International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 5 Issue: 5 1173 – 1188

Automation of Requirement Analysis in Software Engineering

[1]Amit Mishra, [2]A. Awal, [3]Joseph Elijah, [4]A. AbdulG, [5]U. M, Gana, [6]I. Rabiu,
[1,5]Deptt. of Mathematics/Computer Science Ibrahim Badamasi Babangida, University, Nigeria
[1]i.amitmishra@gmail.com,[2]meeayeesha@yahoo.com[3]elijoe3yk@gmail.com,
[4]abdulg2009@yahoo.com[5]umgana2003@yahoo.com[6]idrisrabiu76@yahoo.com

Abstract: Software engineering practices are the most important practices for the success of software. Requirement engineering is the first and crucial phase in the development of software. Success or failure of any software is truly dependent upon the requirement analysis. It involves set of activities like system feasibility study, elicitation analysis, validation and management of the requirements. There are many methods already exist to perform the requirement gathering process and the system analyst apply them to gather the requirements but still they are facing many problems in gathering the requirements. These problems occur due to the communication gaps between customers, engineers and project managers, and requirements information loss might occur across different software development process. This affects the quality of the software and increase the production cost of software. Reviews of user requirement analysis technique from the literatures were studied by listing their advantages and limitation. Based on the limitations of the reviewed literatures an automated system has been developed for gathering user requirements which will go along in bridging the communication gaps between the users and the analyst.

*****

I. INTRODUCTION

System and software development projects have been plagued with problems since the 1960's. Requirement Engineering (RE) has become one of the central research topics in the field of software engineering. Although progress in requirement engineering has been painfully slow with software development projects (**Badariah solemon, et *al*. 2012**). Software engineering typically refers to a regimented and procedural methodology for developing software (**Kerry, E. and Delgado, S. 2009**) the requirement state is the major process in software development life cycle (SDLC). SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process (**tutorials point, SDLC**). Errors made in this process are extremely expensive to correct when they are detected during implementing or testing period. Affected software quality would give a significant impact on society and economy such as increasing costs and business interruption (**Fauziah B, et *al*. 2005**). SDLC include the following phases: planning phase, analysis phase, design phase, development phase, testing phase, implementation and maintenance phase.

Requirement analysis phase describes what the system stakeholders require or expect from the system. It encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product (**Wikipedia, 2014**). Requirement analysis is critical to the success of systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detailed sufficient for system design (**Wikipedia, 2014**). Analysis phase is divided in problem analysis and system analysis: problem analysis is concern withthe process of understanding real-world problems and user needs and proposing solutions to meet those needs.The goal of problem analysis is to gain a better understanding, before development begins, of the problem being solved. While system analysis is concern withprocess by which users and analysts help each other reach an understanding of the system requirements that is sufficient for their accurate specification. System analysis defines the problems to be solved and provides the architecture of the proposed system. As information systems became more complex, system analysts sought advance tools to assist them in the analysis processwhich include requirement elicitation (**Hui zhou 2004**).Requirement elicitation encompasses all activities involved in discovering the requirements of a system. Requirements elicitation techniques are the means by which systems analysts determine the problems, opportunities, and needs of the customers, so that systems developer can construct systems that actually resolve those problems, leverage those opportunities, and/or address customers‟ needs". This process is useful in finding the problems that has to be solved. Requirement elicitation consists of various techniques which include interviewing, questionnaire, document sampling, observation, prototyping and brainstorming. Interviewing was used as the elicitation technique on this research. Interviewing consists of asking the domain expert questions about the domain of interest and how they perform their tasks. The success of an

interview session is dependent on the questions asked and the ability of the expert to articulate their knowledge during requirement gathering. It is widely acknowledged that requirement gathering is done mostly using natural language and that there is a gap between stakeholders and analyst understanding during the requirement gathering process because analyst tend to avoid interaction and direct involvement with users (**Jabar M.A, et *al.* 2012**).This problem will cause many effects on the quality of software for the system development. To overcome this problem, an empirical model for an automated software requirement analysis was proposed and developed. The main function is to close the understanding gap between stakeholder (user) and system analyst in order to construct a better architecture to achieve the usability of software. Usability is the capability of the software to be understood, learned, used and attractive to the user when used under specified conditions (**Kayed A.H, et *al.* 2009**). Based on Cusumano survey, they suggested that early planning and customer specifications are crucial to productivity, whereas "doing it right the first time" is essential for reducing development time. Therefore, it is important to make sure that the requirement process is properly managed (**Cusumano M., et *al.* 2003**).

II. LITERATURE REVIEW

Research focusing on requirement analysis in general has been an increasingly trending topic in software engineering since 1960 (**Badariah solemon et *al.* 2012**). In this project, four papers have been studied and each of it proposed a model for software requirement process. Their advantages and limitation are listed. A tabulated summary based on the discussion are produced to form a picture of user requirement technique.

Based on Zhaoyin study (**Zhang Z., et *al.* 2009**) proposed a kind of software requirement analysis technology method based on event-driven. Even a possible action the user can perform with a system. That is what the user wants to do such as send order or make booking. From the event, it will get the detail on what will trigger the event, the source and action taken from the event table. Event table is the process in event-driven technique to get the requirement. It can be use in structured analysis method and object-oriented method. Structured analysis is a software engineering technique to construct and refine a system specification from requirements, which has been used for a long time. Object-oriented is a programming paradigm that uses "object" consisting of data fields and method together with their interaction. For structured analysis method, DFD fragments and 0-level figure are developed based on what the event table gets. The DFD diagram then will provide the analytical model on the system. Furthermore, the subsystem can be divided through events and external entities interaction, data storage interaction and common point analysis in processing requirement. Regarding the object-oriented method, the event table will develop the Use Case Diagram and Sequence Diagram. Use Case Diagram and Sequence Diagram is the Behavior Diagram in Unified Modeling Language (UML). Behavior diagrams emphasize what must happen in the system being modeled. Since behavior diagrams illustrate the behavior of a system, they are used extensively to describe the functionality of software systems. This paper uses the case analysis to show how to use the method to carry on structured analysis and complete software requirement modeling. The result shows that the method is easy to use.

Chih-Wei Lu (**Chih-Wei Lu, et *al.* 2008**) present a requirement editor, called MOR editor that supports the objectization and modeling of requirement engineering. MOR editor is a tool designed to assist the processes and of requirement documents, which can benefit requirement engineer to objectize requirement artifacts, link-related requirement artifacts, and construct a consistent and traceable formal model for RE. It describes that most requirement documents are written in natural language and represented in less structured and imprecise formats. A case study was done for this model to shows its effectiveness. The model is still in prototype version and it is still in developing process. Part of the objectives of MOR Editor is to enhance the traceability, consistency, completeness, and maintainability of requirement documents.

Researcher showed that (**Chih-Wei Lu, et *al.* 2008**) an integrated framework for semantic requirement engineering. Semantic is the study of meaning. It typically focuses on the relation between signifiers, such as words, phrases, signs and symbols, and what they stand for. It combines domain ontology, enterprise ontology and user ontology to enable semantic representation and reasoning of software requirement. Ontology is a formal representation of the knowledge by a set of concepts within a domain and the relationships between those concepts. It is used to reason about the properties of that domain, and may be used to describe the domain. This is one way to closing the understanding gap between stakeholder and system developer. But the framework never produces any tools yet in order to gather the user requirement.

Another study presents a new approach for linking Requirement Engineering activities into a Process Framework that can be used as a reference for driving concrete Requirement Engineering processes (**Garcia Alcazar, E. and Monzon, A. 2000**). In order to build a common understanding of the problem context, special emphasis is placed on construction Problem Domain models. It is use to situate user requirement with reference to it, and as a technique for user requirements refinement. Process Framework also highlights some aspect such as

separation of the elicitation, analysis and refinement of user requirement from the construction of a system or software specification.There has been a tool that uses the Process Framework and it has been tested in several projects from different organization. The tool seems working but still can be enhance it. There are several repeated problems occur based on the paper. The effort required to build the Problem Domain models and to link the user requirement to them is the problem. Problem Domain is the area of expertise or application that needs to be examined to solve a problem. The process framework is supported by a tool and has been tested in several projects from different organizations.

Model/framework	**Benefits**	**Limitation**
Software requirement analysis technology method based on event-driven (**Zhang Z., et *al*. 2009**)	1. Suitable for structured analysis method and object oriented method. 2. Easy and feasible method.	1. Suitable for functional requirement only. 2. Users need to have basic skill in software engineering to use this model.
Model-driven Object-oriented Requirement Editor (MOR Editor) (**Chih-Wei Lu, et *al*. 2008**)	1. MOR Editor supports the objectization and modeling of requirement engineering.	1. Extended prototype is required to support software development process. 2. No elicitation on the requirement part.
Integrated framework for semantic requirement engineering (**Chih-Wei Lu, et *al*. 2008**)	1. Domain ontology, enterprise ontology and user ontology to enable semantic representation and reasoning of software requirement.	1. No requirement elicitation process involve in the framework. 2. It focuses on the integration part between ontology and database.

 ISSN: 2321-8169
1173 – 1188

Process framework for requirement analysis and specification (**Garcia Alcazar, E. and Monzon, A. 2000**)	1. Can be used as a reference for driving concrete Requirement Engineering processes. 2. Separation of the elicitation, analysis and refinement of user requirement from the construction of a system or software specification. 3. Supported by a tool and has been tested in several projects from different organization.	1. It has a repeated problem that is the effort required to build the Problem Domain models and to link the user requirements to them. 2. Automation system still in research stage. 3. Not explicitly differentiate between functional and non-functional requirements.

Table 2.0 is the summary of the literature review, it is tabulated into model and for each model the benefits and limitation are highlight.

III. RESEARCH METHODOLOGY

This section of the research work describes the system methodology, analysis and design. The Application was developed following the Software Development Life Cycle. SDLC, Software Development Life Cycle is a process used by software industry to design, develop and test high quality software (**Tutorials point, SDLC**).

 ISSN: 2321-8169
1173 – 1188

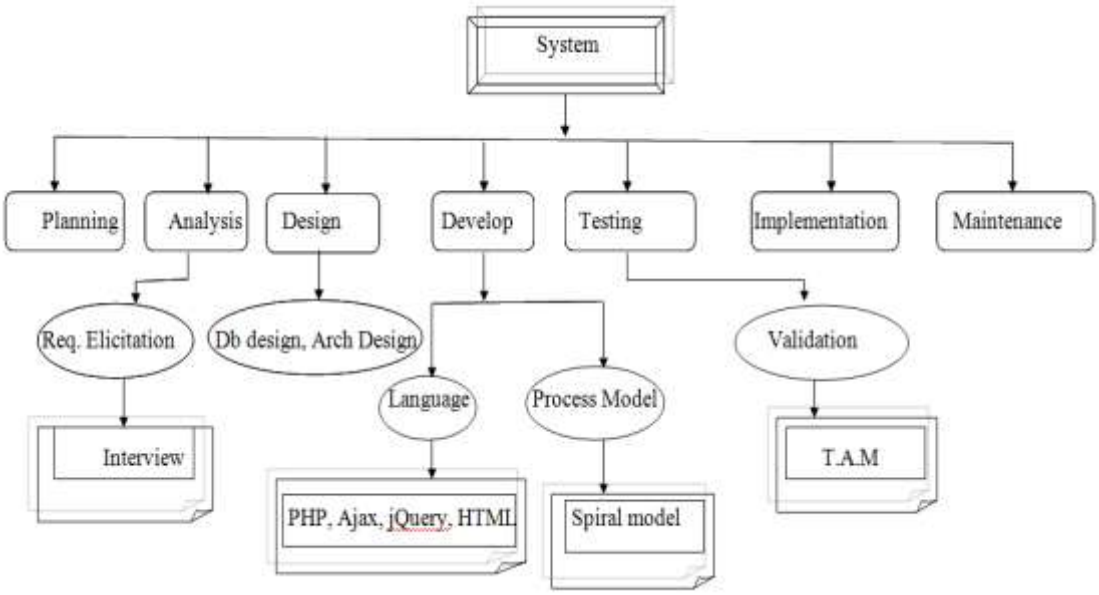

Figure 1: Flow Representation of phases carried out during development of the tool.

IV. SYSTEM ANALYSIS AND DESIGN

There are various SDLC models defined and designed which are followed during software development process. Each process model follows a Series of steps unique to its type, in order to ensure success in process of software development. Following are the most important and popular SDLC models followed in the industry: waterfall model, iterative model, spiral model, v-model and big-bang model. For this research development, the spiral process model was followed.

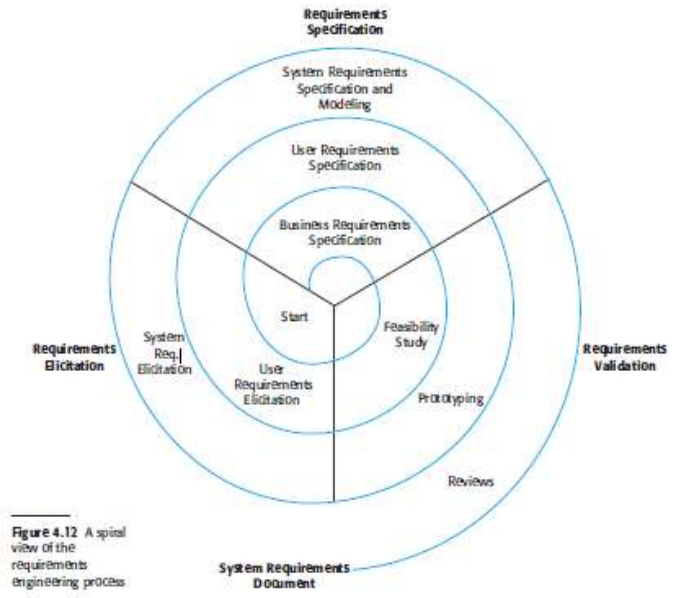

Figure 2: Diagrammatic Representation of a Spiral Model (**Sommerville, 2011**)

Spiral Model is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model. It provides the potential for rapid development of incremental versions of the software.

Function Hierarchy Diagram

The function hierarchy diagram shows the system functions that will be constructed and the implementation process of data diagram.

 ISSN: 2321-8169
1173 – 1188

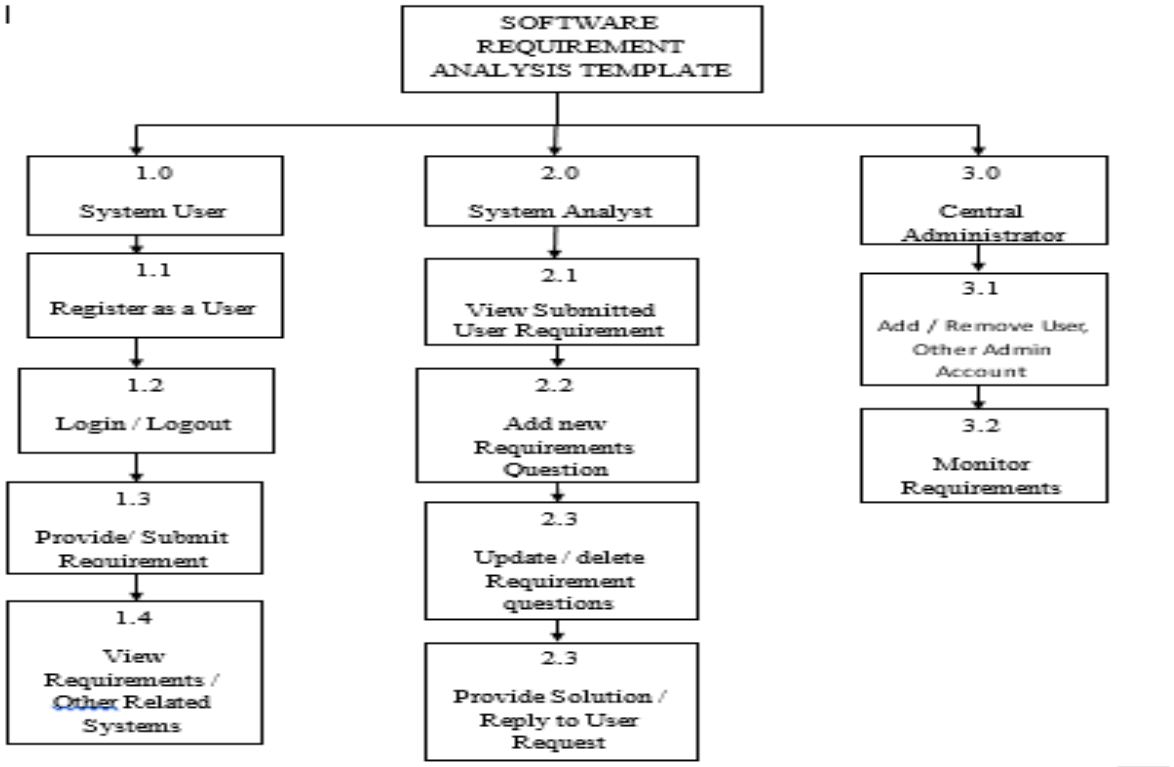

Figure 3: Function hierarchy diagram

Use Case Diagram: A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

Use Case Overview

Figure 4: Use Case Diagrams (High Level)

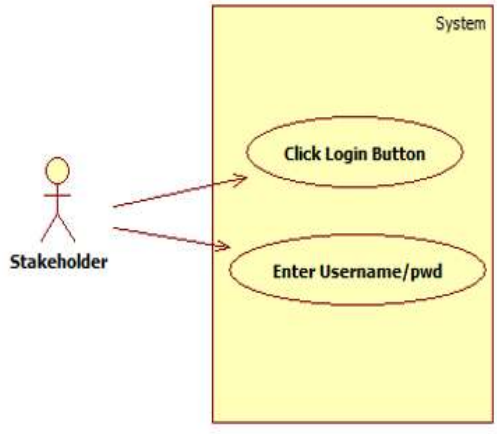

Figure 5: User Login

 ISSN: 2321-8169
1173 – 1188

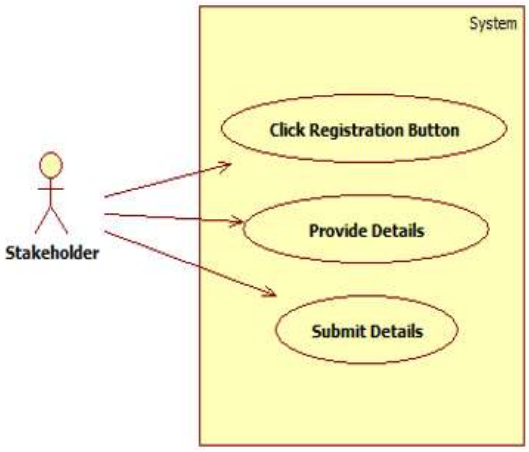

Figure 6: User Registration

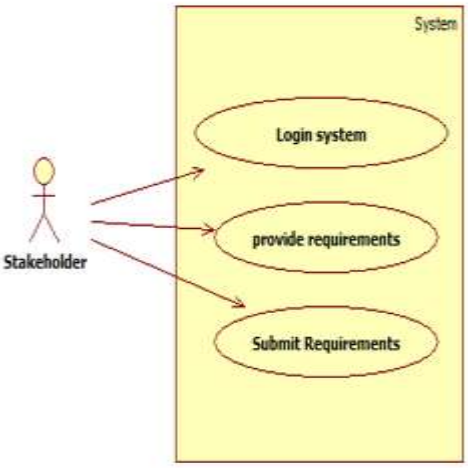

Figure7: User Provide Requirement

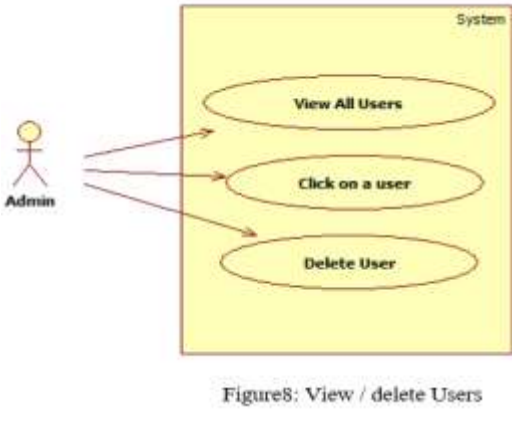

Figure8: View / delete Users

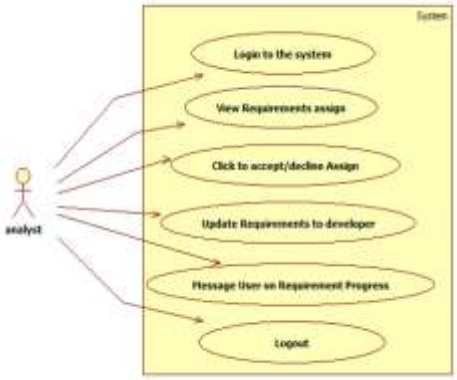

Figure 9: Analyst Manages User Requirement.

DESIGN PHASE

Snapshots of the prototype design:

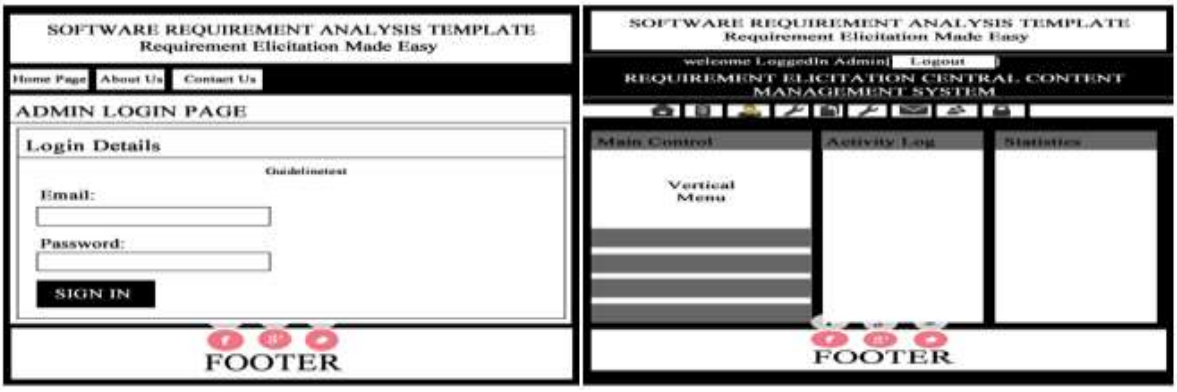

Figure 10: Prototype design for user Homepage and Login Page

Figure 12Prototype design for user Registration Page and Homepage for Logged in Users.

 ISSN: 2321-8169
1173 – 1188

Figure 11: prototype Design for User Registration Page and Homepage for Logged Users

Database Design

The database of choice for this project is MySQL due to its relatively light weight, simplicity and seamless interaction with PHP. The databases take care of all data being store on the site, inputting or retrieving the needed information on request of the server side script. The database was designed based on the repository developed by Musa Agaie.

Database Architecture with views

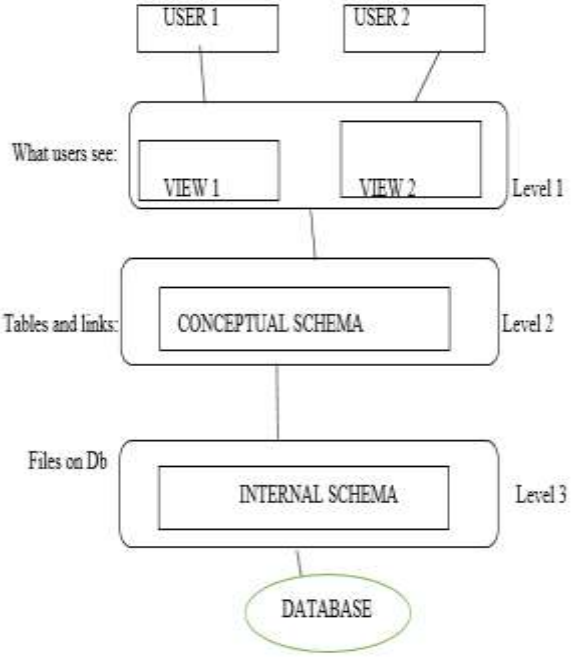

Figure 12: Database Architectural View: Each Level is independent of the level below (**Manos Papagelis, 2014**).

a) Data Independency

Level 1 Level 2: (Logical Independence)

This Independency is the ability to change the logical schema without changing the external schema or application programs

i) Central administration can add new fields, new tables without changing view

ii) Central administrator can change structure of tables without changing view

Level 2 Level 3: (Physical Independence)

The ability to change the physical schema without changing the logical schema

i) Storage space can change

ii) Data Type of some data can change for reasons of optimization

b) Data Dictionary

A data dictionary is the database designer's most important form of documentation it performs the following functions.

- Describes database constraints such as the use of NOT NULL values.
- Documents the internal structure of each table, including all fields and their data types with comments, all indexes, and all views.

Database structures of this Research are shown below:

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	admin_id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Primary
2	username	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
3	password	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
4	firstName	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
5	lastName	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
6	email	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
7	gender	varchar(10)	latin1_swedish_ci		No	None		Change Drop Primary
8	admin_type	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
9	date_reg	date			No	None		Change Drop Primary

Figure 13: Admin Table structure

 ISSN: 2321-8169
1173 – 1188

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	activity_id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Primary
2	activity_name	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
3	activity_details	varchar(200)	latin1_swedish_ci		No	None		Change Drop Primary
4	activity_time	datetime			No	None		Change Drop Primary

Figure 14: Activity Monitoring Table

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	user_id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Primary
2	email	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
3	password	varchar(15)	latin1_swedish_ci		No	None		Change Drop Primary
4	full_name	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
5	phone	int(11)			No	None		Change Drop Primary
6	company	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
7	level_of_experience	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
8	market_business	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
9	comment	text	latin1_swedish_ci		No	None		Change Drop Primary
10	date_join	datetime			No	None		Change Drop Primary

Figure 15: Users Table

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	q_id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Primary
2	category	varchar(25)	latin1_swedish_ci		No	None		Change Drop Primary
3	added_by	int(11)			No	None		Change Drop Primary
4	add_date	datetime			No	None		Change Drop Primary

Figure 16: Class Requirement Question

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	srq_id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Primary
2	q_id	int(11)			No	None		Change Drop Primary
3	srq_details	varchar(50)	latin1_swedish_ci		No	None		Change Drop Primary
4	added_by	int(11)			No	None		Change Drop Primary
5	date_added	date			No	None		Change Drop Primary

Figure 17: Sub Class Requirement Question

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	chat_id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Primary
2	sender	int(11)			No	None		Change Drop Primary
3	receiver	int(11)			No	None		Change Drop Primary
4	message	text	latin1_swedish_ci		No	None		Change Drop Primary
5	date_sent	datetime			No	None		Change Drop Primary
6	status	tinyint(1)			No	None		Change Drop Primary

Figure 18: Message Chat Table

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	t_id	int(10)		UNSIGNED	No	None	AUTO_INCREMENT	Change Drop Primary
2	admin_id	int(11)			No	None		Change Drop Primary
3	uReq_id	int(11)			No	None		Change Drop Primary
4	date_assign	varchar(30)	latin1_swedish_ci		No	None		Change Drop Primary
5	date_to_finish	varchar(30)	latin1_swedish_ci		No	None		Change Drop Primary
6	status	tinyint(1)			No	None		Change Drop Primary

Figure 19: Assign Analyst to User Table

 ISSN: 2321-8169
1173 – 1188

DEVELOPMENT PHASE

In this research work PHP, AJAX, jQuery Library and HTML programming languages were used. Hyper-Text Mark Up Language (HTML) and Asynchronous JavaScript and XML (AJAX) serves as the client side languages. Hyper-Text pre-processor(PHP) serves as the server side language, PHP is a widely used general purpose scripting language that is especially suited for Web development and can be embedded into HTML (**Larry Ullman, 2008**). jQuery library was imported to make the system efficient and fast.

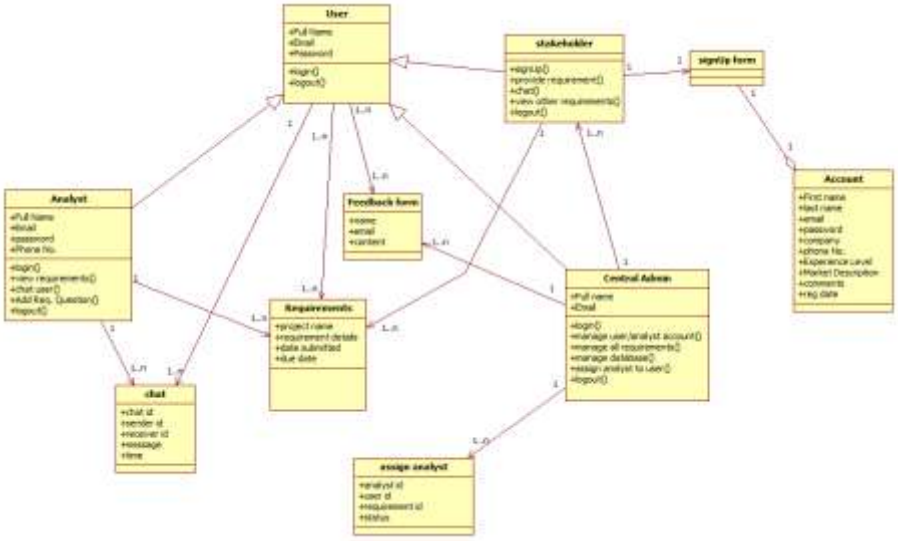

Figure 20: System Class Diagram

MAINTENANCE

Maintenance is making adaptation of the software for external changes (requirements changes or enhancements) and internal changes (fixing bugs). When changes are made during the maintenance phase all preceding steps of the model must be revisited. There are three types of maintenance:

- Corrective (Fixing bugs/errors)
- Adaptive (Updates due to environment changes)
- Perfective (Enhancements, requirements changes).

SNAPSHOTS

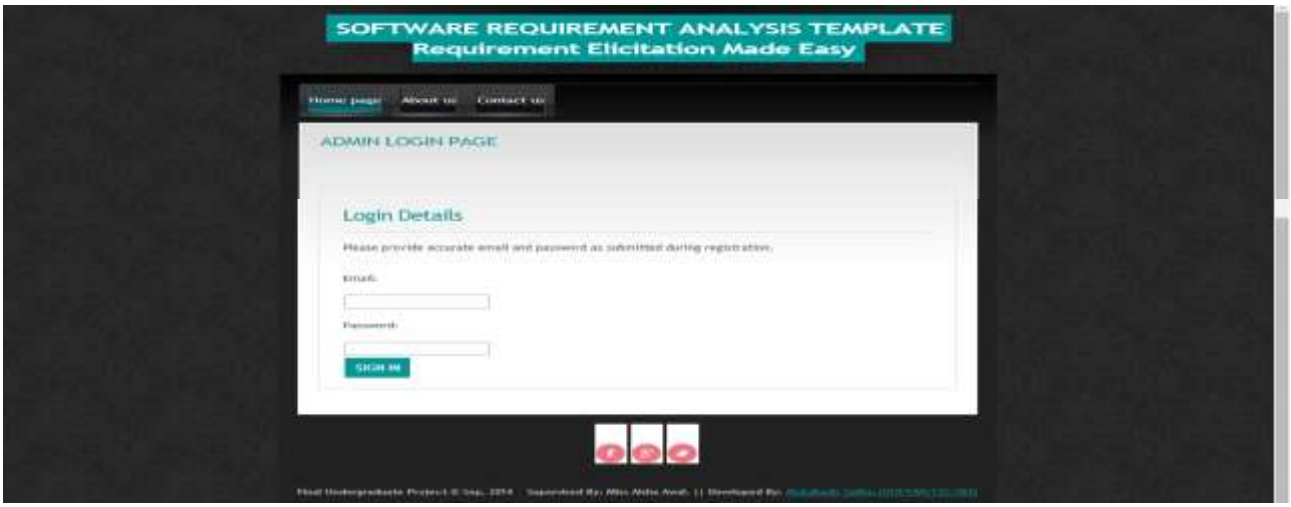

Figure 21 Central Administrator/Analyst Login Page

 ISSN: 2321-8169
1173 – 1188

Figure 22: Central admin Content Management System

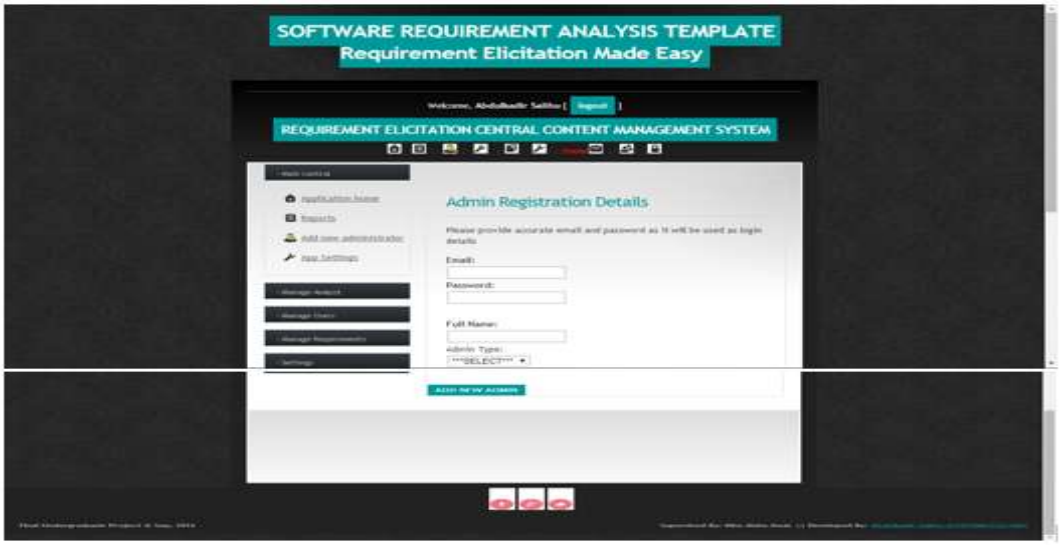

Figure 23: Central admin add new sub admin/analyst page

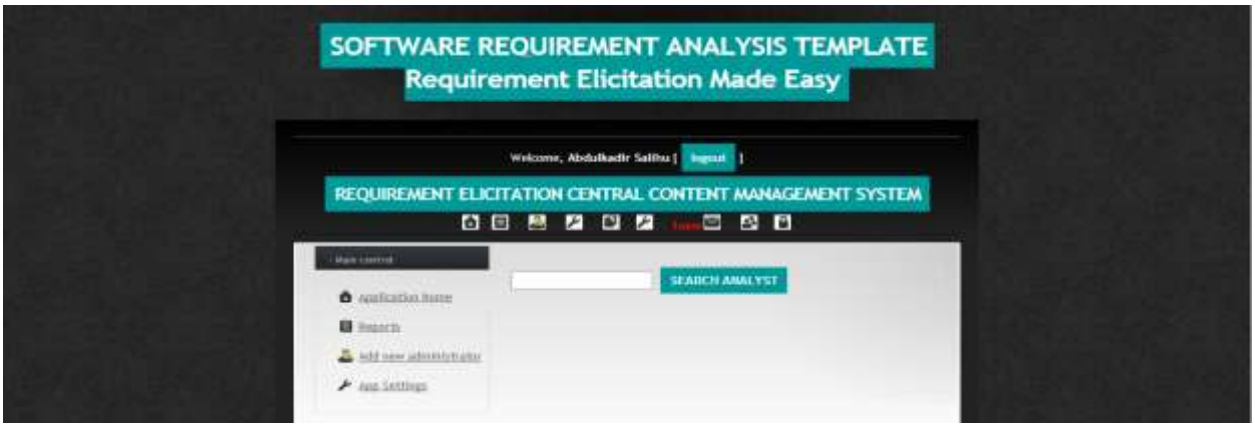

Figure 24 Central admin Search Analyst

 ISSN: 2321-8169
1173 – 1188

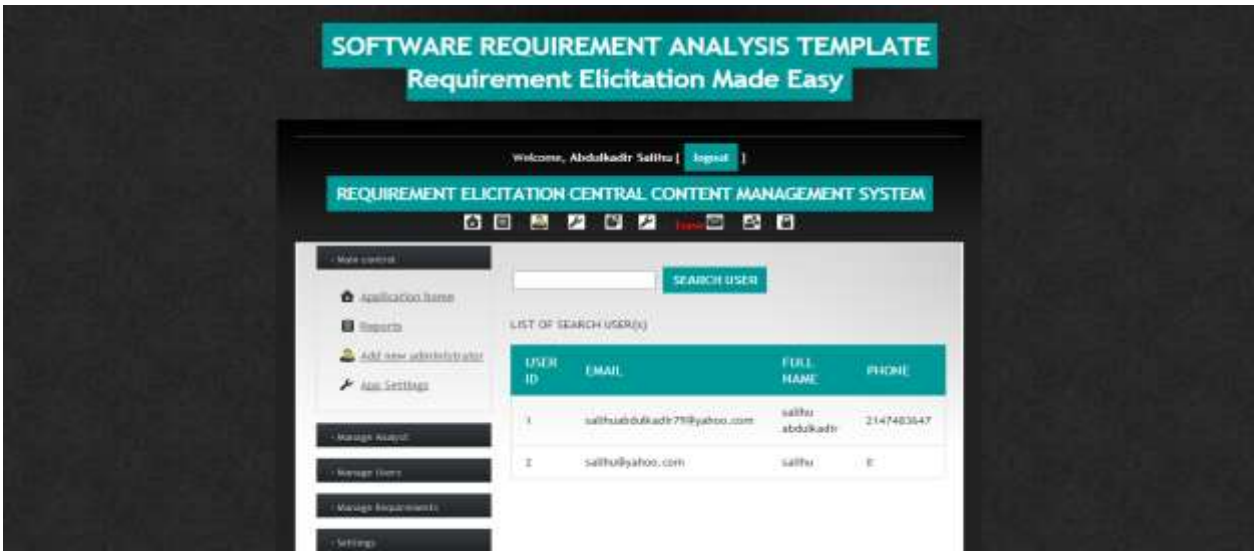

Figure 25 Central admin Search Users

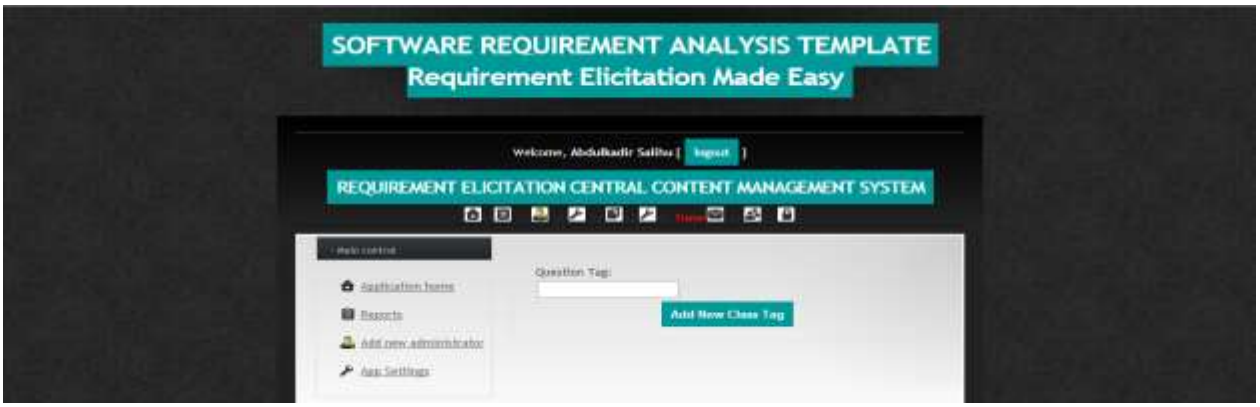

Figure 26 Add New Class Req. Question

Figure 27 Add Sub Class Req. Question.

Figure 28 All Registered Page.

 ISSN: 2321-8169
1173 – 1188

Figure 29: Block User Page

Figure 30Analyst Home Page

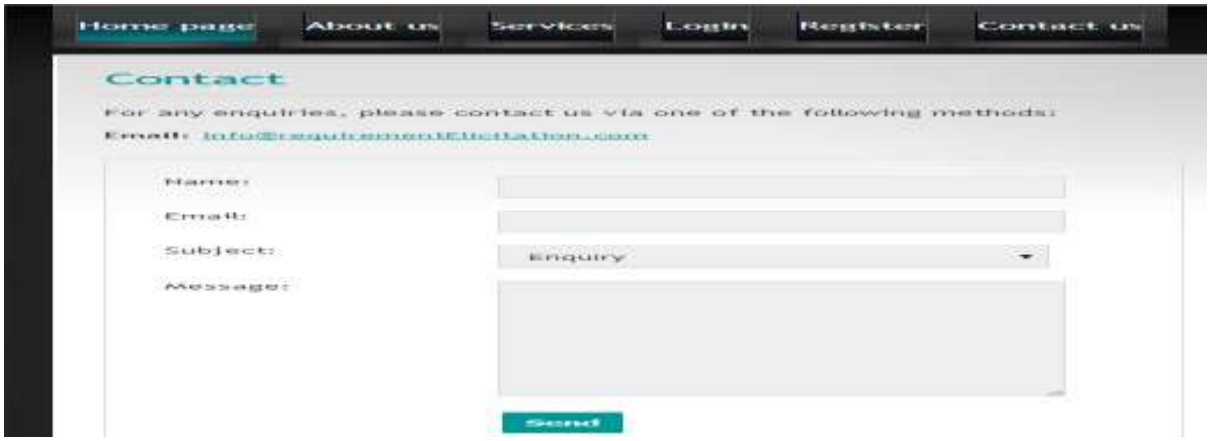

Figure 31: Contact Us Page

 ISSN: 2321-8169
1173 – 1188

Figure 32:User Homepage Slide

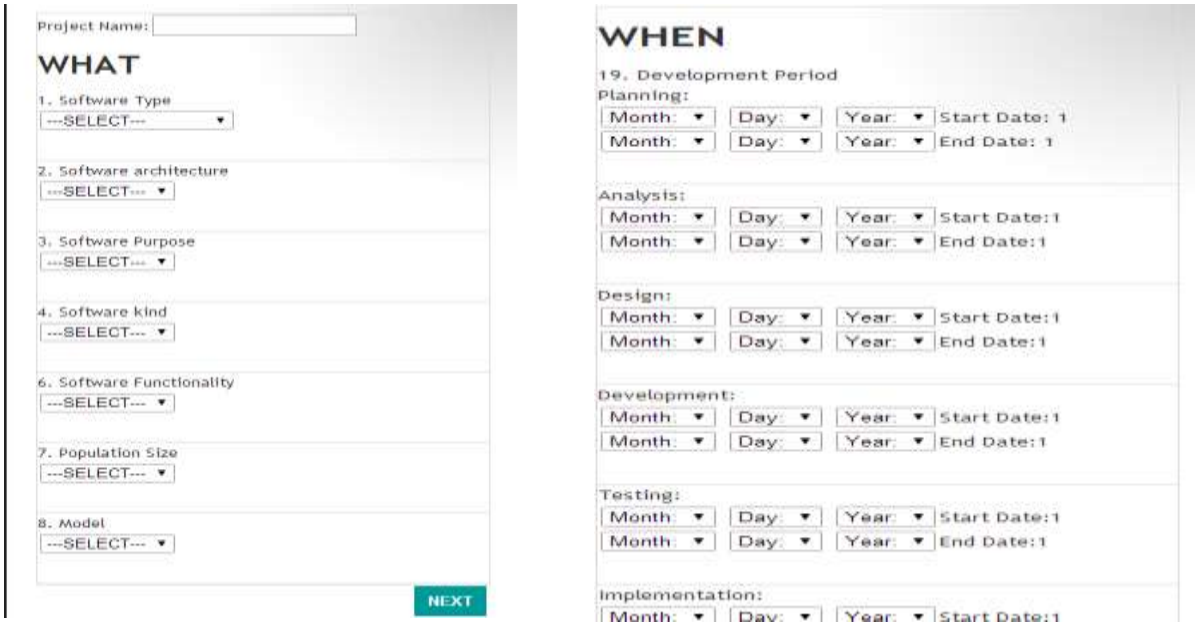

Figure 33: Class Requirement Question WHAT and WHEN

Figure 34: Class requirement Question WHERE and WHO

 ISSN: 2321-8169
1173 – 1188

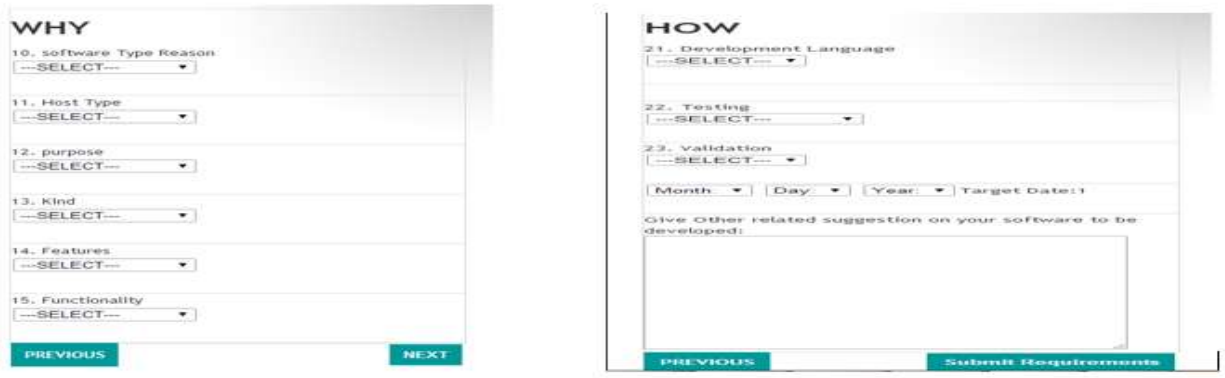

Figure 35:Class requirements Question WHY and HOW.

SUMMARY

In this study, an attempt has been made at computerizing the Requirement elicitation process. The challenge for requirement elicitation was studied; some literature was studied and reviewed based on theirstrengths and limitations.In the course of study, a relational database that is aimed at capturing user's related information and requirements was developed based on the repository. This database can be used not only for data collection and organization, but also for knowledge testing.

The following requirement where met:

1. An easy efficient report generating system for the management of user's requirement records.
2. A system that allows users to register and submit requirement on a system of network.
3. Eliminate the issue of inappropriate requirement specification.
4. An easy and efficient system for analyzing the requirement.

It is obvious that there is always a communication gap between the analyst and customer when it comes to taking requirement, the use of this kind of tool will foster easy requirement gathering. It is using the inquisitive technique or also known probing technique "5 W 1 H" method that leads to deeply understanding on the requirement. The tool has been developed with much care that it is free of errors and at the same time it is efficient and less time consuming. The important thing is that the system is robust. Also, provision is provided for future developments.

REFERENCES:

[1] Badariah, S. Shamsul, S. Ghani A.A.A. (2012). "A New Maturity Model for Requirements Engineering Process: An Overview Journal of Software Engineering and Applications, 2012, 5, 340-350.

[2] Chih-Wei Lu, C.-H. C., Chu, W.C., Ya-Wen Cheng, Hsin-Chien Chang (2008). "A Requirement Tool to Support Model-based Requirment Engineering." Computer Software and Applications, 2008. COMPSAC '08. 32nd Annual IEEE International 712 – 717.

[3] Cusumano, M. M., A.; Kemerer, C.F.; Crandall, B. (2003). "Software development worldwide: the state of the practice." Software, IEEE 20(6): 28 – 34.

[4] Fauziah Baharom, A. D., Abdul Razak Hamdan (2005). "A Survey on the Current Practices of Software Development Process in Malaysia" Journal of ICT 4: 57-76.

[5] Garcia Alcazar, E. M., A. (2000). "A process framework for requirements analysis and specification." Proceedings. 4th International Conference on Requirements Engineering, 2000. : 27 – 35.

[6] Hui zhou (2004), Soft Systems Methodology (SSM) in Information System Analysis, unpublished thesis, University of Missouri – St. Loius.

[7] Jabar, M.A.; Azizan, A,; Sidi, F; Ghani, A.A.A. (2012) "Software Requirement Analysis Template with automation aided system" Information Retrieval & knowledge management (CAMP), 2012 International conference pp 235-239.

[8] Kayed, A. H., N. Samhan, A.A. Alfayoumi, M. (2009). Towards an Ontology for Software Product Quality Attributes. Fourth International Conference on Internet and Web Applications and Services, 2009. ICIW '09.: 200 – 204.

[9] Kerry, E. Delgado, S. (2009). "Applying software engineering practices to produce reliable, high-quality and accurate automated test systems." AUTOTESTCON, 2009 IEEE.

[10] Pressman, Roger S. (2001) Software engineering: a practitioner's approach / Roger S. Pressman.—5th ed. p. cm.— (McGraw-Hill series in computer science).

[11] Rumbaugh, J., Jacobson, I. and Booch, G. (1999). The Unified Software Development Process. Reading, Mass.: Addison-Wesley.

[12] Sai Ganesh Gunda (2008). "Requirements Engineering: Elicitation Techniques." University Wes, Department of Technology, Mathematics and Computer Science, S-461 86 Trollhattan, SWEDEN.

[13] Sommerville, Ian Software engineering / Ian Sommerville. — 9th ed. p. cm. Includes index. ISBN-13: 978-0-13-703515-1 ISBN-10: 0-13-703515-21. Software engineering. I. Title. QA76.758.S657 (2011) 005.1—dc22.

[14] Sommerville, Ian: Software Engineering, Eight Edition (ISBN 13: 978-0-321-31379-9,ISBN 10: 0-321-31379-8).
[15] Zhang, Z.; Li Yanfang; Chen, C. (2009). "Software Requirement Analysis Research Based on Event-Driven." International Forum on Computer Science-Technology and Applications, 2009. IFCSTA '09. 1: 247 – 250
[16] http://en.m.wikipedia.org/wiki/Requirements_analysis
[17] http://en.m.wikipedia.org/wiki/Systems_analysis
[18] http://tutorialspoint.com/sdlc