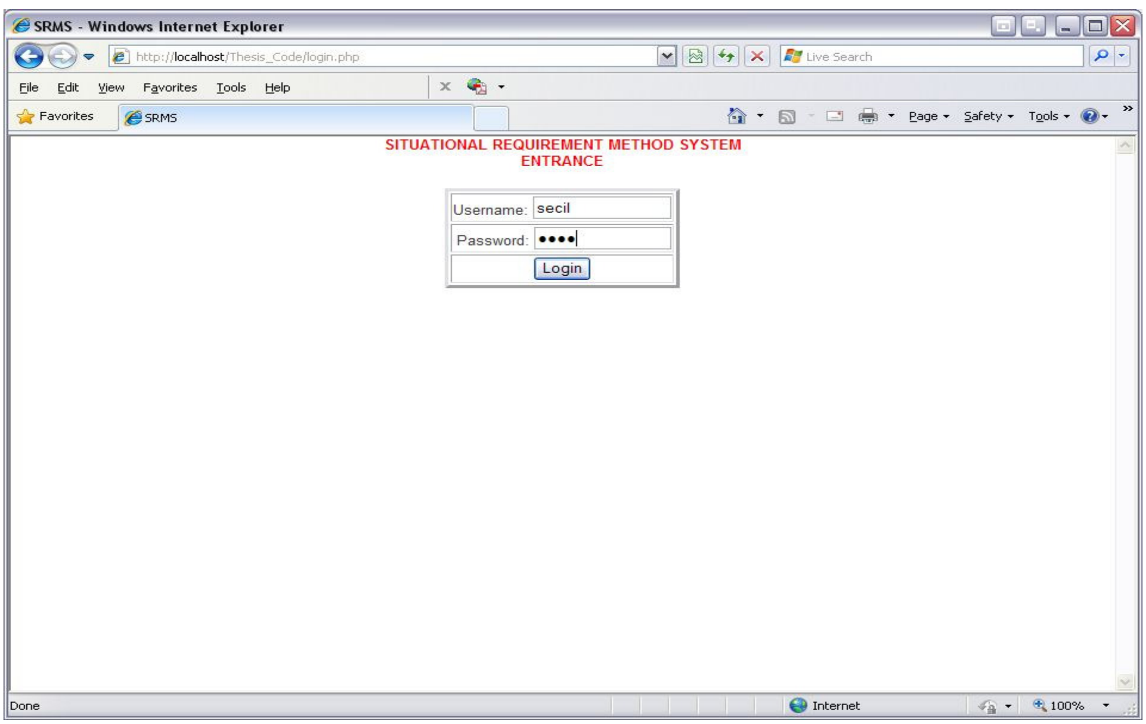


Figure 3.21 Login Page

According to user type, main menu is displayed. Figure 3.22 shows the page for typical user.

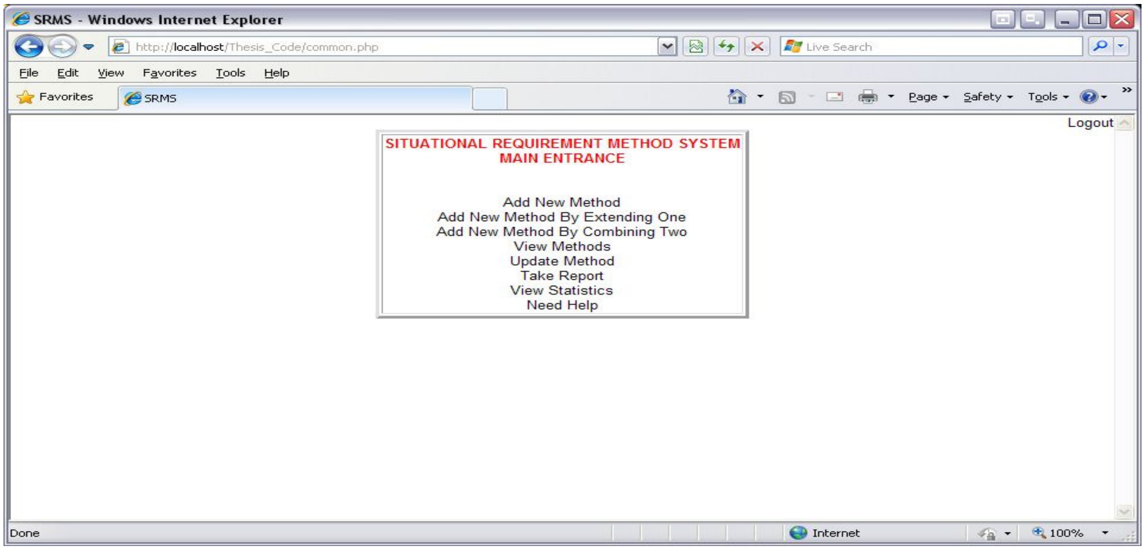


Figure 3.22 Main Page

Methods that are stored can be viewed by View Method module. It is important to see all methods in the repository before creating a new one. User can also view the statistical data in graphical representation by View Statistics module. This module will be explained later in detail. When user enters View Method module, there are search criteria which are already explained in Section 3.3 as shown in Figure 3.23. User can search for the methods based on these criteria to best suit the project situation at hand. All the methods satisfying given criteria are displayed. If there are no criteria specified, all the methods stored in the system will be displayed to the user as shown in Figure 3.24.

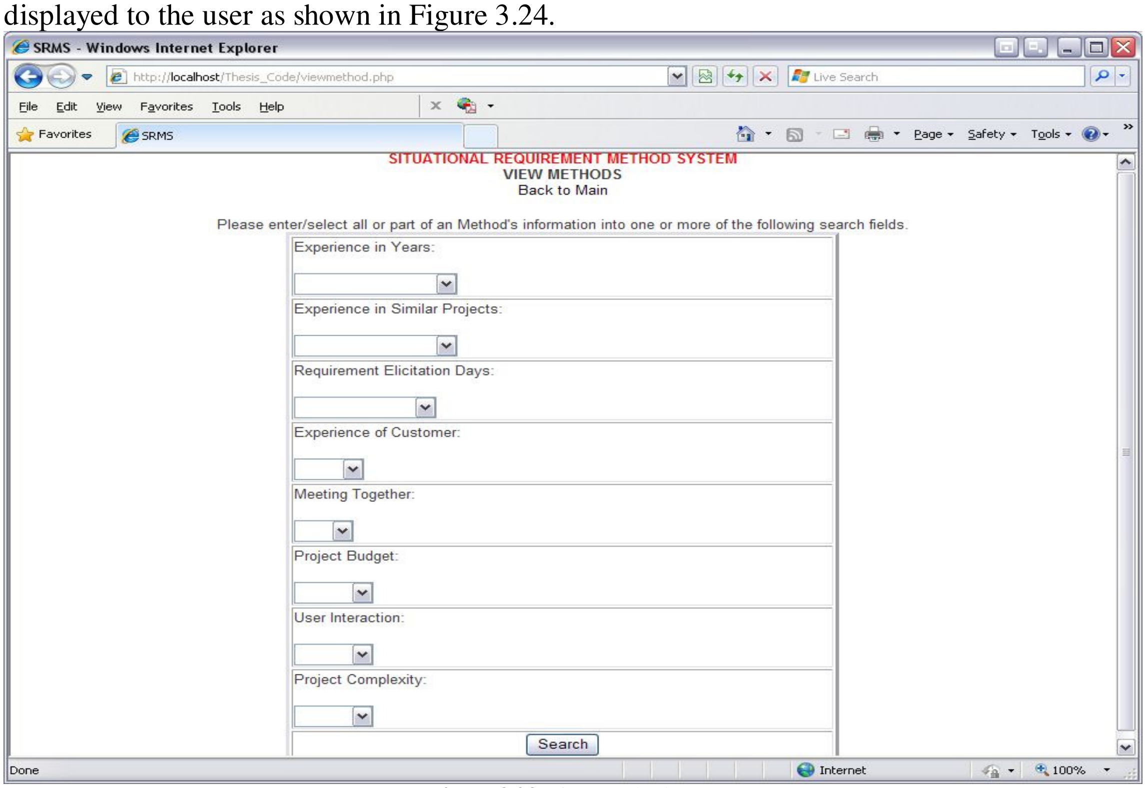
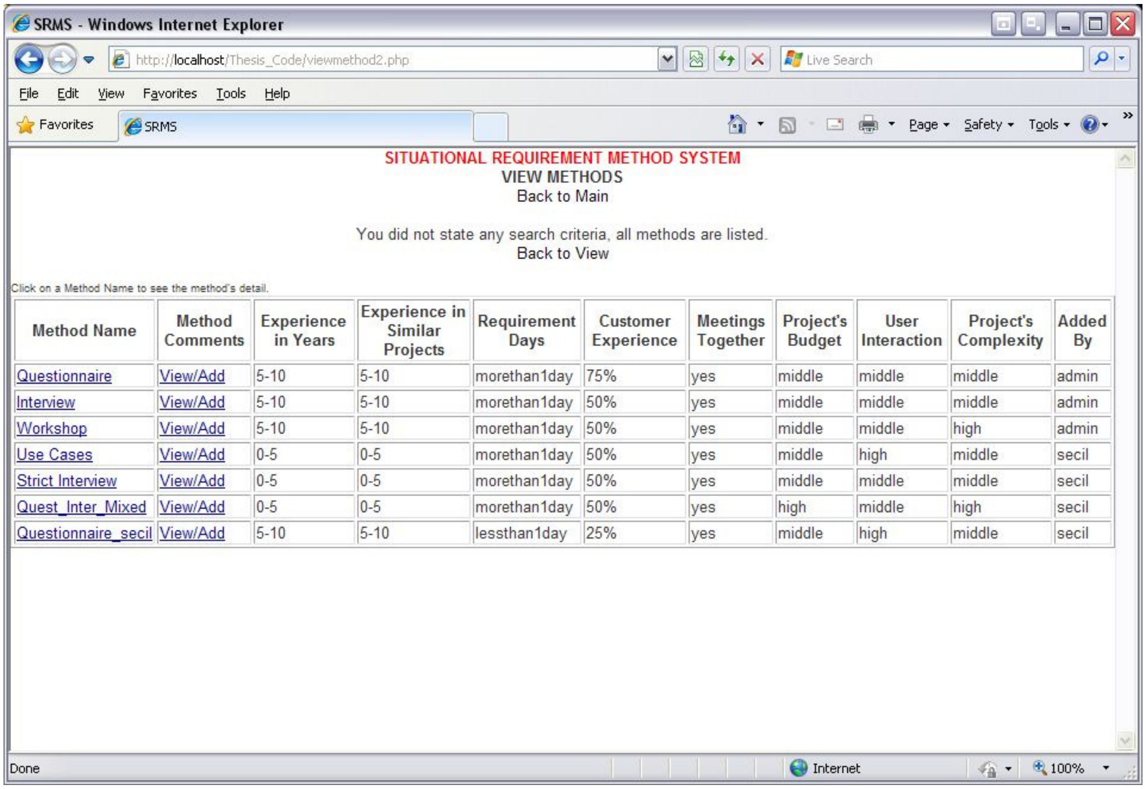


Figure 3.23 View Method Page



Method Name	Method Comments	Experience in Years	Experience in Similar Projects	Requirement Days	Customer Experience	Meetings Together	Project's Budget	User Interaction	Project's Complexity	Added By
Questionnaire	View/Add	5-10	5-10	morethan1day	75%	yes	middle	middle	middle	admin
Interview	View/Add	5-10	5-10	morethan1day	50%	yes	middle	middle	middle	admin
Workshop	View/Add	5-10	5-10	morethan1day	50%	yes	middle	middle	high	admin
Use Cases	View/Add	0-5	0-5	morethan1day	50%	yes	middle	high	middle	secil
Strict Interview	View/Add	0-5	0-5	morethan1day	50%	yes	middle	middle	middle	secil
Quest_Inter_Mixed	View/Add	0-5	0-5	morethan1day	50%	yes	high	middle	high	secil
Questionnaire_secil	View/Add	5-10	5-10	lessthan1day	25%	yes	middle	high	middle	secil

Figure 3.24 View Method Page(all methods listed)

User can also view the details of a method by clicking the method name listed in Method Name column of the table as shown in Figure 3.25.

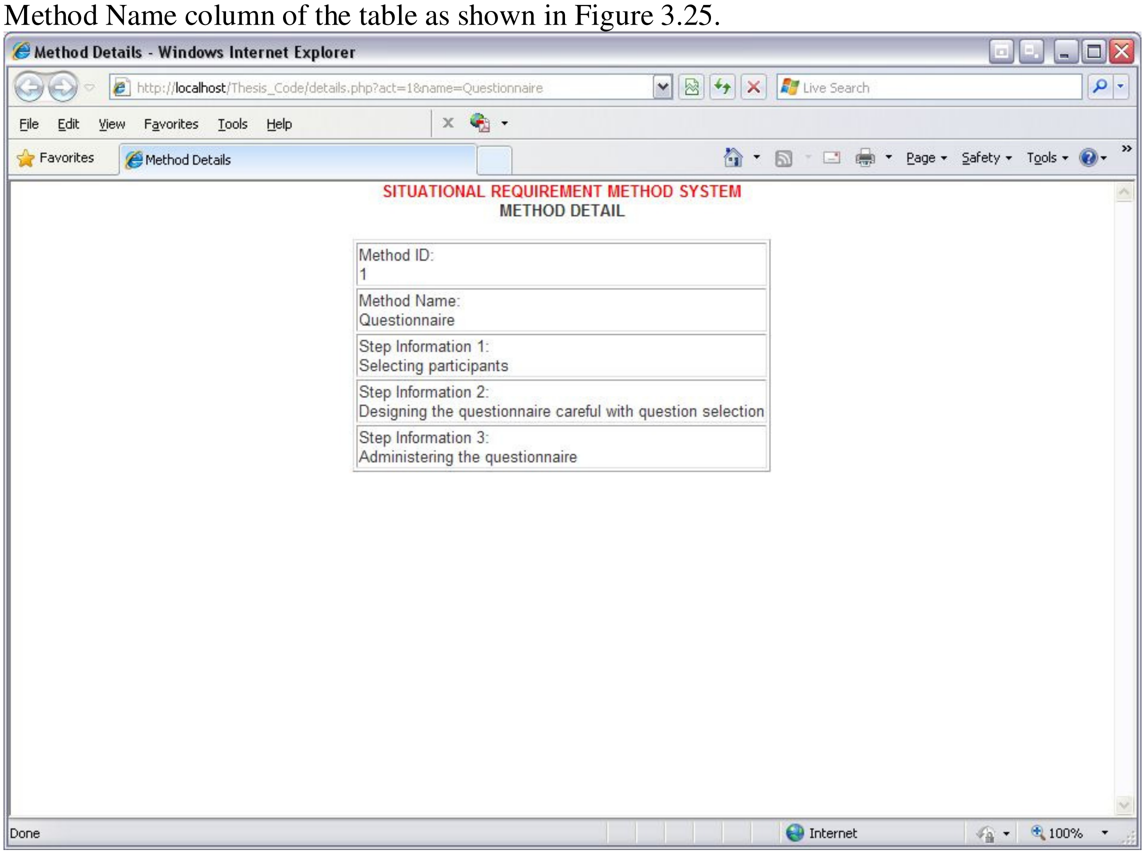


Figure 3.25 Method Detail Page

User can also use View/Add Comment feature. This feature provides viewing, adding of important comments about each method. For specifying the domain of the method this feature can be used. Each method listed in the table has a view/add link for viewing the method comments. When user clicks on this link, user can view comments and can add new comments related with this method at the same page as shown in Figure 3.26.

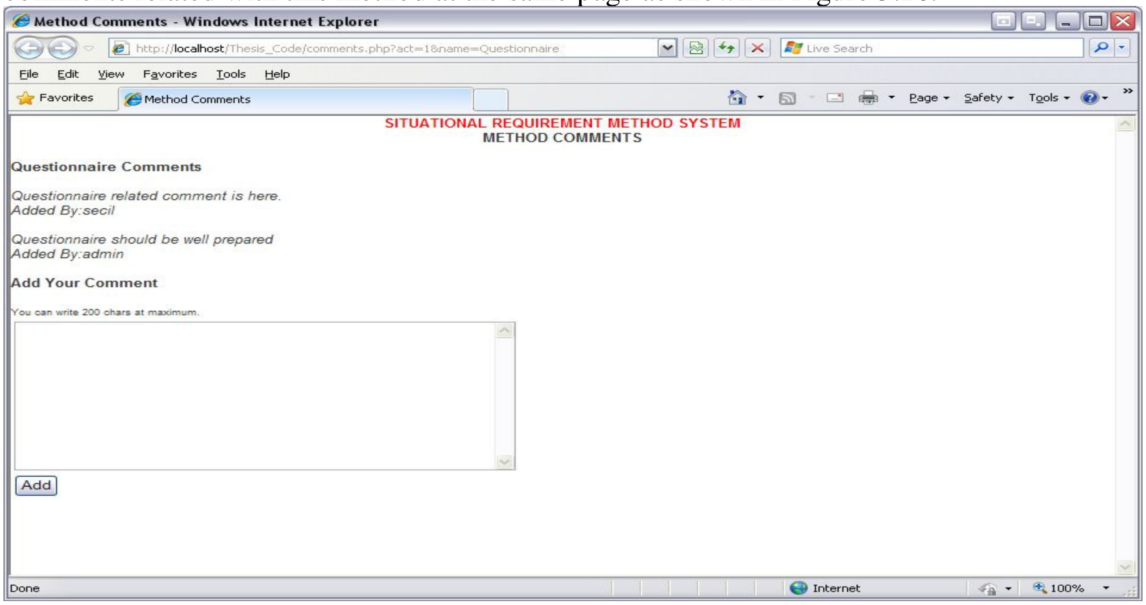


Figure 3.26 Method View/Add Comment Page

If user states a criterion (only one criteria) such as Project Complexity is *high* and clicks Search button as shown in Figure 3.27; if there are any matching method it will displayed in the results as shown in Figure 3.28. If there is not any method matching with the criteria system gives an error message in red and advices user to return and state a new search criteria.

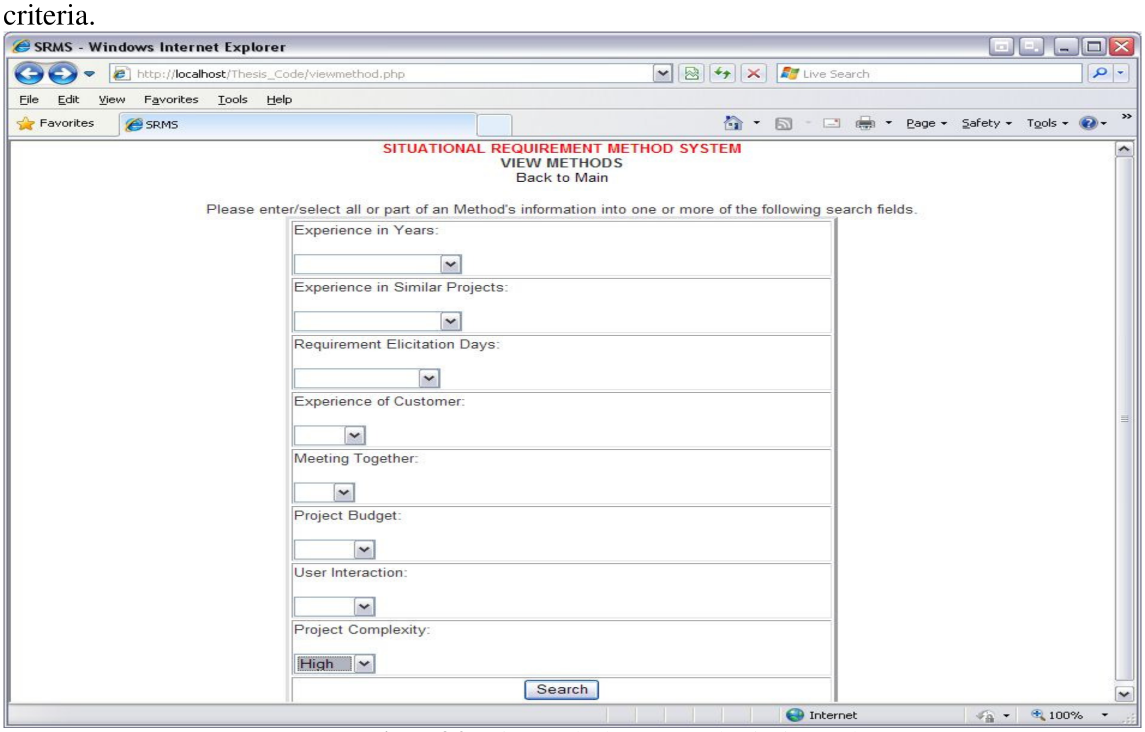


Figure 3.27 View Method Page (search criteria stated)

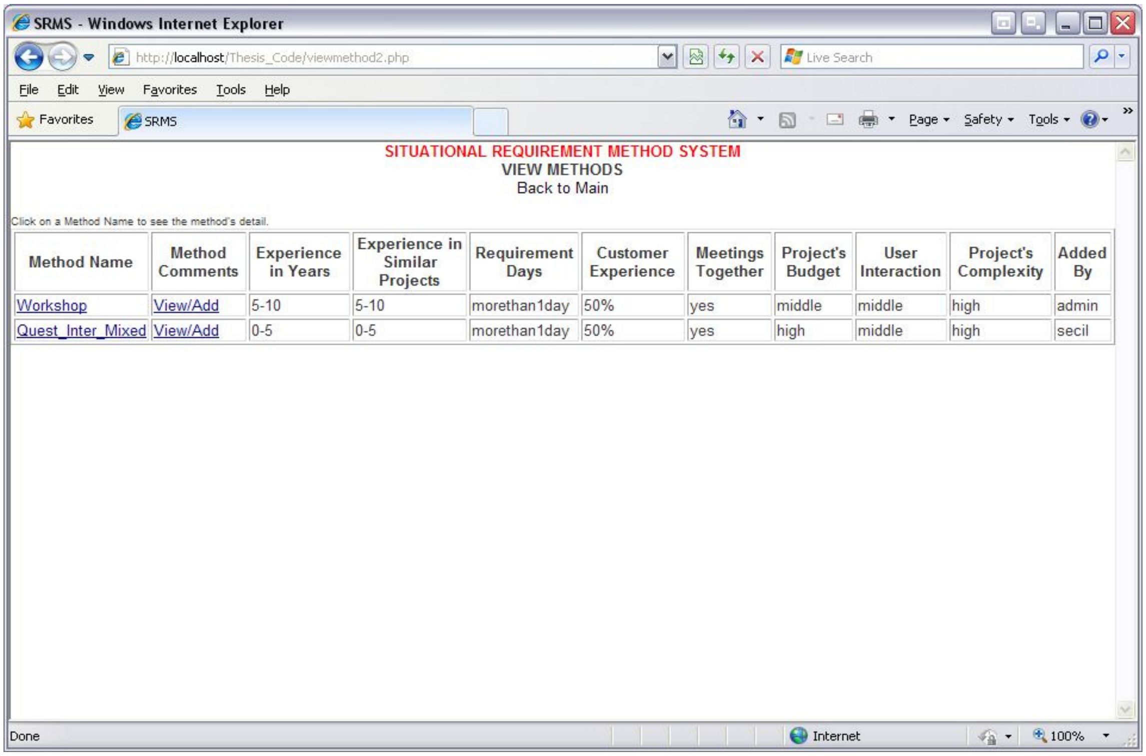


Figure 3.28 View Method Page (search result)

More than one criterion can be stated such as Experience in Years, Experience in Similar Projects and requirement elicitation days and Search button is clicked as shown in Figure 3.29, results matching with the criteria is displayed to the user with the percentage of match. This feature provides user to see how many of criteria matched with the user's criteria as shown in Figure 3.30.

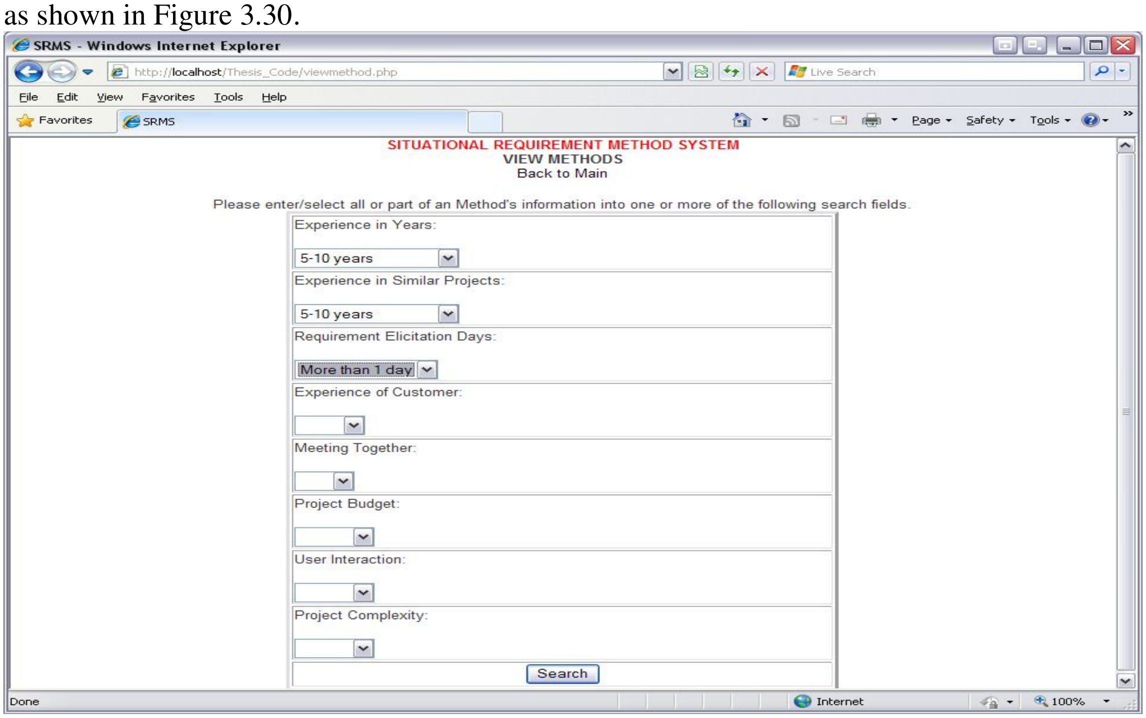
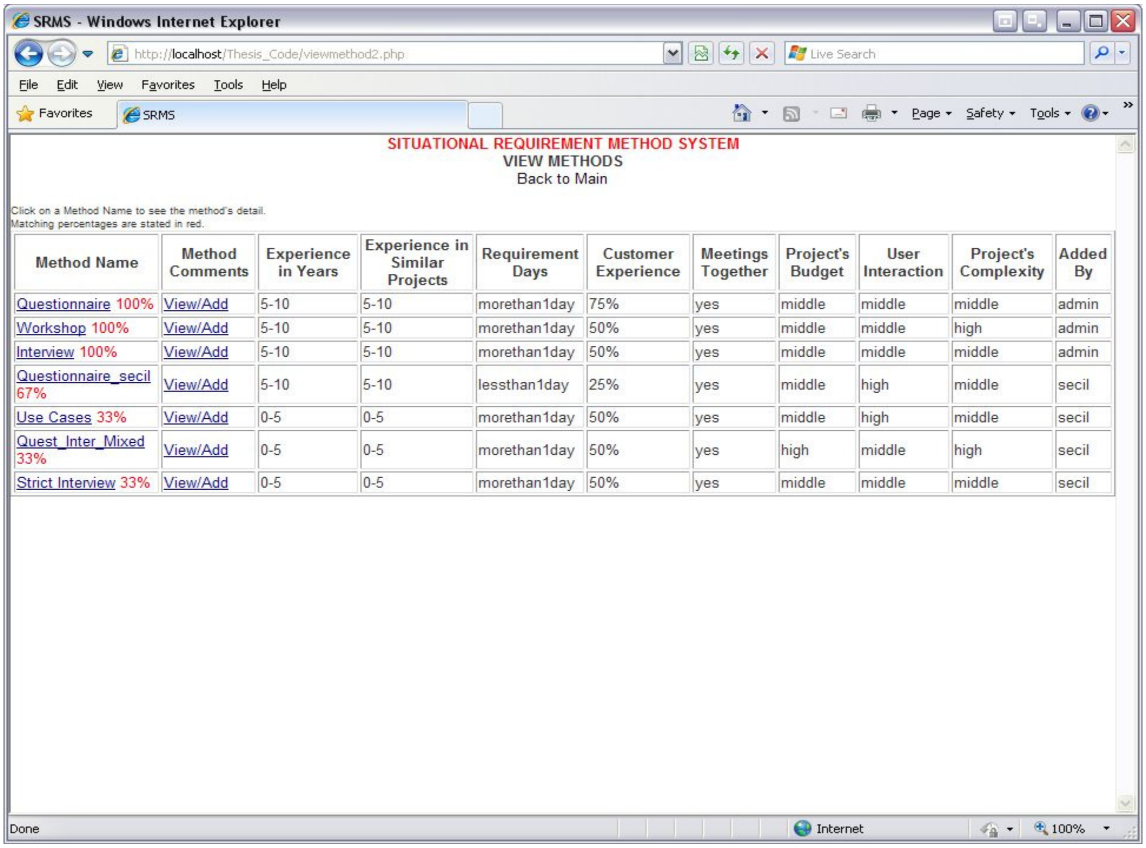


Figure 3.29 View Method Page (more than one criterion specified)



SITUATIONAL REQUIREMENT METHOD SYSTEM
VIEW METHODS
Back to Main

Click on a Method Name to see the method's detail.
Matching percentages are stated in red.

Method Name	Method Comments	Experience in Years	Experience in Similar Projects	Requirement Days	Customer Experience	Meetings Together	Project's Budget	User Interaction	Project's Complexity	Added By
Questionnaire 100%	View/Add	5-10	5-10	morethan1day	75%	yes	middle	middle	middle	admin
Workshop 100%	View/Add	5-10	5-10	morethan1day	50%	yes	middle	middle	high	admin
Interview 100%	View/Add	5-10	5-10	morethan1day	50%	yes	middle	middle	middle	admin
Questionnaire_secil 67%	View/Add	5-10	5-10	lessthan1day	25%	yes	middle	high	middle	secil
Use Cases 33%	View/Add	0-5	0-5	morethan1day	50%	yes	middle	high	middle	secil
Quest_Inter_Mixed 33%	View/Add	0-5	0-5	morethan1day	50%	yes	high	middle	high	secil
Strict Interview 33%	View/Add	0-5	0-5	morethan1day	50%	yes	middle	middle	middle	secil

Figure 3.30 View Method Page (results with matching rates)

If desired method is still not found in the system, SRMS enables user to create a new method. There are three different method creation types in SRMS. A method can be created by an empty method template, by modifying a method stored in the repository and by combining two methods stored in the repository. This type of storage provides flexibility to the user to emphasize and transfer his/her experience.

If user wants to add a new method from beginning clicks Add New Method and redirected to a page for selecting criteria and giving a name for new method as shown in Figure 3.31.

After that, user clicks Next and all the information is added to the repository and a page called addstep is displayed to the user. User adds steps of the method as shown in Figure 3.32 and clicks SaveAll to add them to the repository so all the steps that are written by the user added to the repository, system gives a message to user as shown in Figure 3.33.

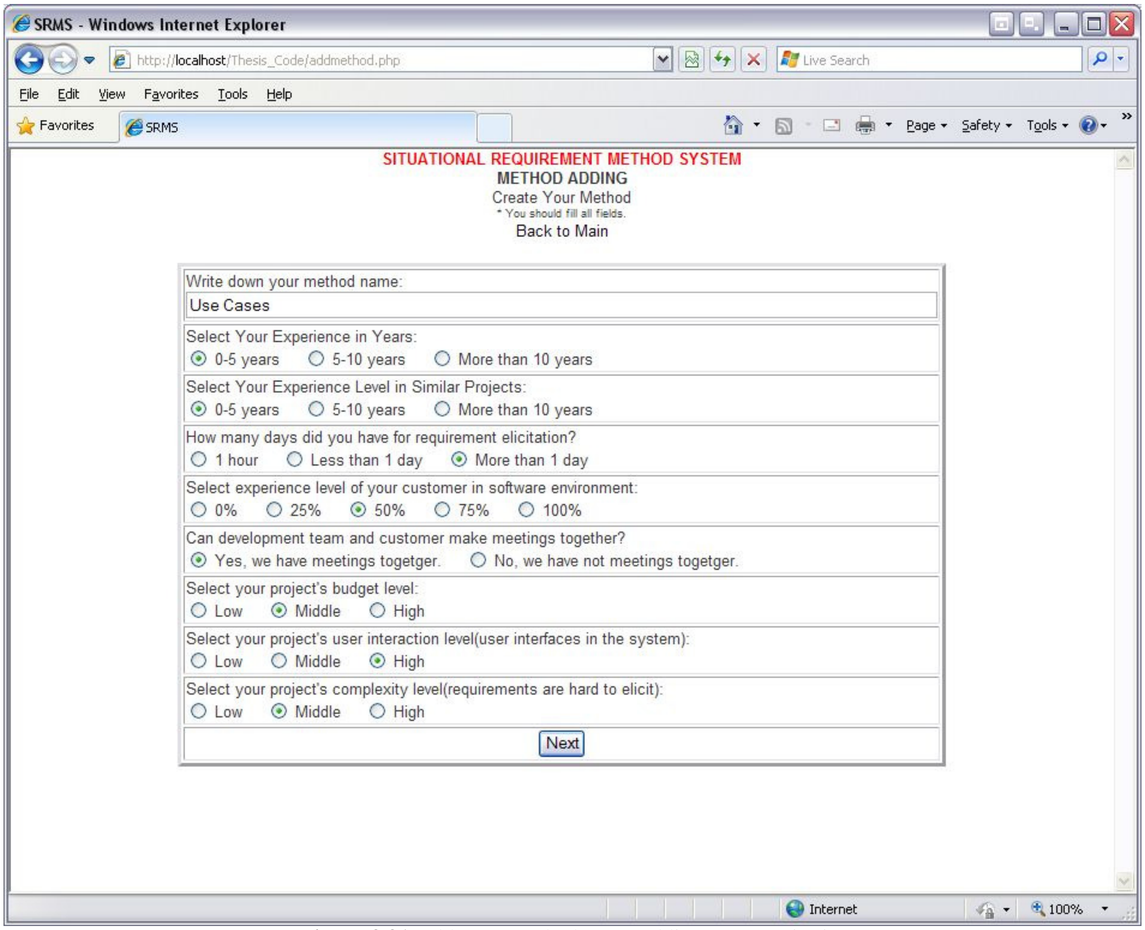


Figure 3.31 Add New Method Page (adding name&criteria)

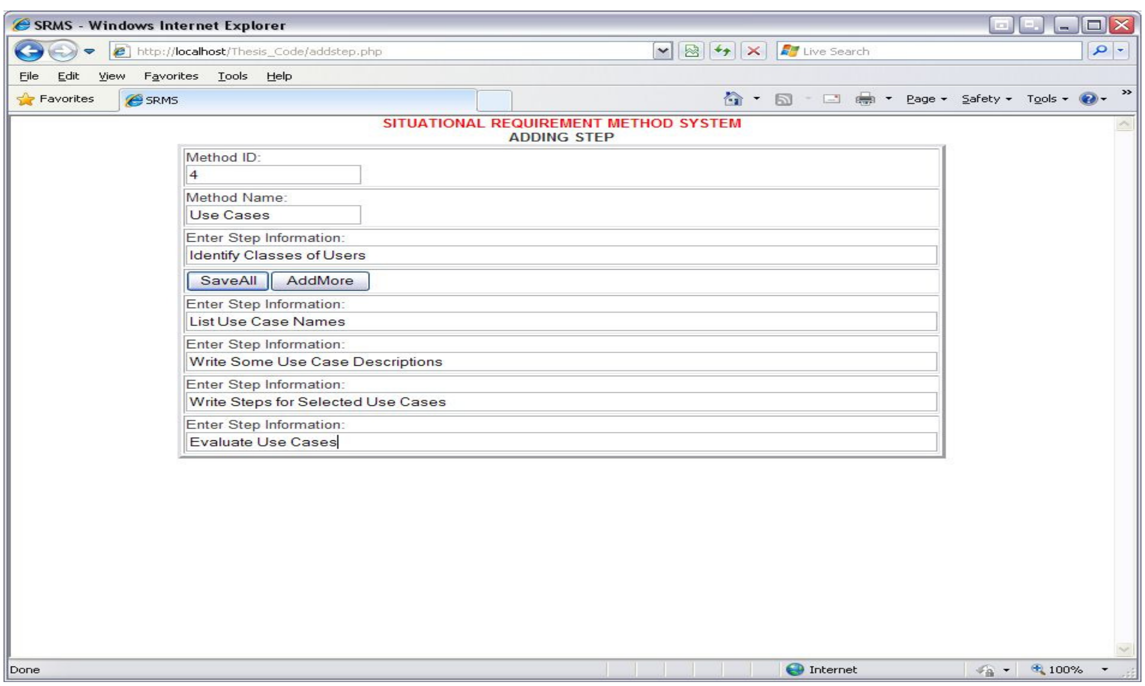


Figure 3.32 Add New Method Page (adding steps)

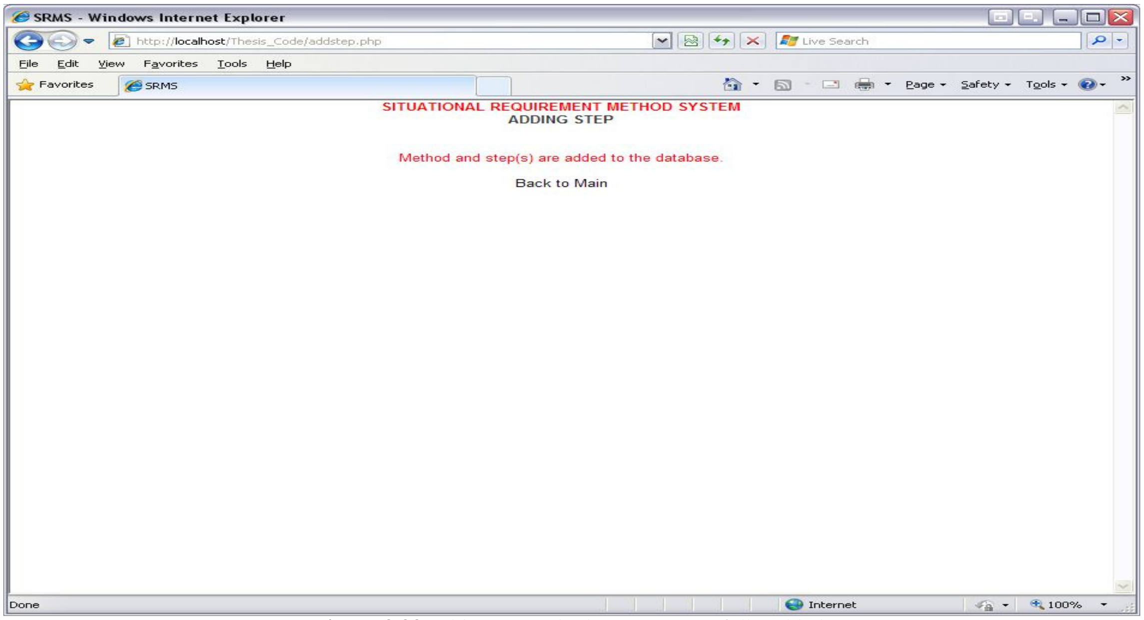


Figure 3.33 Add New Method Page (successfully added)

If user wants to add new method by extending an existing method stored in the system, Add New Method By Extending One functionality is used. When user clicks on this link, it first lists the all methods stored in the system as shown in Figure 3.34 then user selects the method he/she wants to use for extending. It also checks for if there is any method is clicked or not when user wants to start. If a method is not selected by the user, SRMS redirects user to main page.

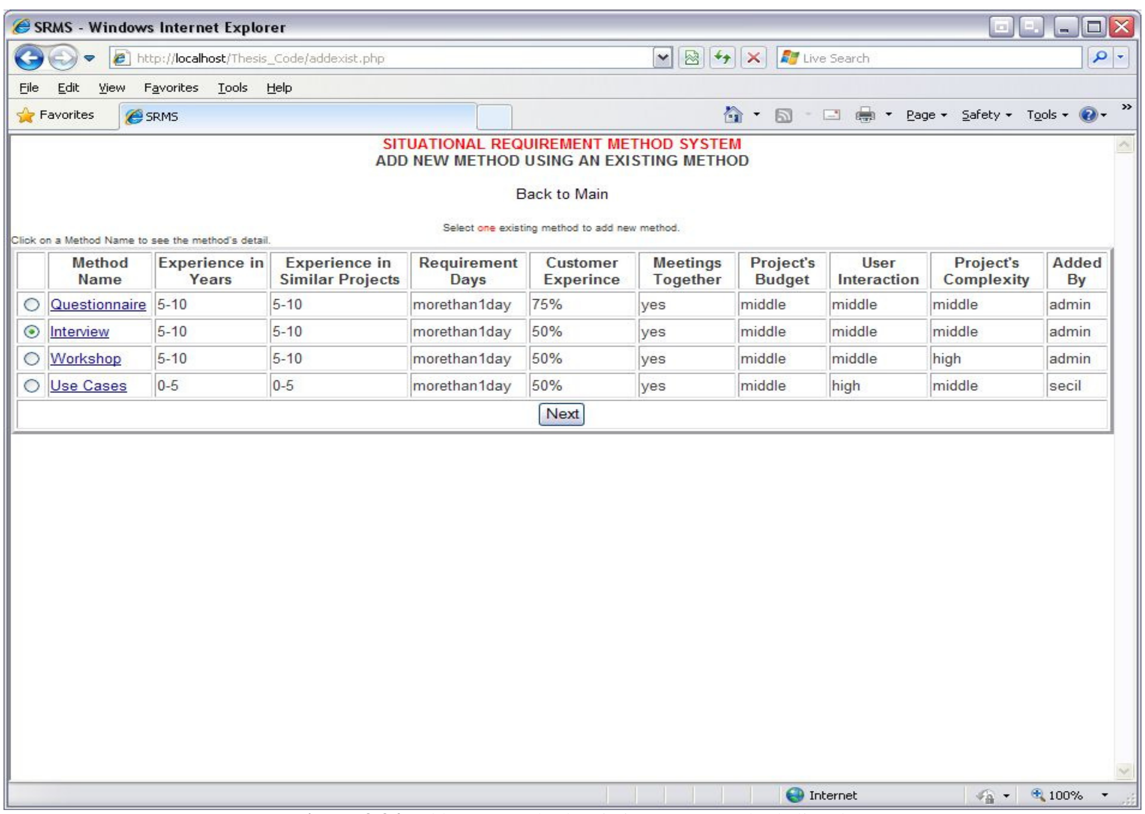


Figure 3.34 Add New Method Existing Page (methods listed)

After that, the selected method's criteria are displayed to the user. User is required to enter a new method name. Moreover, if it is required user can change the method's criteria as shown in Figure 3.35 then user clicks Add New Method then a new page is displayed to the user. Steps of the old method are listed one by one in this page. User can change/select steps for adding them to new method as shown in Figure 3.36. When user completes the changes and the selection and clicks Save, all the steps are added to the repository.

This capability provides user to reuse a method which is used before but it does not match with user's case by giving a new name, conditions and the steps which user had experienced. By the help of this, the knowledge heritages between the users of the system.

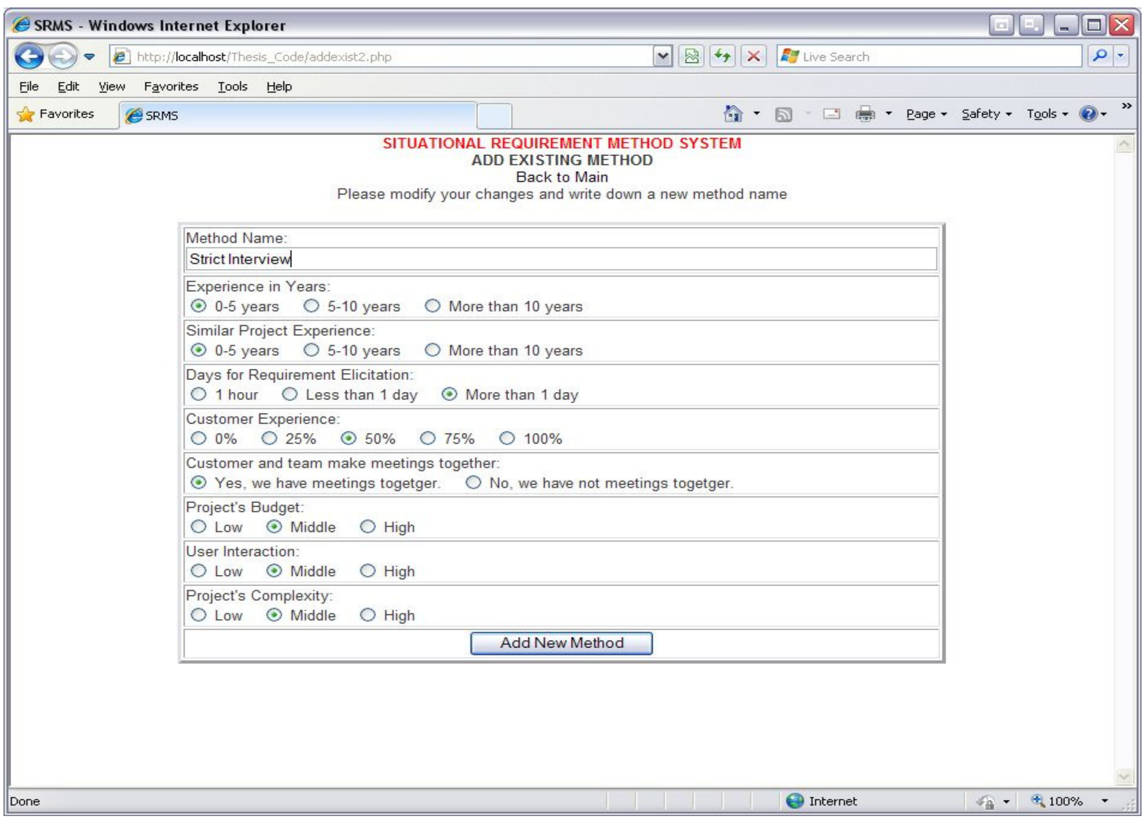


Figure 3.35 Add New Method Existing Page (criteria adding)

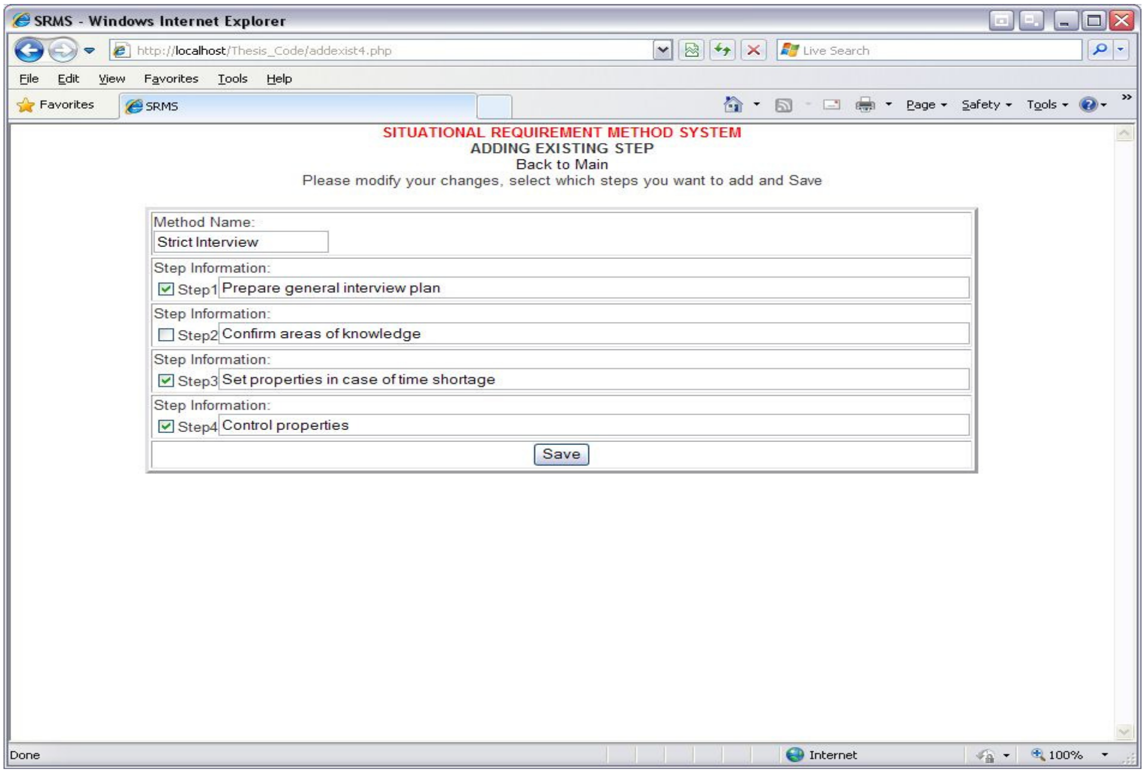


Figure 3.36 Add New Method Existing Page (step adding)

If user wants to add new method by combining two methods stored in the system, Add New Method By Combining Two functionality is used. When user clicks on this link, it first lists the all methods stored in the system as shown in Figure 3.37 then user selects two methods he/she wants to use for combining. It checks for if there is any method is clicked or not when user wants to start. If a method is not selected by the user, SRMS redirects user to main page. If one method is selected by the user, system informs user to select one more method. If more than two methods are selected by the user, system informs user to select two methods only.

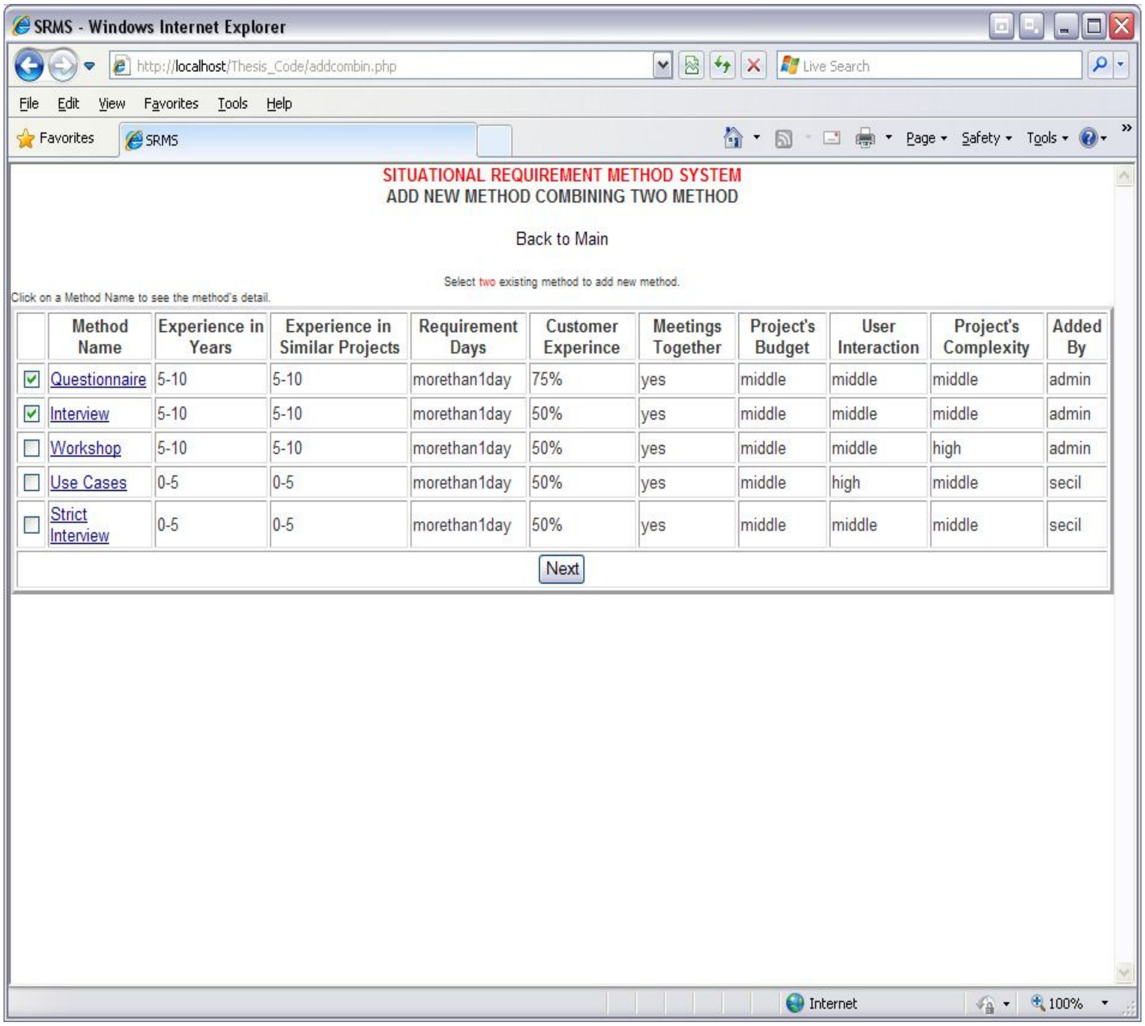


Figure 3.37 Add New Method By Combining Two Page (methods listed)

If user selects two methods from this page and clicks Next, a new page is displayed to the user with two method's criteria and a new blank criteria part. User can see the two method's criteria at the same page as shown in Figure 3.39. User is required to fill the blank criteria part and give name to new method. If one of the field is missed, system gives and error to fill the all fields as shown in Figure 3.38. If all the fields are filled as shown in Figure 3.40 and user clicks Add New Method, criteria are added to the repository with the new method's name.

Figure 3.38 Add New Method By Combining Two Error Message

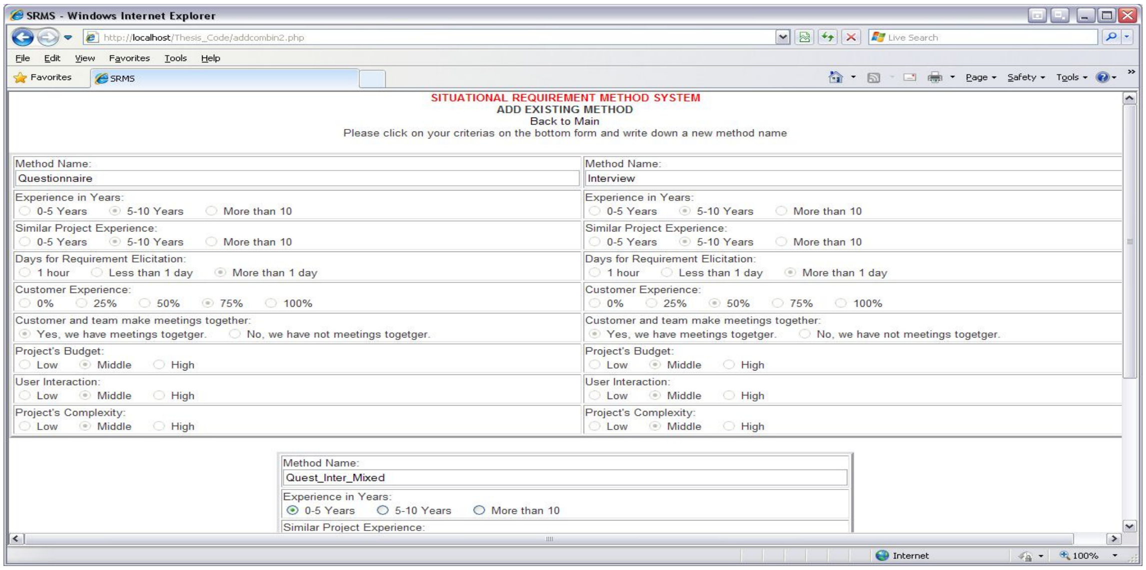


Figure 3.39 Add New Method By Combining Two Page (two methods info and blank one)

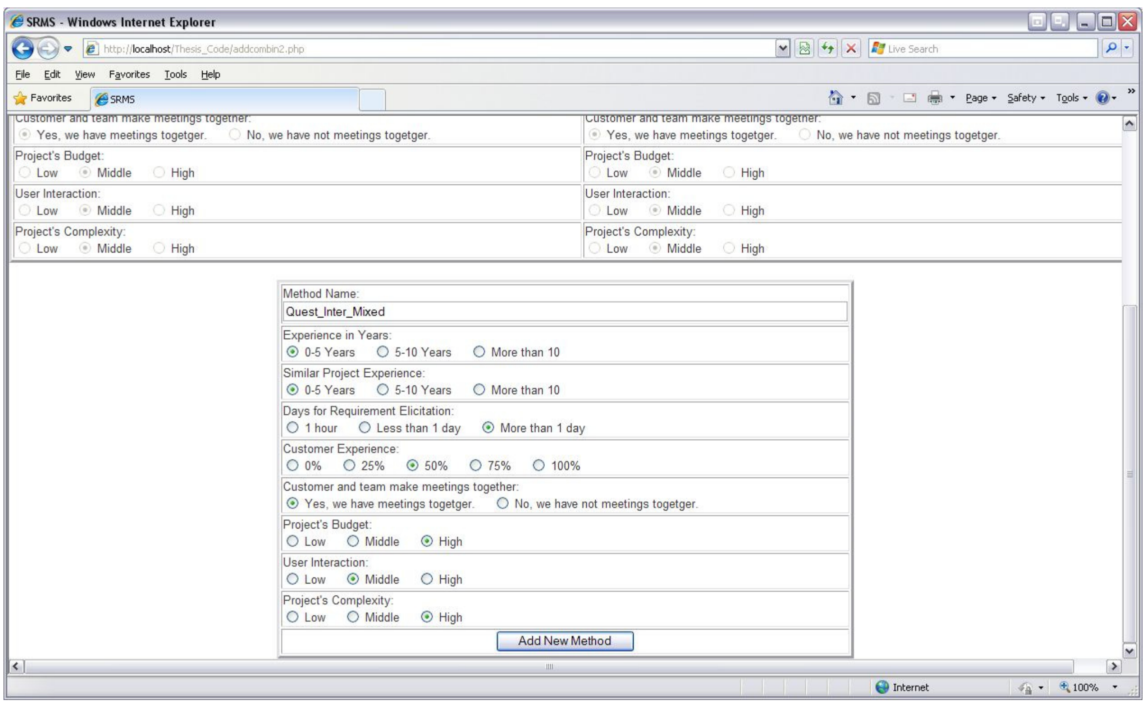


Figure 3.40 Add New Method By Combining Two Page (new method's info filled)

After criteria added to the repository, steps of methods' are displayed to the user. User is required to give numbers to the steps to add new method as shown in Figure 3.41. When user clicks Add Selected Steps, all selected steps are added to the system repository.

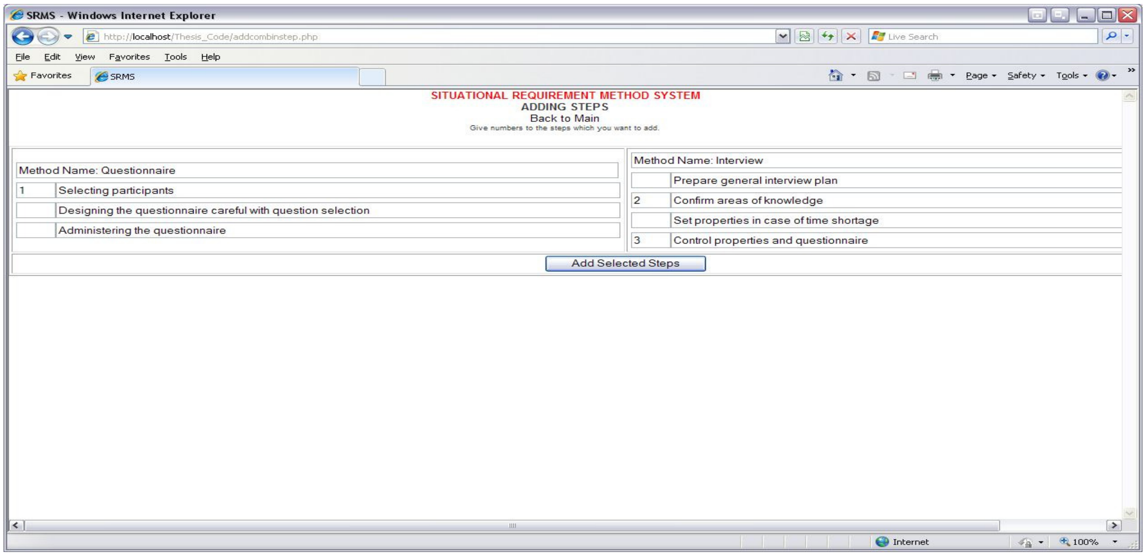


Figure 3.41 Add New Method By Combining Two Page (step adding)

Till now, adding method capabilities are explained. When user needs to update a method, Update Method module is used. User clicks on Update Method, then all the methods in the system are displayed to the user same as Add Existing Page's method listing. User is required to select a method, if a method is not selected by user system redirects user to main page. When a method is selected and Update button is clicked, a new page displayed with the criteria of selected method to the user. User changes the desired fields and clicks Update Method then a new page is displayed to the user for updating method's steps. User can add more steps by clicking Add Step button to the method or just only change the existing steps information and save by clicking Update and Save All button as shown in Figure 3.42.

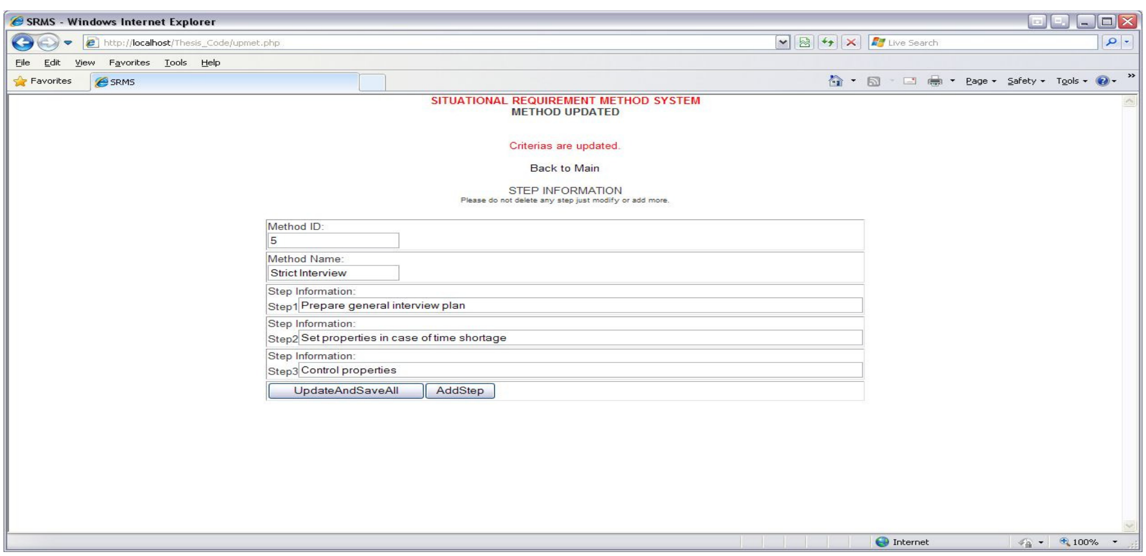


Figure 3.42 Update Method Page (updating steps)

If Add Step button is clicked by user, a new page is displayed to the user to add more steps or just add one step as shown in Figure 3.43.

In Update Method Page, the most important thing is; methods can only be updated by the owner of the method. If it is tried, system gives a message to user.

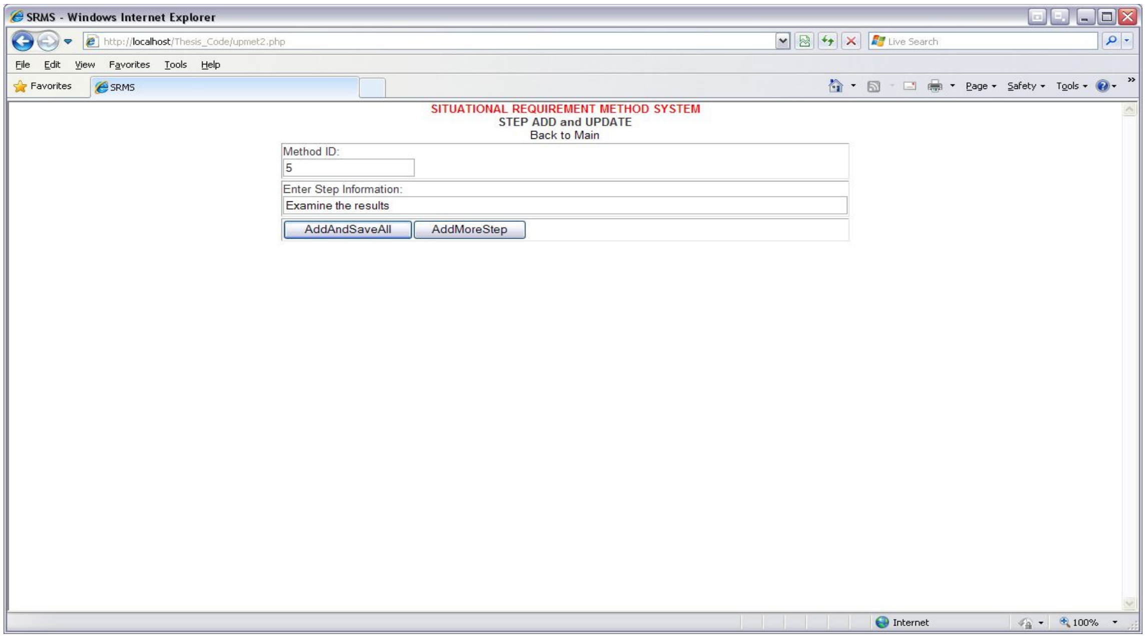


Figure 3.43 Update Method Page (adding steps)

When a method is added to the system, it is required to take a detailed report to see the all information at one page in report format. When user clicks Take Report, all methods is listed to the user. User selects one of the methods and clicks Take button, method's detailed information will be displayed to the user as shown in Figure 3.44.

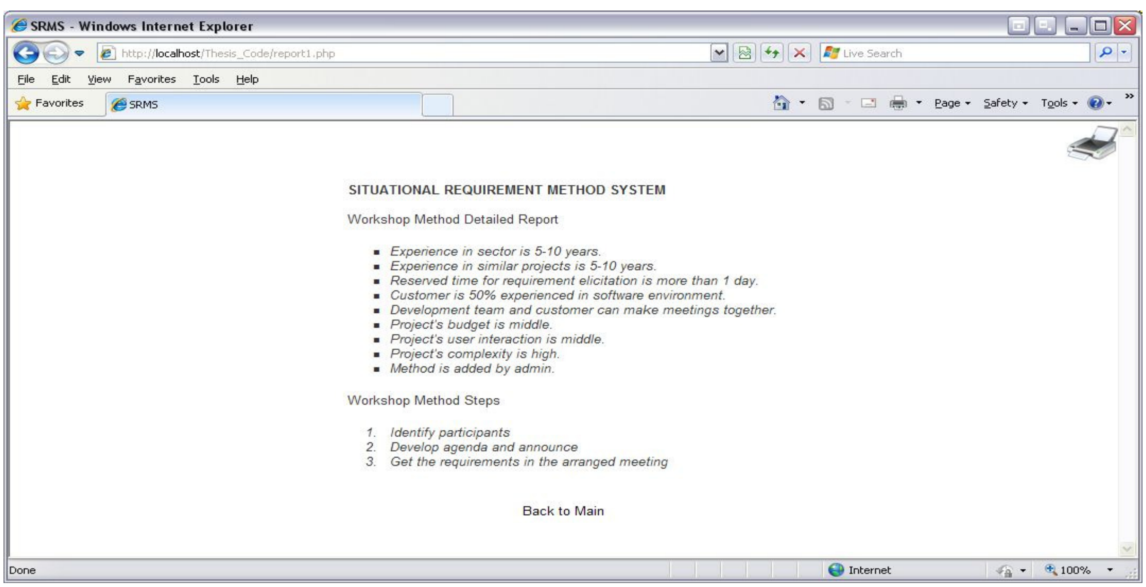


Figure 3.44 Take Report Page

It is also provided that the user can simply click on printer logo at the top of the page and take the print of the method report as in the Figure 3.45.

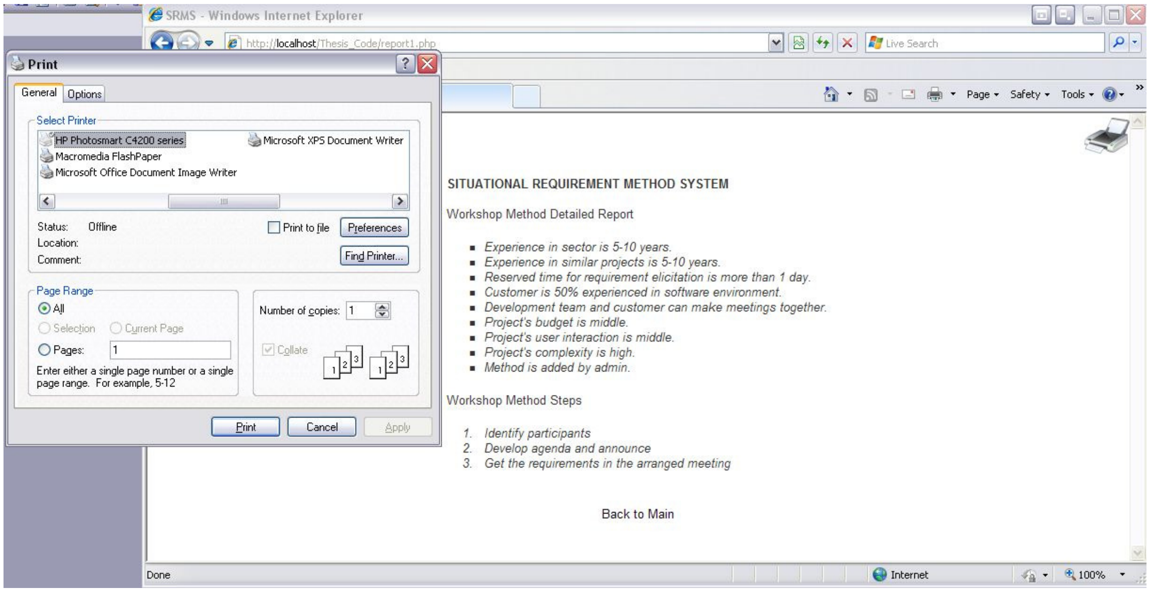


Figure 3.45 Take Report Page Print Capability

One of the capabilities of SRMS is showing statistical data about the system. When user clicks View Statistics, there are five criteria displayed to the user as shown in Figure 3.46.User selects the desired criteria to see the related statistics. Criteria are; based on user, based on method type, based on user and method type, based on view rate and based on usage for creation.

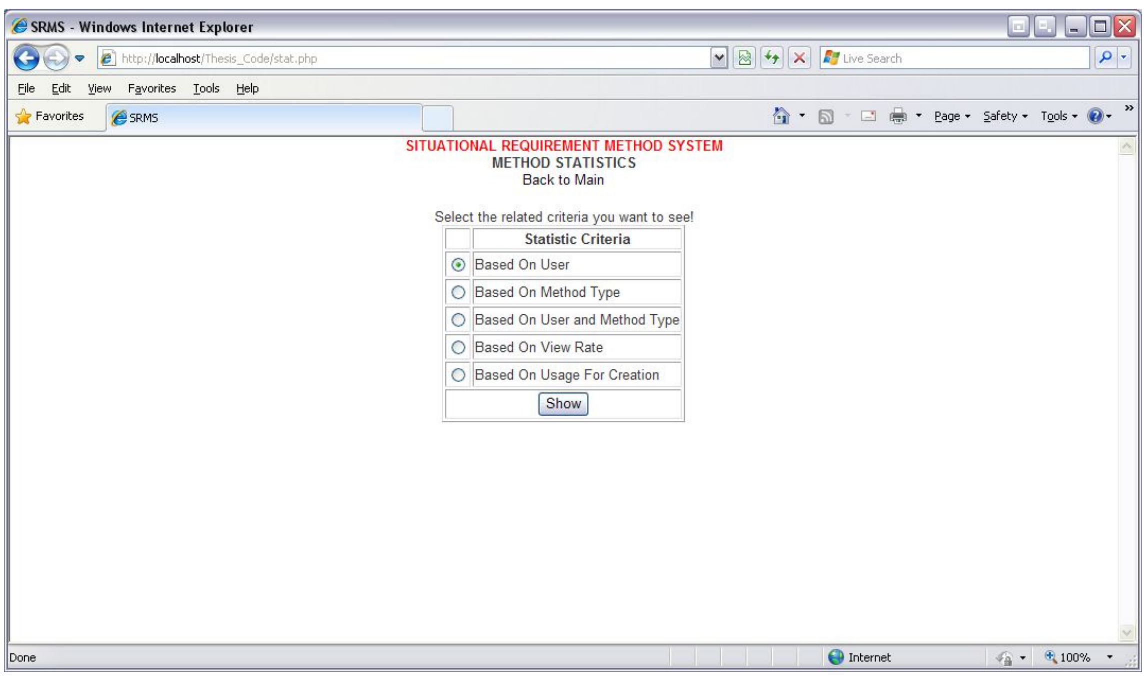


Figure 3.46 View Statistics Page

When user selects based on user and clicks Show button, related graph will be displayed to the user as shown in Figure 3.47.

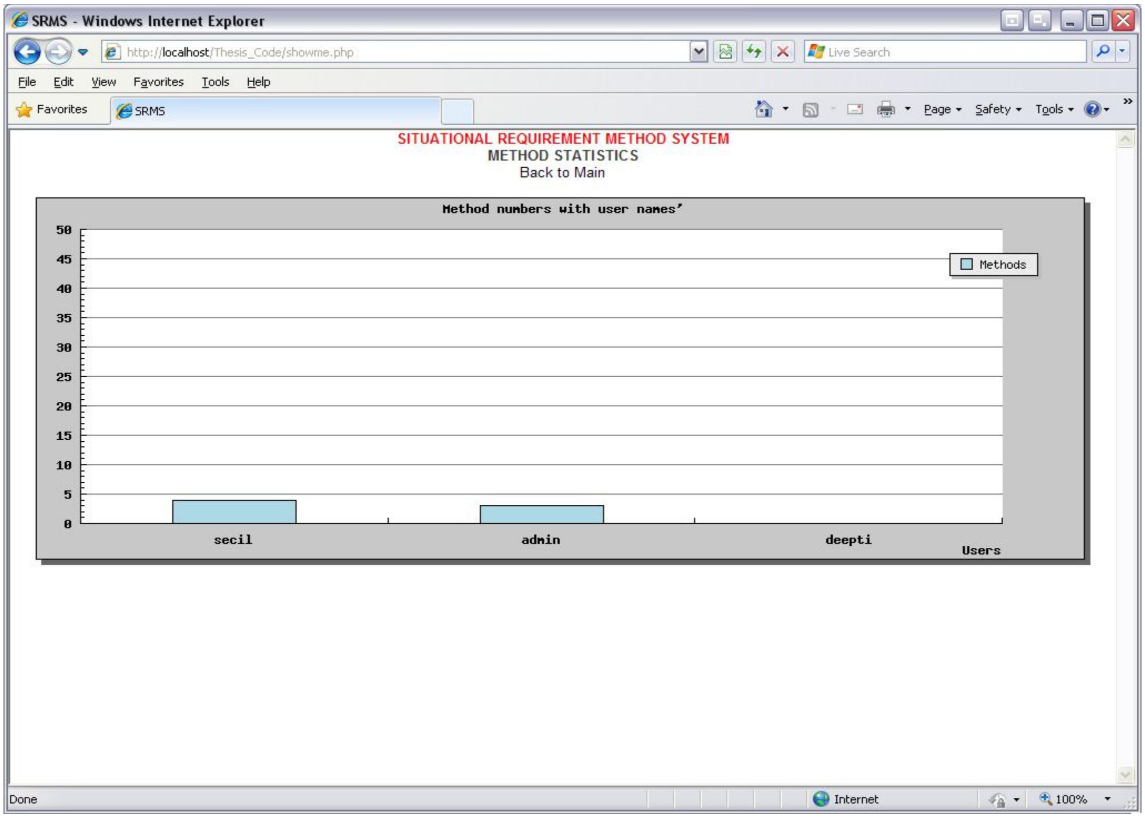


Figure 3.47 View Statistics Based On User

When user selects based on method type and clicks Show button, related graph will be displayed to the user as shown in Figure 3.48.

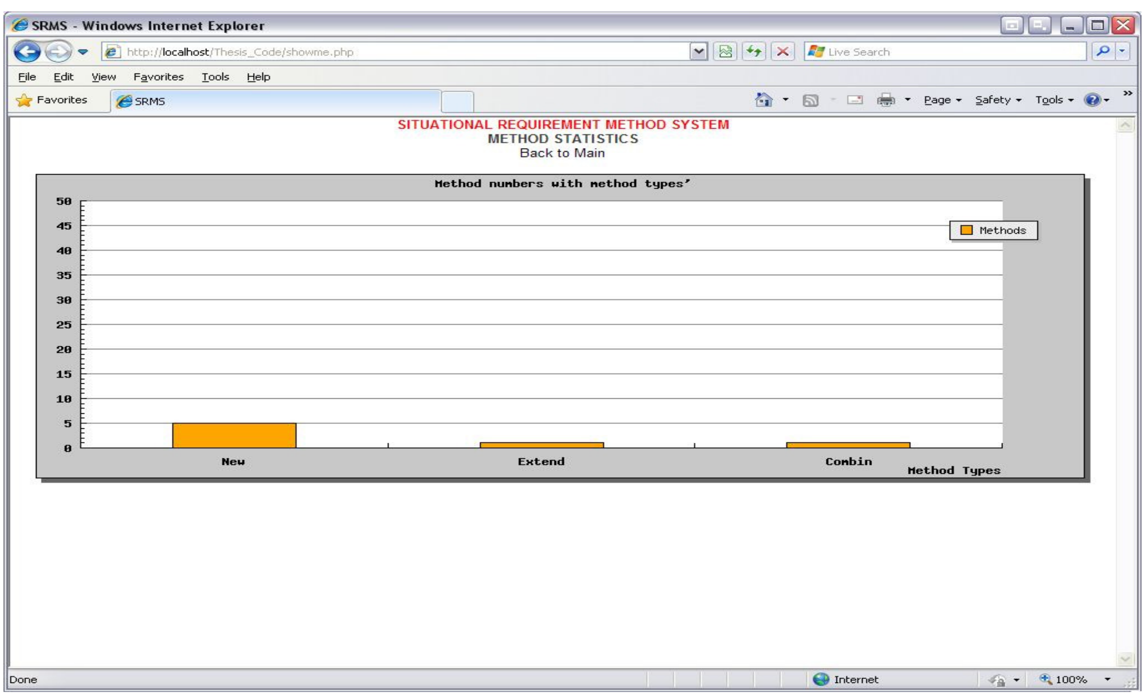


Figure 3.48 View Statistics Based On Method Type

When user selects based on user and method type, then clicks Show button, related graph will be displayed to the user as shown in Figure 3.49.

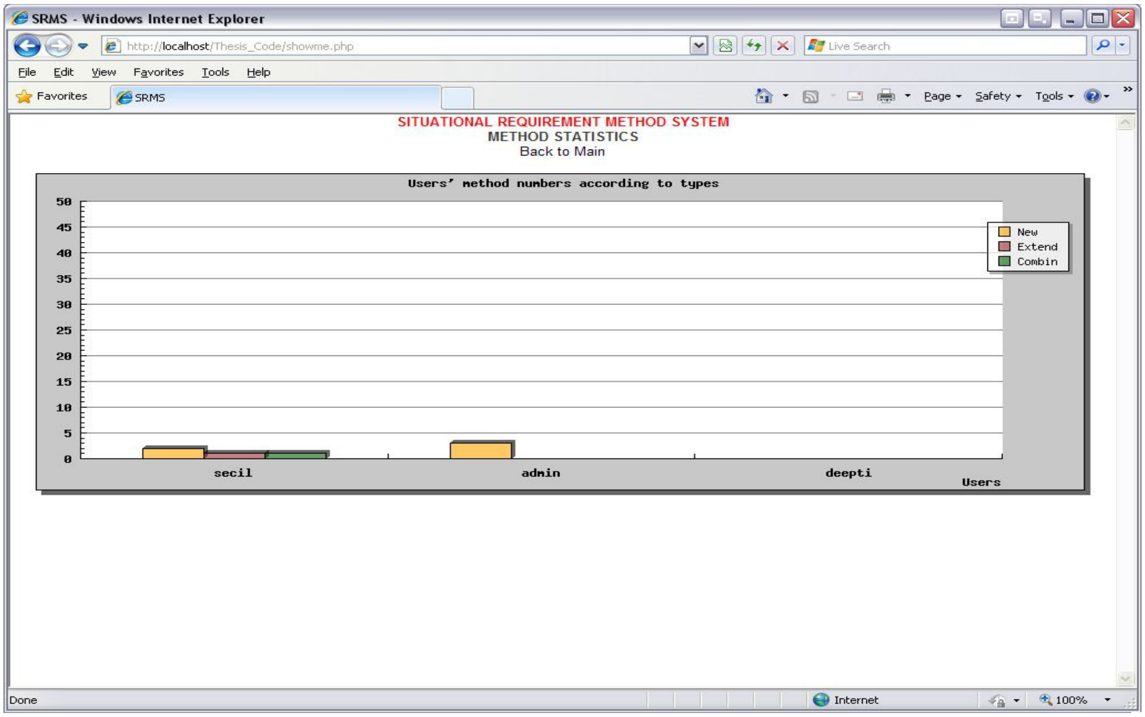


Figure 3.49 View Statistics Based On User and Method Type

When user selects based on view rate, then clicks Show button, related graph will be displayed to the user as shown in Figure 3.50.

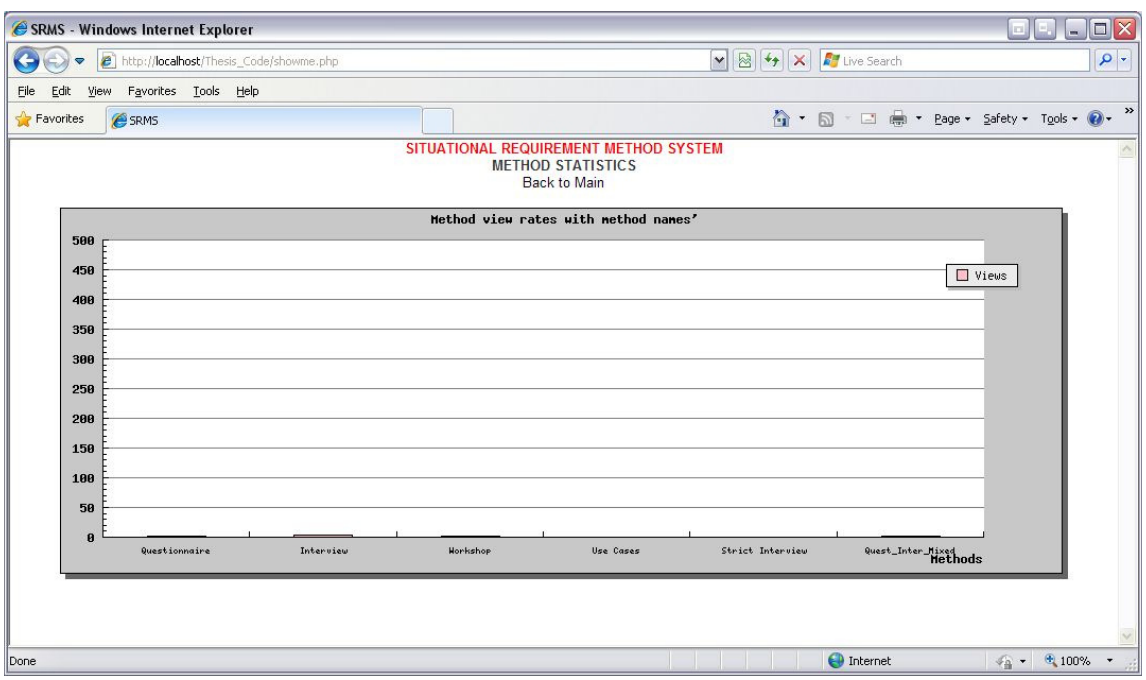


Figure 3.50 View Statistics Based On View Rate

When user selects based on usage for creation, then clicks Show button, related graph will be displayed to the user as shown in Figure 3.51.

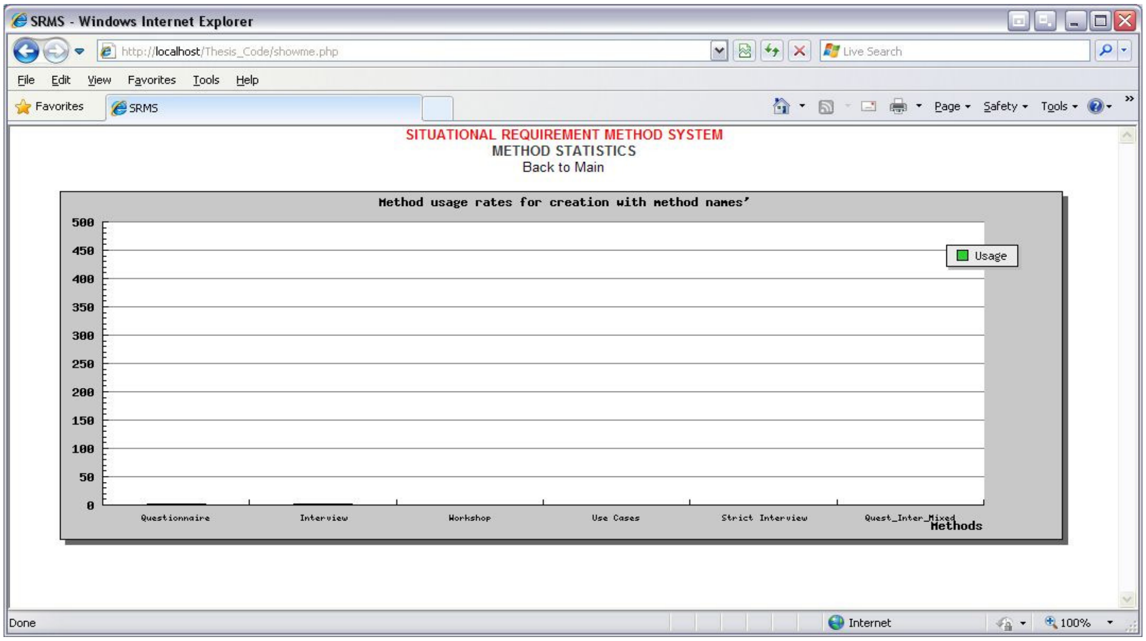


Figure 3.51 View Statistics Based On Usage for Creation

When user clicks on Needs Help, SRMS features and help topics are listed as shown in Figure 3.52 and Figure 3.53.

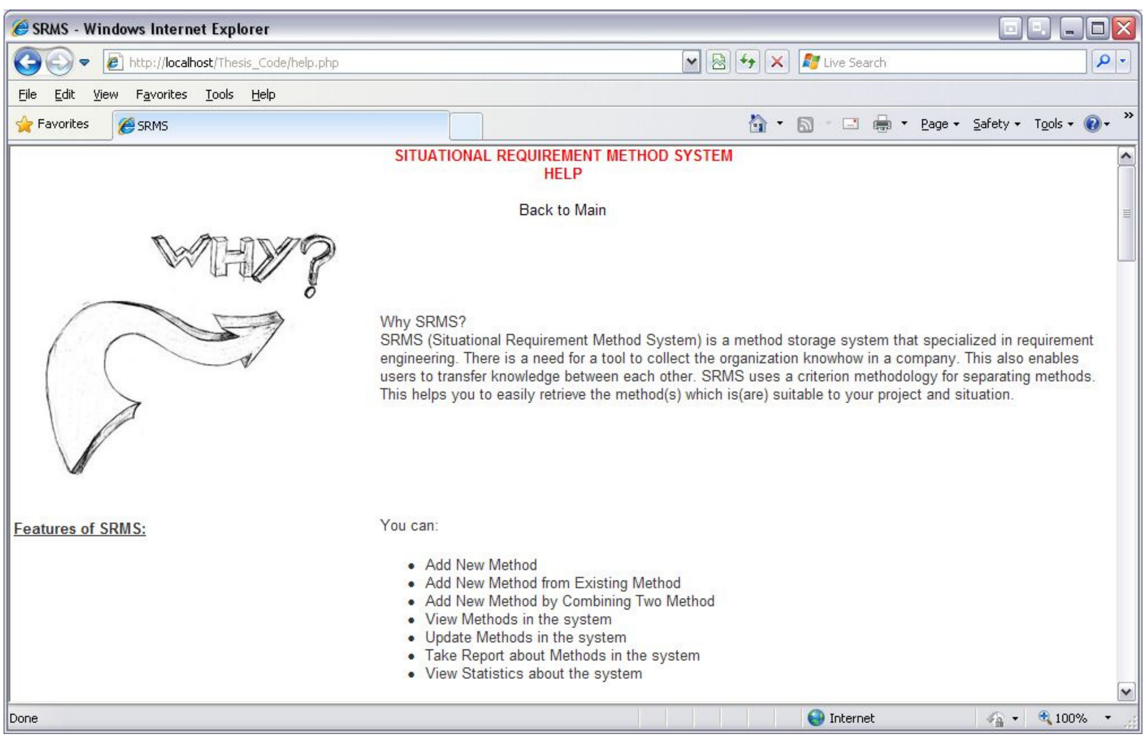


Figure 3.52 Need Help Page (features listed)

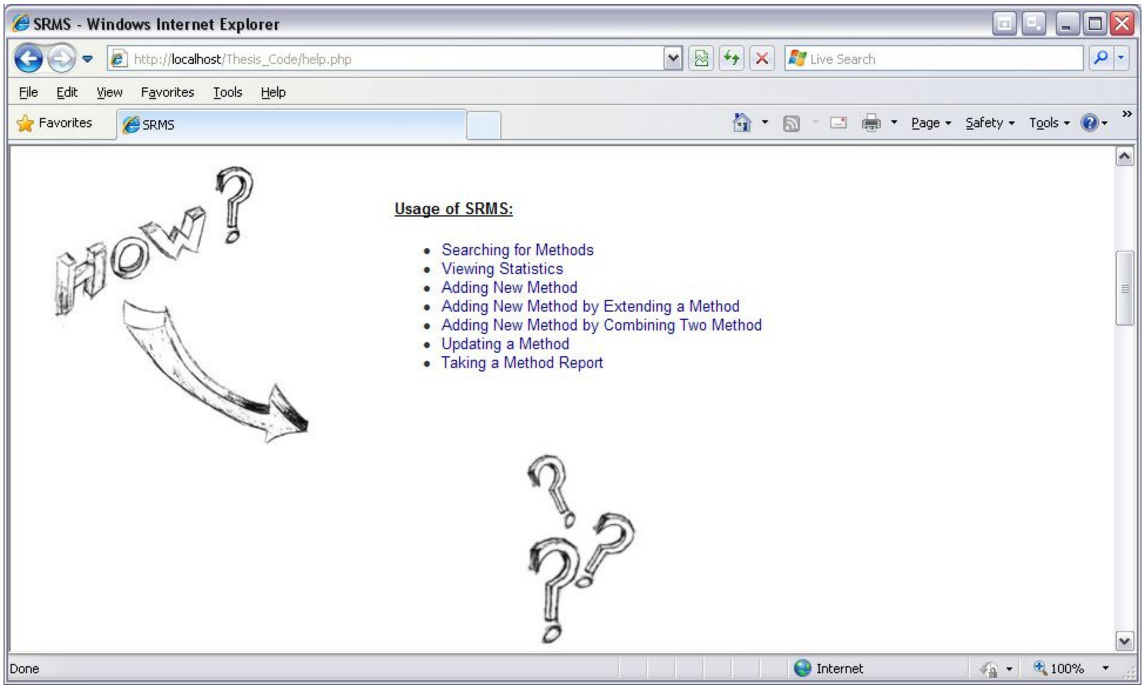


Figure 3.53 Need Help Page (feature links listed)

When user clicks on a topic such as Adding New Method, related help item is displayed to the user as shown in Figure 3.54 so user can easily learn the usage of the system.

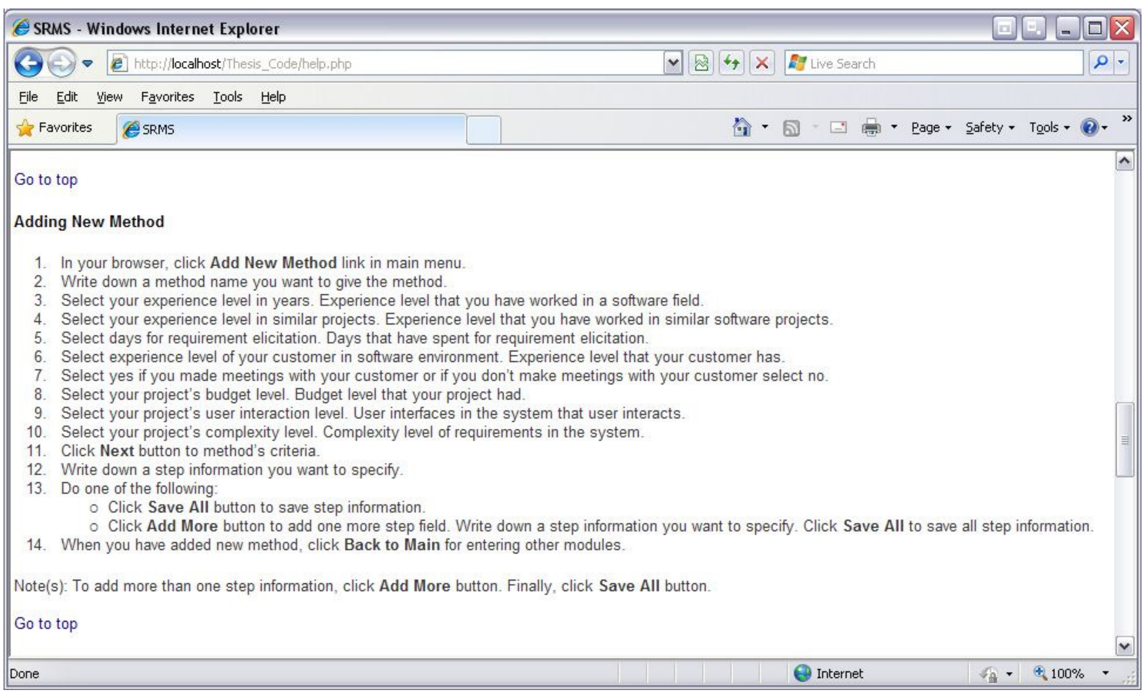


Figure 3.54 Need Help Page (items listed)

SRMS typical user capabilities are described till now. There is another actor of the system called admin. Admin can do the same capabilities as typical user and four more capabilities extra. Admin can delete a method, add a new user to the system, set new password for a user and delete a user. When admin opens his/her browser and the system address is entered, system welcomes the admin same as typical user. When admin clicks on the system logo, admin redirected to login page. Admin enters the username and password, clicks Login, admin redirected to main page shown in Figure 3.55.

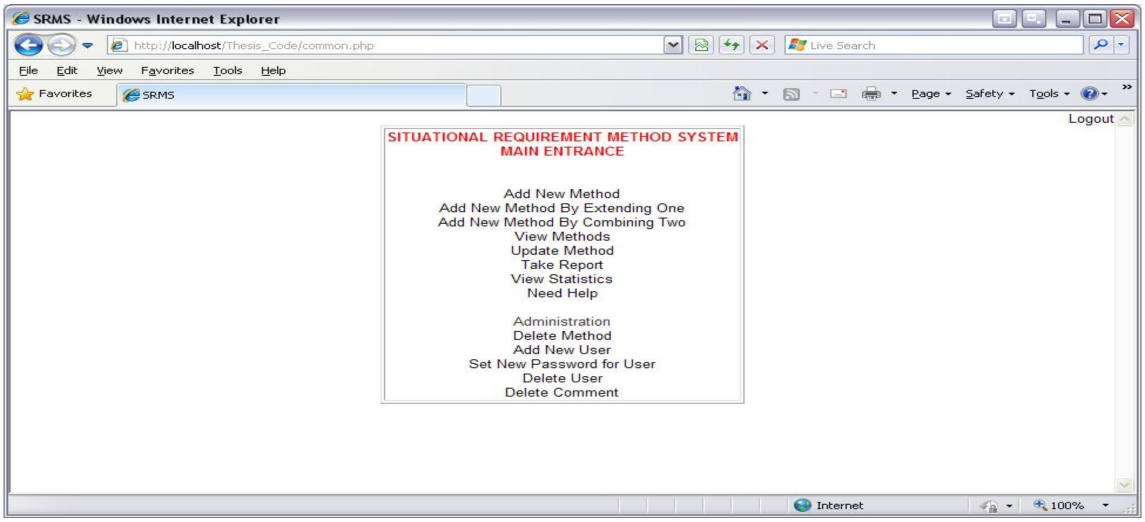
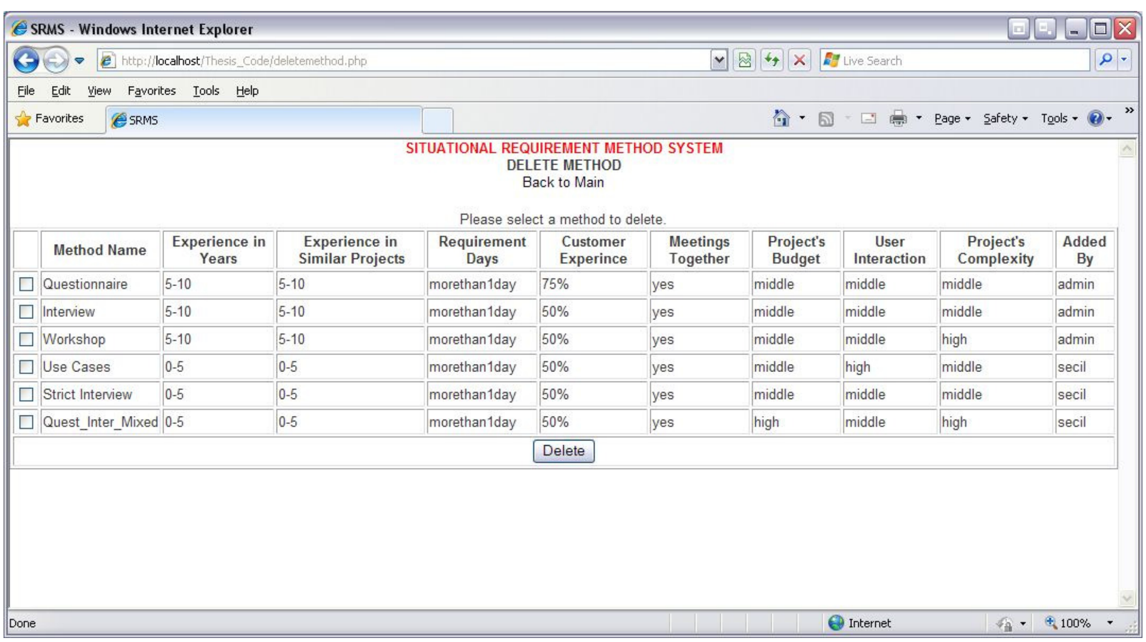


Figure 3.55 Main Page for Admin

Typical user does not have right to delete a method from the system. This feature is placed in the system for preventing knowledge loss. When a method deletion request comes from a typical user, admin clicks on Delete Method on the main page, then all the methods stored in the system displayed to the admin as shown in Figure 3.56.



	Method Name	Experience in Years	Experience in Similar Projects	Requirement Days	Customer Experince	Meetings Together	Project's Budget	User Interaction	Project's Complexity	Added By
☐	Questionnaire	5-10	5-10	morethan1day	75%	yes	middle	middle	middle	admin
☐	Interview	5-10	5-10	morethan1day	50%	yes	middle	middle	middle	admin
☐	Workshop	5-10	5-10	morethan1day	50%	yes	middle	middle	high	admin
☐	Use Cases	0-5	0-5	morethan1day	50%	yes	middle	high	middle	secil
☐	Strict Interview	0-5	0-5	morethan1day	50%	yes	middle	middle	middle	secil
☐	Quest_Inter_Mixed	0-5	0-5	morethan1day	50%	yes	high	middle	high	secil

Figure 3.56 Delete Method Page

When requested method is selected and Delete button is clicked, method is deleted and a message is displayed to the admin for successful deletion. User adding requests are handled by admin. When admin wants to add a new user to the system, admin can use Add New User module, a new page is displayed to the admin for giving username and password to the user as shown in Figure 3.57.

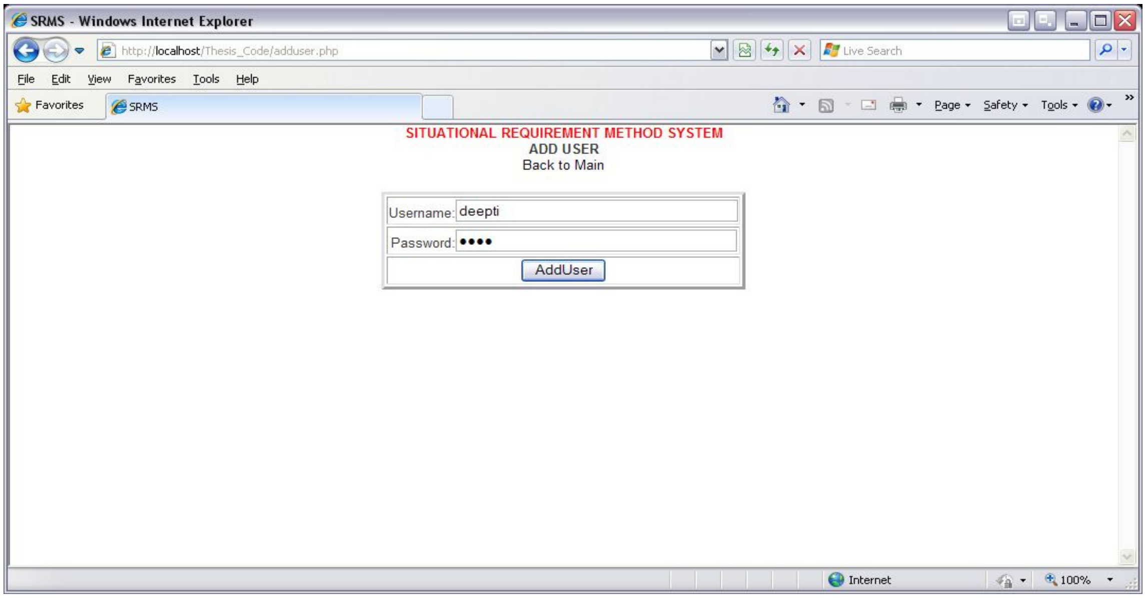


Figure 3.57 Add New User Page

When admin completes the entrance of user information and clicks Add User, user is added to the system and successful addition message is displayed to the admin. When a user forgets his/her password, Set New Password for User module can be used for this. When admin clicks on this module, all the users in the system is displayed to the admin as shown in Figure 3.58.

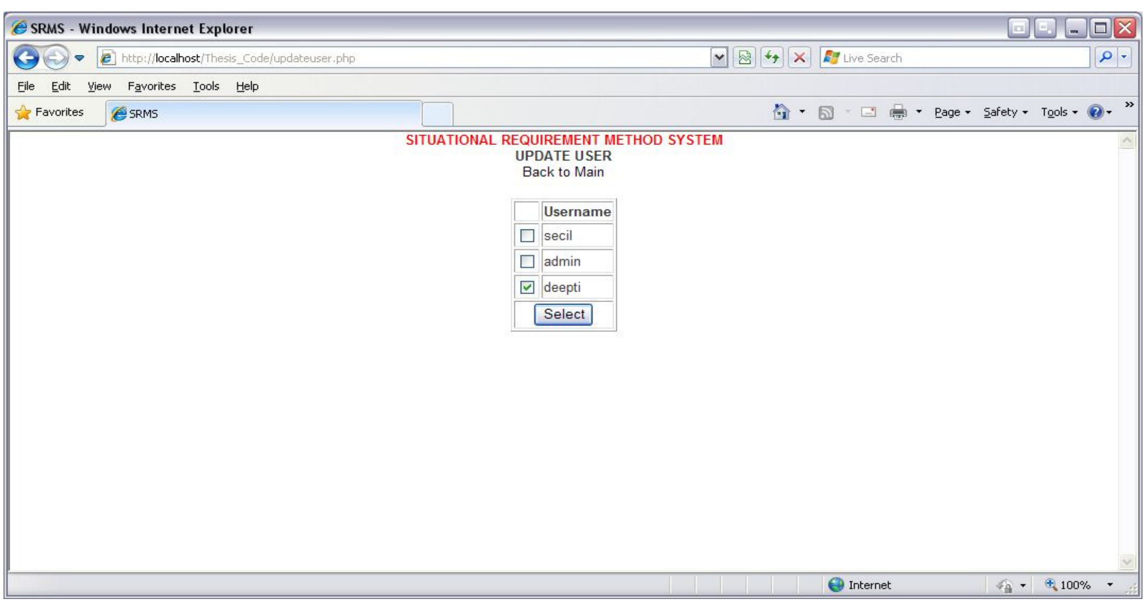


Figure 3.58 Set New Password For User Page(users listed)

After that, new page is displayed to the admin for entering a new password for user. When a new password is entered and admin clicks Set as shown in Figure 3.59, new password is updated and successfully addition message is displayed to the user.

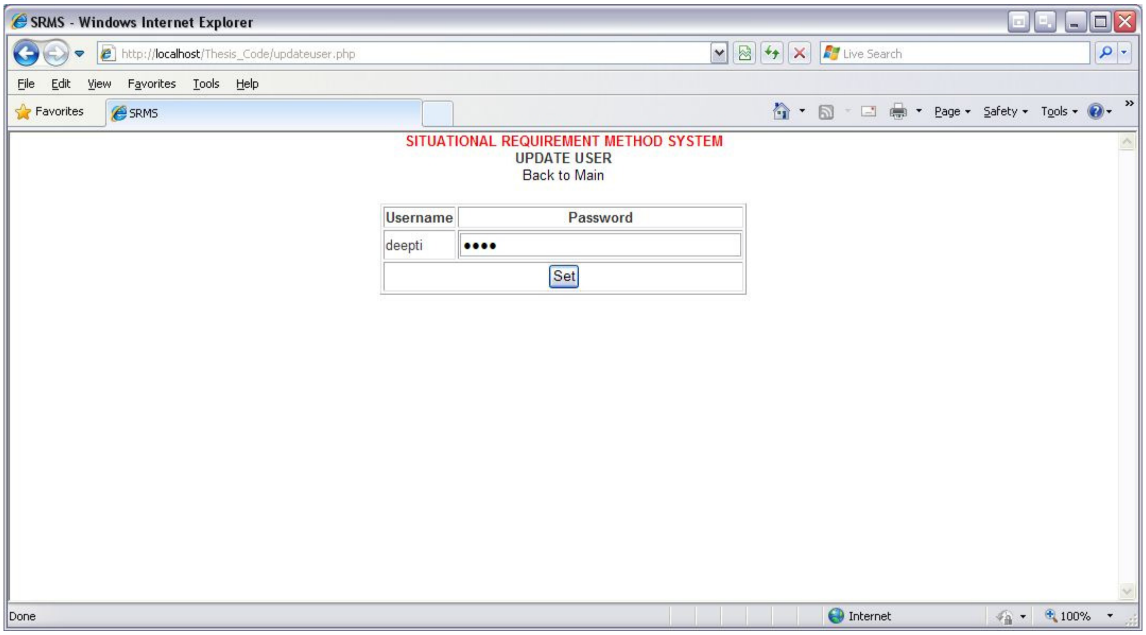


Figure 3.59 Set New Password For User Page(user password updating)

Another capability of admin is deleting a user. When admin clicks on Delete User, all the users in the system is displayed to the admin as shown in Figure 3.60. Admin selects a user from the list and clicks Select, user is deleted from the system and successfully deletion message is displayed to the admin.

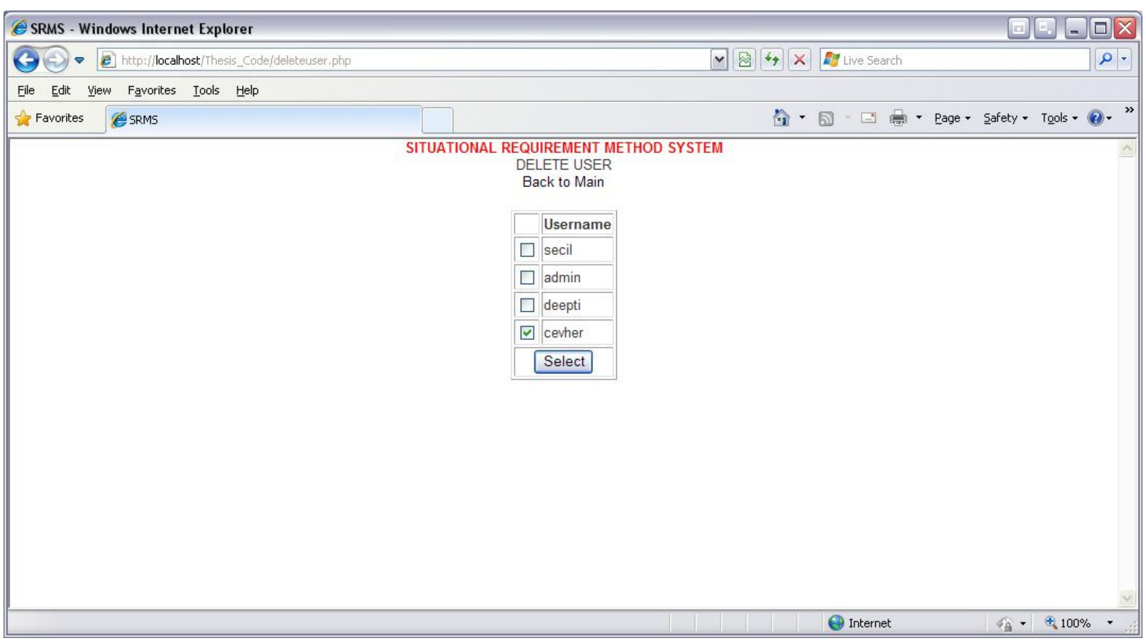
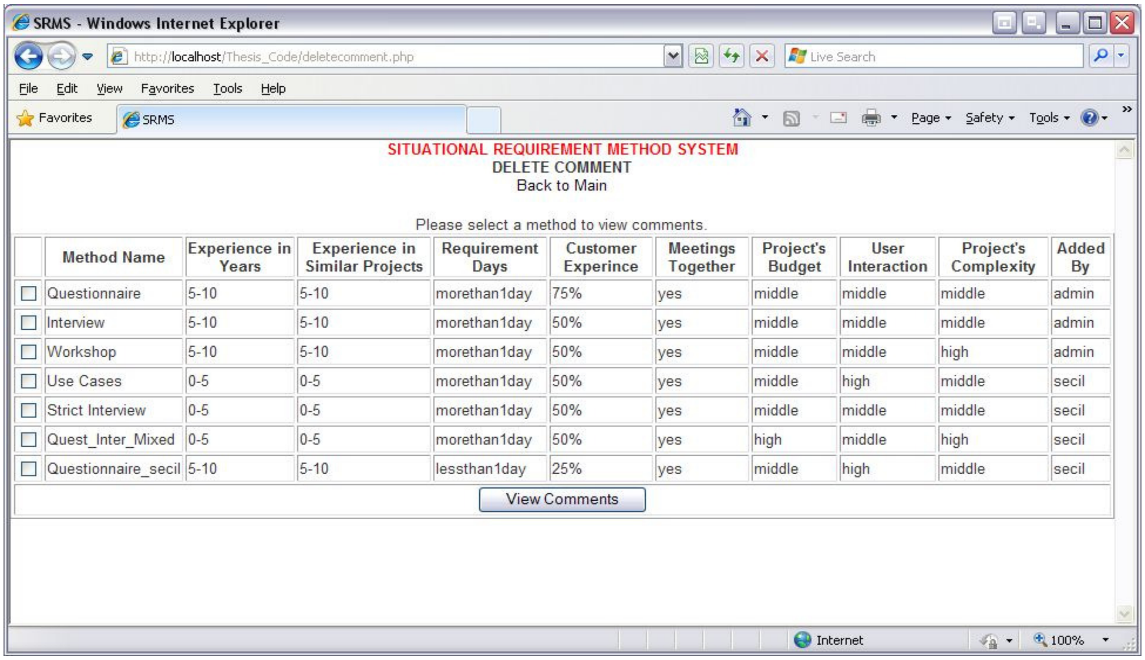


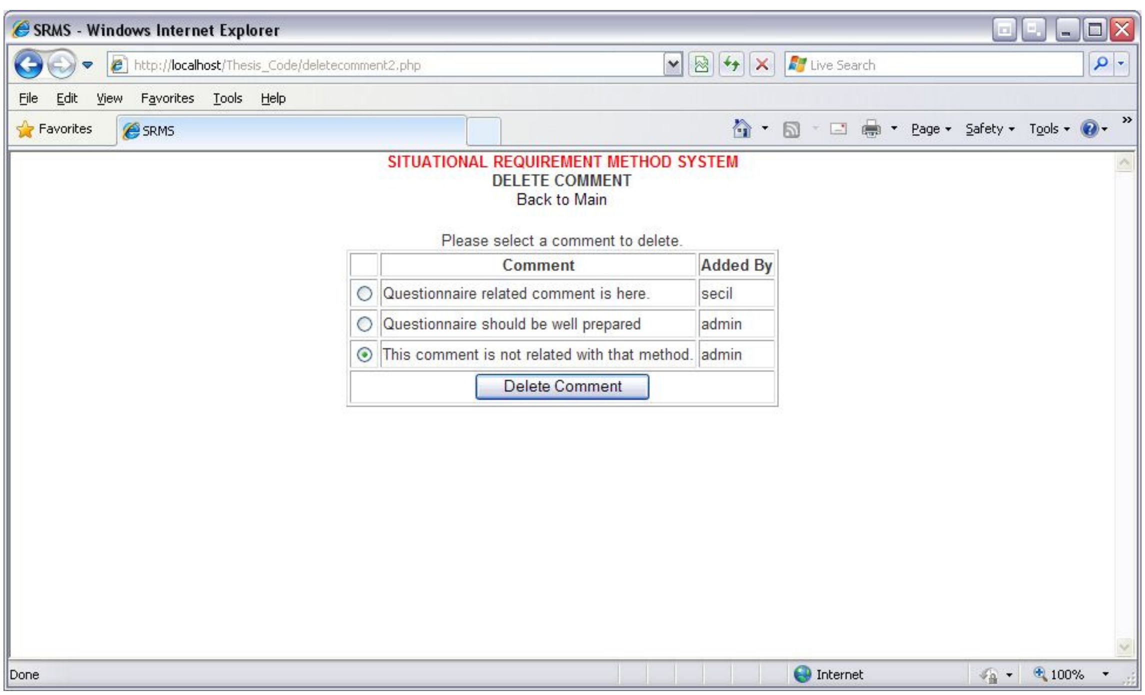
Figure 3.60 Delete User Page

Last capability of admin is deleting a comment. When admin clicks on Delete Comment, all the methods in the system is displayed to the admin as shown in Figure 3.61. Admin selects a method such as Questionnaire from the list and clicks View Comments, all the comments of this method is displayed to the admin as shown in Figure 3.62. After that, admin selects a comment and clicks Delete, the comment is deleted from the system and successfully deletion message is displayed to the admin.



	Method Name	Experience in Years	Experience in Similar Projects	Requirement Days	Customer Experince	Meetings Together	Project's Budget	User Interaction	Project's Complexity	Added By
☐	Questionnaire	5-10	5-10	morethan1day	75%	yes	middle	middle	middle	admin
☐	Interview	5-10	5-10	morethan1day	50%	yes	middle	middle	middle	admin
☐	Workshop	5-10	5-10	morethan1day	50%	yes	middle	middle	high	admin
☐	Use Cases	0-5	0-5	morethan1day	50%	yes	middle	high	middle	secil
☐	Strict Interview	0-5	0-5	morethan1day	50%	yes	middle	middle	middle	secil
☐	Quest_Inter_Mixed	0-5	0-5	morethan1day	50%	yes	high	middle	high	secil
☐	Questionnaire_secil	5-10	5-10	lessthan1day	25%	yes	middle	high	middle	secil

Figure 3.61 Delete Comment Page (methods listed)



	Comment	Added By
○	Questionnaire related comment is here.	secil
○	Questionnaire should be well prepared	admin
◉	This comment is not related with that method.	admin

Figure 3.62 Delete Comment Page (comments listed)

3.5 SRMS Tool Comparison with Existing Tools

Although there exist several studies in the literature which explores SME, but there are very few tool prototypes and tools to use for this issue. All the environments are coming from research areas, a few support is coming from the industry. This is the reason of why a little contribution has been added to this methodology. Most of the tools existing in this field, lack of documentation so they are not preferred by the people who are working in the industry.

CAME environments that are existing in the literature are; Decamerone, Mentor [Plihon, 96], Meret [Heym&Osterle, 92], Meru, MetaEdit+, MethodBase and Method Editor [Niknafs&Ramsin, 08]. They are compared based on SME approach, method textual language, method meta model, origin, process oriented and product oriented. Table 3 shows the complete comparison between CAME tools.

Decamerone, Meru, MetaEdit+ and Method Editor are assembly-based which is specifying method requirements, selecting method fragments, and assembling them into a method. Moreover, Meret tool represents a comprehensive methodology representation model. MethodBase aim is to facilitate method customization rather than assembly-based SME. Mentor tool is both assembly-based and paradigm-based. On the other hand, SRMS tool provides method customization from existing methods, by combining two methods according to their context and also constructing a new method from beginning. This approach provides SRMS becoming more flexible for method construction. It also provides more user (experienced engineer) relation with the creation of method so the experience is directly stored on the repository with all characteristics.

Decamerone has own ontology called Methodology Data Model (MDM) which consist of the basic concepts of information system development products and the associations between them and it has a language for representation of method fragments called method engineering language (MEL). As desribed before, Meret tool's representation model uses a semantic data-model called ASDM. ASDM provides a powerful means for modeling objects and their interrelationships. Mentor uses the nature contextual approach to describe method fragments. It uses the nature so it has not got any language but it has contexts that are pairs of the form <situation, decision>. Meru is based on a meta-model called MVM. In this methodology, method concepts which are called things are partitioned to links, constraints and product elements. Meru has own language called method requirements specification language (MRSL). MetaEdit+ uses the GOPPRR conceptual data model as its method specification language. GOPPRR composed of graph, object, port, property, relationship and role. MethodBase's data model is divided into product and process parts. It is using Object Z to represent method fragments. Method Editor uses UML for meta-modeling technique to represent method fragments. Class diagrams are used for the specification of product fragments, while process fragments are described by means of activity diagrams. Despite these techniques, SRMS uses a criterion methodology for storing and representing methods. There is no need to learn a language or learn modeling a method. SRMS stores methods in natural language; this feature gives flexibility to user.

Table 3. Comparison between SRMS and existing tools

CAME Environments	SME Approach	Method Textual Language	Method Meta Model	Origin	Availability	Process Oriented	Product Oriented	Year of Introduction
Decamerone	Assembly-based	MEL	MDM	Research	Web based	✓	-	1995
Mentor	Assembly-based, Paradigm-based	-	NATURE	Research	Not web based	-	✓	1996
Meret	Method Customization	Methodology Rep. Model	ASDM	Research	Not web based	✓	-	1992
Meru	Assembly-based	MRSL	MVM	Research	Not web based	✓	-	2001
MetaEdit+	Assembly-based	-	GOPPRR	Research and Commercial	Cross-platform	✓	-	1994
MethodBase	Method Customization	Object Z	-	Research	-	✓	-	1992
Method Editor	Assembly-based	MEL	UML	Research	Not web based	✓	-	2003
SRMS	Method Customization	English	Complete Method, Criterion used	Research	Web based, cross-platform	✓	-	2009

Based on origin, Decamerone, Mentor, Meret, Meru, MetaEdit+, Method Editor and SRMS is developed for research purposes. Only MetaEdit+'s commercial version has also released. Mentor, Meret, Meru and Method Editor are not web based tools. Decamerone and SRMS are web based tools. SRMS and MetaEdit+ are cross-platform tools. Since SRMS is a web based tool, the project members can be in different physical locations still they can easily access it.

Depending on the way CAME environments facilitate the enactment of the method engineering process; they are grouped as product-oriented and process-oriented. The tools that are providing less support for method's process model and its enactment and focuses on product related issues are classified as product-oriented. Process-oriented CAME environments focuses on the process related issues and supports the enactment of the process model. Decamerone, Meret, Meru, MetaEdit+, MethodBase, Method Editor and SRMS are basically process oriented. On the other hand, Mentor is the only product-oriented environment.

Odell and Harmsen suggest CAME tool should support the seven features; definition and evaluation of contingency rules and factors, storage of method fragments, retrieval and composition of method fragments, validation and verification of the generated methodology, adaptation of the generated methodology, integration with a meta-CASE tool and interface with a methodbase [Odell, 95], [Harmsen, 94]. Tools that described above are carrying these seven features at least partially. SRMS is also carrying these features. As the comparison shows, SRMS is the easiest tool for method storage and management.

3.6 SRMS Tool Validation

A questionnaire is used to study the effectiveness of proposed approach and ease of usage of tool. Questionnaire contains twelve questions shown in Figure 3.63. Moreover, SRMS was published in World Wide Web to make it accessible to the respondents of the questionnaire. User accounts were opened for each of the respondents for accessing the system so that respondents can easily access and evaluate the system. After that, copies of questionnaire were distributed to 14 people who are mainly working in large software companies (totally 10) and are more likely to have experience with these types of systems. Their experience level and the positions are sufficient for getting high quality feedback. Questionnaire results were collected from the respondents and Table 4 shows the response rates and options.

SRMS

SITUATIONAL REQUIREMENT METHOD SYSTEM QUESTIONNAIRE

This questionnaire is about Situational Requirement Method System. This system is created for a Master of Science Thesis to demonstrate the stated theory. Please feel free to give your ideas. The information you provide will be treated in confidence. Thank you for your contribution. ' fields are optional.

YOUR CONTACT INFORMATION	
Name':	Surname':
E-Mail':	Phone':
Expertise of Work':	Today's date:

SYSTEM RELATED QUESTIONS

1. Have you ever used this type of a method engineering system?

Yes ☐	No ☐

2. Do you have any lessons learned database?

Yes ☐	No ☐

3. If you answered Yes to Question 2, please state your type of expertise in lessons learned databases and select your choice according to your experience.
SRMS is better than storing a lessons learned database?

Expertise:

Strongly Agree ☐	Agree ☐	Disagree ☐	Strongly Disagree ☐

If your answer is Disagree or Strongly Disagree; please state your reason in brief.

4. We prefer to use SRMS in our intranet system.

Strongly Agree ☐	Agree ☐	Disagree ☐	Strongly Disagree ☐

5. SRMS is easy to use.

Strongly Agree ☐	Agree ☐	Disagree ☐	Strongly Disagree ☐

SRMS

6. I found SRMS's method storage principle useful.

Strongly Agree ☐	Agree ☐	Disagree ☐	Strongly Disagree ☐

7. Which module do you found very useful? (You can select more than one)

- ☐ Add New Method
- ☐ Add New Method By Extending One
- ☐ Add New Method By Combining Two
- ☐ View Methods
- ☐ Update Method
- ☐ Take Report
- ☐ View Statistics

8. Do you have any suggestion for adding a new module to the system? Please explain.

9. Criteria used in the system are useful for separating methods' situations.

Strongly Agree ☐	Agree ☐	Disagree ☐	Strongly Disagree ☐

10. Do you have any suggestion for adding new criteria to the system? Please explain with the reason.

11. This system provides knowledge transfer between project members.

Strongly Agree ☐	Agree ☐	Disagree ☐	Strongly Disagree ☐

12. Other comments or issues

Figure 3.63 SRMS's Questionnaire

Table 4. Summary of Responses to Questionnaire(in %)

Question Numbers	Options						
1	Yes **16.7**			No **83.3**			
2	Yes **75**			No **25**			
3	Strongly Agree **12.5**	Agree **62.5**		Disagree **25**	Strongly Disagree **0**		
3	Comment Available **25**			Comment Not Available **75**			
4	Strongly Agree **27.275**	Agree **45.45**		Disagree **27.275**	Strongly Disagree **0**		
5	Strongly Agree **58.4**	Agree **41.6**		Disagree **0**	Strongly Disagree **0**		
6	Strongly Agree **20**	Agree **80**		Disagree **0**	Strongly Disagree **0**		
7	Add New Method **17.64**	Add New Method By Extending **8.82**	Add New Method By Combining **20.59**	View Methods **5.89**	Update Method **5.89**	Take Report **14.70**	View Statistics **26.47**
8	Comment Available **50**			Comment Not Available **50**			
9	Strongly Agree **30**	Agree **70**		Disagree **0**	Strongly Disagree **0**		
10	Comment Available **25**			Comment Not Available **75**			
11	Strongly Agree **66.7**	Agree **33.3**		Disagree **0**	Strongly Disagree **0**		
12	Comment Available **16.7**			Comment Not Available **83.3**			

12 of the questionnaires were returned so the response rate of the questionnaire is approximately 86%. As shown in Table 4, a few questions are omitted by respondents. Moreover, questionnaire was applied to respondents who have at least few ideas about these types of systems and used lessons learned databases before. In addition to this, they have standards and have a quality group for process improvement. They are using these types of data for improvement of process used in the company.

Question 1 was asked to learn experience of the respondents about this type of method storage system. Only 16.7 % of the respondents stated that they have used this type of a system before so they are familiar with the concept very much. On the other hand, 83.3 % of the respondents stated that they have not used this type of system before. So it can be said that, minority of the respondents used this type of a system before because there are very few tools implementing situational method engineering approach and these are especially research tools. So SRMS is a new tool that implemented this approach.

As mentioned before, lessons learned databases keeps a knowledge base for future projects. Question 2 was asked to learn lessons learned database usage experience of the respondents. Since SRMS is logically similar to lessons learned, it is important to know this rate. 75% of the respondents stated that they have used lessons learned database before, whereas 25% of the respondents stated that they have not used lessons learned database before. As the rate shows, it is obvious that the respondents could easily compare SRMS with lessons learned database.

To learn which system is better, Question 3 was asked to respondents. This question is linked with Question 2. If Question 2's answer was yes, respondents were able to answer this question so 75% of the respondents have easily answered Question 3. 12.5% of the respondents strongly agreed that SRMS is better than lessons learned database. 62.5% of them agreed that SRMS is better than lessons learned database whereas 25% of the respondents disagrees that SRMS is better than lessons learned database. So majority of the respondents found that SRMS is a better system than lessons learned database. In general, their experience was good with SRMS. Moreover, if there was disagree or strongly disagree answer given to Question 3 to learn the reason of that, a space for stating the reason was provided to the respondents. 25% of the respondents stated their reasons in this area, whereas 75% of them did not. They thought that they already have a more customized tool than SRMS. Respondents did not give much detail about their tool actually. There can be much detailed tools than SRMS but SRMS's motivation is ease and effectiveness of usage. Respondent thought that in lessons learned databases, topics are organized in a tree view structure that means the inheritance between the subjects can easily observed but in SRMS the methods are organized in a flat manner. Another suggestion was that it would be nice to see which methods are extended (that means not created from scratch) and the source method used in extension. It is possible to incorporate this suggestion. At the background of the SRMS, it is stored that the method is new, extended or combined type. For methods that are extended from some existing method, an additional field storing the source method name can be added to incorporate this suggestion fully. Moreover, respondent thought that lessons learned databases and SRMS are incompatible but they have similarities coming from the origin. They both provide knowledge transfer between the team members. On the other hand, their storage approaches are different from each other.

Question 4 was asked to learn the tendency of the respondents about using SRMS in their intranet system. 27.275% of the respondents stated that they strongly agree that they prefer to use SRMS in their intranet systems. Moreover, 45.45% of the respondents agreed that they prefer SRMS. On the other hand, 27.275% of the respondents stated that they do not prefer to use SRMS in their intranet system. So as the rates are showing, SRMS is preferred by majority of the respondents according to their experience with SRMS.

One of the most important features of any tool is the ease of use so Question 5 is asked to find the view of the respondents about the ease of the use of SRMS. 58.4% of the respondents strongly agreed that SRMS is easy to use. Moreover, 41.6% of the respondents agreed that SRMS is easy to use. On the other hand, none of the respondents thought that SRMS is hard to use. So, SRMS usage is very easy. Every engineer involving the development can easily store their methods in SRMS and search for new methods in the system that is used before.

In SRMS, methods are stored in two parts. First part is characteristics of that method and then the steps of that method. Question 6 was asked to respondents to learn their view about the SRMS's method storage principle's usefulness. 20% of the respondents strongly agreed that SRMS's method storage principle is useful. Moreover, 80% of the respondents are also agreed that SRMS's method storage principle is useful whereas, none of the respondents disagreed that SRMS's method storage principle is useful. It is understandable that SRMS's one of promises is fulfilled.

There are seven main modules in SRMS. Their names are: Add New Method, Add New Method By Extending One, Add New Method By Combining Two, View Methods, Update Method, Take Report and View Statistics. To learn which modules are more useful, Question 7 was asked. Respondents can choose more than one option. 17.64% of the respondents stated that Add New Method module is useful. 8.82% of the respondents stated that Add New Method By Extending One module is useful. 20.59% of the respondents stated that Add New Method By Combining Two is useful. Moreover, 5.89% of the respondents stated that View Methods module is useful and also Update Method module stated useful by 5.89% of the respondents. Take Report module stated useful by 14.70% of the respondents. Furthermore, 26.47% of the respondents stated that View Statistics module is useful. It can be easily analyzed from this result that, the majority of the respondents found that View Statistics module is the first number module in terms of usefulness. Second one is Add New Method By Combining Two and the third one is Add New Method. So it is obvious from this result that viewing statistics about the methods stored in the system is important for the users. It can easily affect the usage of a method and creation of a method.

Although there are mainly seven modules in the system, one or more modules can be added to the system according to needs. To take the suggestion of the respondents on this issue, Question 8 was asked and a blank area was provided to user for collecting their comments on this subject. 50% of the respondents stated their comments about adding a new module to the system. On the other hand, 50% of the respondents did not state any comment about adding a new module to the system. Respondent stated that tree viewing functionality that shows the relationships can be added to the system. As mentioned in Question 3, it can be added to the system. Respondents stated that while adding a new method to the system, how to use and purpose of the system can added and there can be more detailed questions.

Explanations of modules are already defined in the help part of SRMS. Respondent stated that a feedback system from user is beneficial, that is, a rating from each user who are using the method. It is very beneficial to add this feature to the system so users can easily view the ratings of the methods. There is a comment is system, it can be attached to this module. Respondent stated that suggesting methods with respect to process can be good for the system. Methods that are stored in the system, stored according to the characteristics of the project. They can be applied to any process.

As mentioned in Section 3, there are eight criteria explained. To learn effectiveness of the criteria used in the system Question 9 was asked to the respondents. 30% of the respondents strongly agree that the criteria are good at separating methods' situations. Moreover, 70% of the respondents agree that the criteria used in the system are good at separating methods. On the other hand, none of the respondents disagree or strongly disagree that criteria used in the system is good at separating the methods. Criteria used in the system are defined to separate the method's situations so majority of the respondents thought that they are good at separation. It can be said that the aim of the criteria is accomplished.

Although there are eight criteria in the system, Question 10 is asked to respondents about any suggestion for adding new criteria to the system. A blank area was provided to respondents for collecting their ideas. 25% of the respondents stated their ideas about this issue. On the other hand, 75% of the respondents did not state any comment about this issue. Respondent stated that "Select acquaintance level of the customer to the project domain" criteria can be added to the system because even if the customer is the owner of the project, he/she may not have enough know-how in the application domain. This is generally the case in the outsourced projects. This suggestion is good but this criterion is already included in SRMS as "Select experience level of your customer in software environment". On the other hand, respondent stated that comments field may be added to be more explanatory. Commenting on methods is already provided to SRMS's users. They can add their comments on methods. For example: if one step in the method is more important than the others, it can be added as a comment from View Method's View/Add link.

Knowledge transfer is the most valuable activity for the organizations nowadays. SRMS's aim is providing knowledge transfer between team members. To learn effectiveness of SRMS in knowledge transfer between team members is, Question 11 was asked to respondents. 66.7% of the respondents strongly agree that SRMS provides knowledge transfer between team members. 33.3% of the respondents agree that SRMS provides knowledge transfer between team members. On the other hand, none of the respondents disagree or strongly disagree that SRMS provides knowledge transfer between team members. It is obvious that, majority of the respondents thought that SRMS provides knowledge transfer between team members so SRMS can be valuable if it can be used effectively by team members.

Most of the opinions about SRMS were obtained from eleven questions but there is a need to provide a blank part for respondents for collecting any other opinions about SRMS. 16.7% of the respondents stated their thoughts about SRMS. On the other hand, 83.3% of the respondents did not state any other comment about SRMS. Respondent stated that association between processes and SRMS methods should be allowed. This issue is already explained in Question 8's related paragraph. Respondent also stated that keeping track of in which projects, which methods were utilized, what was the result, conditions and people involved can be good. Characteristics of the project is important but our main motivation is storing methods not project management related data. Other comment is stated that, it would help to specify ranges for budget level, interaction level and complexity level. It was searched from the literature but these ranges are different in some situations so it was preferred to define as in SRMS.

CHAPTER 4

4. CONCLUSION

This thesis proposes a new situational method engineering approach for the situation that is related with project characteristics. Although there are many method engineering approaches described in Section 2.3 existing, none of them can be used for all situations. Combination of methods or more than one method can be used in one project to cover the situation at hand. Since knowledge management has an important role in current projects in the world, new approach was created. This approach accepts method as a hole rather than other approaches. New approach focuses on requirement engineering phase. Since, requirements engineering is the first phase in the software development, it is very critical. This phase should be conducted very well because the problems that are created in this phase will cause the project failure at the end. Although there are many of the methods in requirement engineering phase, they have advantages and disadvantages. These methods were examined in Section 2.1.3. Moreover, the advantages and disadvantages were also exposed in Section 3.2. It was realized that there was a need to categorize the methods according to situations at hand so criteria were established according to these studies. The criteria were explained in Section 3.3 in detail. After establishing the criteria, proposed approach was demonstrated in a web-based tool. This tool is called Situational Requirement Method System, SRMS in short. Its purpose is providing a new, easy tool for situational method engineering.

SRMS tool architecture and usage was described in Section 3.4. Strengths of the tool can be easily understandable from this section. There are many tools that are compared with SRMS in Section 3.5. Tools that described in this section are carrying common seven features [Odell, 95], [Harmsen, 94] of CAME tools at least partially. SRMS is also carrying these features. SRMS is one of the easy tools for method storage and management. One of the interesting thing is in this comparison was, there was only one commercial tool in these tools. This also shows that there is not much qualified tool for this purpose. SRMS brings a new perspective to this type of tools. As mentioned before, it uses assembly-based and extension-based situational method engineering approaches. SRMS prevents knowledge loss that is very important in software industry. Enables sharing of knowledge and holds them alive. With using SRMS, methods that are suited to a specific project can be easily extracted from the system. When a method is entered to the system, its format is controlled by SRMS. Moreover, it provides more user contribution.

As mentioned in Section 3.6, SRMS related questionnaires were distributed to the users who are working in large software companies and have a good working knowledge about this type of systems. 86% of the questionnaire was responded. Compared to lessons learned databases majority of them found SRMS is better. Moreover, this can be represented in percentages. 75% of the respondents have used lessons learned databases before. Further more, 12.5% of respondents strongly agreed and 62.5% of respondents agreed that SRMS is better than storing lessons learned databases.

As results showing, SRMS and its approach brings new perspective to situational method engineering. Some suggestions are gathered from the questionnaire from the respondents of this questionnaire. As mentioned in Section 3.6, some of the suggestions may be added to SRMS. It is possible to incorporate the suggestion that which methods are extended (that means not created from scratch) and the source method used in extension. At the background of the SRMS, it is stored that the method is new, extended or combined type. For methods that are extended from some existing method, an additional field storing the source method name can be added to incorporate this suggestion fully. In addition to this, tree viewing functionality that shows the relationships may be added to the system. Since SRMS focuses on requirements engineering phase, with little modifications, it can be also customized and used in different phases of the software development.

When collecting questionnaires, some of the respondents asked for permission to try SRMS in some of their projects so SRMS will be used in real projects. According to their feedbacks, SRMS can be modified to match with the needs of real life projects. After that, SRMS tool can be distributed as a free situational method engineering tool.

Accepted Paper:

Seçil Aydın & Deepti Mishra (2009) “A Tool to Enhance Cooperation and Knowledge Transfer among Software Developers” in the 6th International Conference on Cooperative Design, Visualization and Engineering (CDVE) will be held at Luxemburg on 20th – 23rd September 2009 and will be published in Lecture Notes in Computer Science-Springer.

REFERENCES

[Leffingwell&Widrig, 2000] Leffingwell D., Widrig D., Managing Software Requirements A Unified Approach, Addison Wesley, 2000.

[Arni-Bloch, 99] Arni-Bloch, N., Towards a CAME Tools for Situational Method Engineering, University of Geneva, 1999.

[Ralyte&Deneckere&Rolland, 03] Ralyte J., Deneckere R., Rolland C., Towards a Generic Model for Situational Method Engineering, Springer-Verlag Berlin Heidelberg, 2003.

[IEEE, 93] IEEE Standards Collection: Software Engineering, IEEE Standard 610.12-1990, IEEE, 1993.

[Boehm, 81] Boehm B.W., Software Engineering Economics, Prentice-Hall, 1981.

[Pressman, 05] Pressman, R.S., Software Engineering A Practitioner's Approach, McGraw-Hill, 2005.

[Bauer, 68] Bauer, F., Software Engineering: A Report on a conference sponsored by the NATO Science Committee, 1968.

[Berry, 92] Berry, D.M., Academic Legitimacy of the Software Engineering Discipline, Software Engineering Institute, 1992.

[Sommerville, 95] Sommerville I., Software Engineering, Addison Wesley, 1995.

[Sanz, N/A] Sanz, L.F., Software Quality: Concepts and Evidences, Universidad Europe de Madrid.

[Paulk&Curtis&Chrissis&Weber, 93] Paulk M.C., Curtis B., Chrissis M.B., Weber C.H., Capability Maturity Model for Software, Version 1.1, Software Engineering Institute, 1993.

[Oy, 86] Oy I., Specifying Systems and Applications With SA/SD/RT Method, 1986.

[Scacchi, 01] Scacchi W., Process Models in Software Engineering, Encyclopedia of Software Engineering, December 2001.

[Ravindran, 07] Ravindran, C. ,Rapid Application Development Model (RAD), EzineArticles.com, 2007 January 9.

[Scnach, 04] Scnach. S.R., Object-Oriented and Classical Software Engineering, McGraw-Hill, 2004.

[Boehm, 88] Boehm B.W. A Spiral Model of Software Development and Enhancement, COMPUTER, 1988.

[Hughes&Cottrell, 06] Hughes B., Cottrell M., Software Project Management, McGraw-Hill,2006.

[Pillai,96] Pillai K., The Fountaion Model and Its Impact On Project Schedule, ACM SIGSOFT, Software Engineering Notes, 1996.

[IEEE, 90] IEEE Standard Glossary of Software Engineering Terminology, IEEE, 1990.

[Ghose, 99] Ghose, A.K., Managing Requirements evolution: Formal support for functional and non-functional requirements, 1999.

[Mylopoulos&Chung&Nixon, 92] Mylopoulos J., Chung L. and Nixon B. Representing and Using Non-Functional Requirements: A Process-Oriented Approach, IEEE Transactions on Software Engineering, 1992.

[Thayer&Dorfman, 97] Thayer R.H., Dorfman M., Software Requirements Engineering, IEEE Computer Society, 1997.

[SEI, 91] Software Engineering Institute Requirements Engineering Project, Requirements Engineering and Analysis Workshop Proceedings, Technical Report, Carnegie Mellon University, 1991.

[Christel&Kang, 92] Christel M.G., KANG K.C., Issues in Requirements Elicitation, Software Engineering Institute, 1992.

[Goguen&Linde, 93] Goguen J.A, Linde C., Techniques for Requirements Elicitation, IEEE Computer Society, 1993.

[Wang&Lai, 01] Wang Q., Lai X., Requirements Management for the Incremental Development Model, IEEE, 2001.

[Dominguez&Zapata, 05] Dominguez E., Zapata M.A., Noesis: Towards a situational method engineering technique, Science Direct, 2005.

[Weer&Souer&Versendaal&Brinkkemper, 05] Weerd I.V., Souer J., Versendaal J., Brinkkemper S., Situational Requirements Engineering of Web Content Management Implementations, Proceedings of SREP, Paris, 2005.

[Bruegge&Herrmann&Rauschmayer&Renner, N/A] Bruegge B., Herrmann K., Rauschmayer A., Renner P., Situational Requirements Engineering gets distributed!, Technical University of Munich, Germany, N/A.

[Hickey&Davis&Kaiser, 03] Hickey A.M., Davis A.M., Kaiser D., Requirement Elicitation Techniques Analyzing the Gap Between Technology Availability and Technology Use, Comparative Technology Transfer and Society, December 2003.

[Maiden&Rugg, 96] Maiden R., Rugg G., ACRE: Selecting methods for requirements acquisition. Software Engineering Journal 11(3): 183-192, 1996

[JpGraph, 09] JpGraph Version 1.27, http://www.aditus.nu/jpgraph/index.php, Aditus Consulting, 2009.

[Brinkkemper, 96] Brinkkemper, S., Method Engineering: Engineering of Information Systems Development Methods and Tools. Information and Software Technology 38, 1996.

[Serour&Sellers, 04] Serour M.K., Sellers H.B., Introducing Agility: A Case Study of Situational Method Engineering Using the OPEN Process Framework, Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC'04), 2004.

[Mirbel&Ralyte, 05] Mirbel I., Ralyte J., Situational method engineering: combining assembly-based and roadmap-driven approaches, Requirements Eng (2006) 11: 58–78 DOI 10.1007/s00766-005-0019-0, Springer-Verlag London Limited, 2005.

[Coulin&Zowghi&Sahraoui, 06] Coulin C., Zowghi D., Sahraoui A., A Situational Method Engineering Approach to Requirements Elicitation Workshops in the Software Development Process, *Softw. Process Improve. Pract.* 2006; 11: 451–464, Wiley InterScience, 2006.

[Niknafs&Ramsin, 08] Niknafs A., Ramsin R., Computer-Aided Method Engineering: An Analysis of Existing Environments, Advaced Information Systems Engineering, Volume 5074, Springer Berlin/Heidelberg, 2008.

[Heym&Osterle, 92] Heym M., Osterle H., A semantic data model for methodology engineering, Computer-Aided Software Engineering, 1992. Proceedings., Fifth International Workshop, IEEE, 1992.

[Plihon, 96] Plihon, V. 1996. MENTOR: An Environment Supporting the Construction of Methods. In *Proceedings of the Third Asia-Pacific Software Engineering Conference* (December 04 - 07, 1996). APSEC. IEEE Computer Society, Washington, DC, 384.

[Harmsen, 94] Harmsen, A.F., Brinkkemper, S., Oei, H., Situational Method Engineering for Information Systems Projects In Methods and Associated Tools for the Information Systems Life Cycle. Proceedings of the IFIP WG8.1 Working Conference Cris/94, T.W. Olle, A.A. Verrijn-Stuart, Eds. North Holland, Amsterdam, 1994, 169-194

[Odell, 95] Odell, J.J., Introduction to Method Engineering. Object Magazine, 1995

[Hickey&Davis, 03] Hickey, A. M. and Davis, A. M. 2003. Requirements Elicitation and Elicitation Technique Selection: A Model for Two Knowledge-Intensive Software Development Processes. In *Proceedings of the 36th Annual Hawaii international Conference on System Sciences (Hicss'03) - Track 3 - Volume 3* (January 06 - 09, 2003). HICSS. IEEE Computer Society, Washington, DC, 96.1.

[Kumar&Welke, 92] Kumar, K. and Welke, R. J. 1992. Methodology EngineeringR: a proposal for situation-specific methodology construction. In *Challenges and Strategies For Research in Systems Development*, W. W. Cotterman and J. A. Senn, Eds. John Wiley Information Systems. John Wiley & Sons, New York, NY, 257-269.

[Henderson, 03] Henderson-Sellers, B. 2003. Method engineering for OO systems development. *Commun. ACM* 46, 10 (Oct. 2003), 73-78. DOI= http://doi.acm.org/10.1145/944217.944242

[Agerfalk&Ralyte, 06] Agerfalk P.J., Ralyte J., Guest Editorial Situational Requirements Engineering Processes: Reflecting on Method Engineering and Requirements Practice, Software Process Improvement and Practice.2006, 11:447-450, Wiley InterScience, 2006.

[Rolland, 07] Rolland C., Method Engineering: Achievements, Trends & Challenges, Keynotes, IFIP Conference Geneva, 2007.

[Ralyte&Rolland&Ayed, 05] Ralyte J., Rolland C., Ayed M.B.: An Approach for Evolution-Driven Method Engineering. Information Modeling Methods and Methodologies 2005: 80-101.

[Mirbel&Ralyte, 05] Mirbel, I. and Ralyté, J. 2005. Situational method engineering: combining assembly-based and roadmap-driven approaches. *Requir. Eng.* 11, 1 (Dec. 2005), 58-78. DOI= http://dx.doi.org/10.1007/s00766-005-0019-0

[Chandani& Neeraja& Sreedevi, 07] Chandani A., Neeraja B., Sreedevi. Knowledge Management: An overview & its ımpact on Software Industry, International Conference on Information and Communication Technology in Electrical Sciences (ICTES 2007), pp. 1063-1068.

[Saeki, 03] Saeki M., Toward Automated Method Engineering: Supporting Method Assembly in CAME, Invited talk in EMSISE03 Workshop, Geneva, Switzerland, 2003.

[Ucertify, 08] Ucertify.com, What are the layers of software engineering?, http://www.ucertify.com/article/what-are-the-layers-of-software-engineering.html, 2008.

[Tolvanen, 98] Tolvanen J.P., Incremental Method Engineering with Modeling Tools: Theoretical Principles and Empirical Evidence, University of Jyvaskyla, 1998.