International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 09 Issue 12 ǁ 2021 ǁ PP. 20-25

A Study of Requirement Engineering Methods for Small & Medium Enterprise Applications

Mohammed Musharraf Aamir[1], Shahadev B. Ubale [2], Dr. Razaullah Khan[3]
[1](Research Scholar in Mechanical Engineering, MGM's Jawaharlal Nehru Engineering College, Aurangabad, (MS), India)
[2](Associate Professor in Mechanical Engineering, MGM's Jawaharlal Nehru Engineering College, Aurangabad, (MS), India)
[3]Associate Professor, Department of Commerce & Management Science, Maulana Azad College Of Arts, Science & Commerce, Aurangabad, (MS), India)

ABSTRACT: *Requirement Engineering (RE) is the earliest phase in software development process. It is the most essential and crucial phase of software development process. The goal is to capture and understand the problem to be resolved. Designing and building an elegant software system will be of no use if it does not serve the purpose of the intended needs. The main views and descriptions of RE process are Requirement Elicitation, Requirement Modeling And Analysis, Requirement Specification, Requirement Validation and Requirement Management. RE is the first and most important process of development. All other processes depends on RE. If defects are left out through this phase the product further developed will not conform to the user requirement and cause problems during implementation.*
Even if the defects are found prior to implementation, during further phases of development or testing, it takes much larger cost, effort and time to resolve the defect. This cost, effort or time varies proportionately with the delay in detecting the defect. It has been found through experience of development that most of the software have the highest percentage of defects traced back to the requirement analysis phase. In this paper, we are going to study the various processes and methodologies for RE and evaluating and comparing the methods with focus on ERP software.
Keywords: *ERP, Enterprise Resource Planning Software, Requirement Analysis, Requirement Engineering, RE*

Date of Submission: 28-11-2021 Date of acceptance: 12-12-2021

I. INTRODUCTION

1.1 REQUIREMENT ENGINEERING

Requirement Engineering (RE) is the earliest phase in software development process. It is the most essential and crucial phase of software development process. The goal is to capture and understand the problem to be resolved. Designing and building an elegant software system will be of no use if it does not serve the purpose of the intended needs.

1.2 REQUIREMENT ENGINEERING PROCESS

This methodology was presented by Kotomia and Sommervile in 1998 in their work which describes five inputs to the process and three outputs to the process. The inputs are: existing system information, stakeholder needs, organizational standards, regulations and domain information. A detailed process is applied on these inputs and the agreed requirements, system specifications and system models are achieved as outputs. These inputs and outputs are similar for all organizations in most of the cases but only the requirements vary.
There are many views and description on RE process. In summary, it involves a set of the following activities:-

1.2.1 Requirement Elicitation

Requirement elicitation is a process of identifying the requirements of the problem to be resolved. It is the core process in RE which defines the details understanding of specific problem, the scope and the nature of the problem, criteria to be met and its constraints. The requirements are obtained from the stakeholders as the primary resources, careful analyses of the organization, the application domain and business process where the system will be used. Obviously those activities require intensive communication, collaboration and cooperation between requirement engineers and system stakeholders.

1.2.2 Requirement Modeling And Analysis

Requirement modeling phase focuses on developing a refined technical model of software functions, features and constrains. It can be used as an elicitation tool to further expand and refine the requirements. The goal of requirement analysis is to identify any discrepancy and conflict in the draft documents produced as the output of requirement elicitation. Modeling artifacts can be used as analysis medium to check and verify the gathered requirements and provide a mechanism to detect any uncovered conflicts in the requirements.

1.2.3 Requirement Specification

Requirement specification can be a written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype or any combination of those. The principals in expressing the requirements are that they must cover what the user expects, must be practicable, can be verified and validated and must include the properties of the machine and how its interacts with the environment – the hardware and people

1.2.4 Requirement Validation

Requirement validation is the last stage of RE process. It is the process of clarifying the requirement specification for consistency, completeness and accuracy. The output of this process is to get customer specification and acceptance on the formal and final requirement specification.

1.2.5 Requirement Management

Requirement management is the process of managing the life cycle of the requirements throughout the process of software development. It involves a process of documenting the requirement change, analyzing the impact of the requirement change, changing the specification, analyzing the cost incurred and implementing the change accordingly.

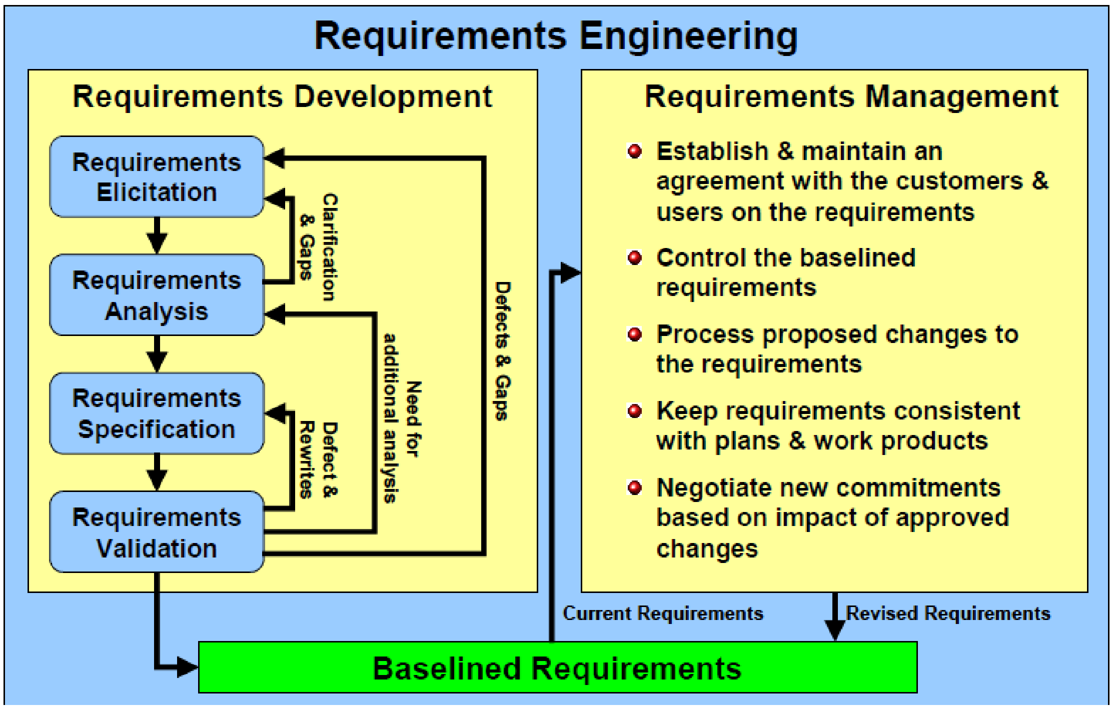

Requirements Engineering
Requirements Development
Requirements Elicitation
Requirements Analysis
Requirements Specification
Requirements Validation
Clarification & Gaps
Defect & Rewrites
Need for additional analysis
Defects & Gaps
Requirements Management
Establish & maintain an agreement with the customers & users on the requirements
Control the baselined requirements
Process proposed changes to the requirements
Keep requirements consistent with plans & work products
Negotiate new commitments based on impact of approved changes
Current Requirements
Revised Requirements
Baselined Requirements

Fig. 1 - Requirement Engineering Process

1.3 LEVELS AND TYPES OF REQUIREMENTS

Business requirements define the business problems to be solved or the business opportunities to be addressed by the software product. In general, the business requirements define why the software product is being developed. Business requirements are typically stated in terms of the objectives of the customer or organization requesting the development of the software. User requirements look at the functionality of the software product from the user's perspective. They define what the software has to do in order for the users to accomplish their objectives. Multiple user level requirements may be needed in order to fulfill a single business requirement.

As opposed to the business requirements, business rules are the specific policies, standards, practices, regulations, and guidelines that define how the users do business (and are therefore considered user-level requirements). The software product must adhere to these rules in order to function appropriately within the user's domain. User-level quality attributes are nonfunctional characteristics that define the software product's quality. Sometimes called the "ilities," quality attributes include reliability, availability, security, safety, maintainability, portability, usability, and other properties. A quality attribute may translate into product-level functional requirements for the software that specify what functionality must exist to meet the nonfunctional attribute.

The external interface requirements define the requirements for the information flow across shared interfaces to hardware, users, and other software applications outside the boundaries of the software product being developed.

The constraints define any restrictions imposed on the choices that the supplier can make when designing and developing the software. The data requirements define the specific data items or data structures that must be included as part of the software product.

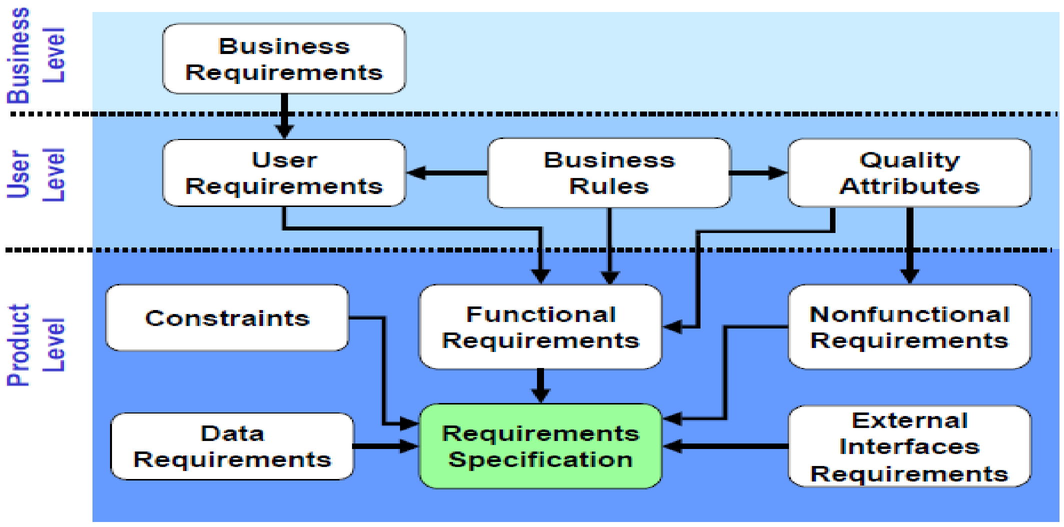


Fig. 2 - Levels and types of requirements

II. LITERATURE REVIEW

Elicitation is a very difficult process for many reasons: (i) Stakeholders may have a difficulty in expressing their needs, or they may ask for a solution that does not meet their real needs. (ii) Stakeholders can have conflicting demands. (iii) Users find it difficult to imagine new ways of doing things or to imagine the consequences of things they ask for. When they, for instance, see the system that has been built for them, they often realize that it does not fulfill their expectations, although it fulfills the written requirements.

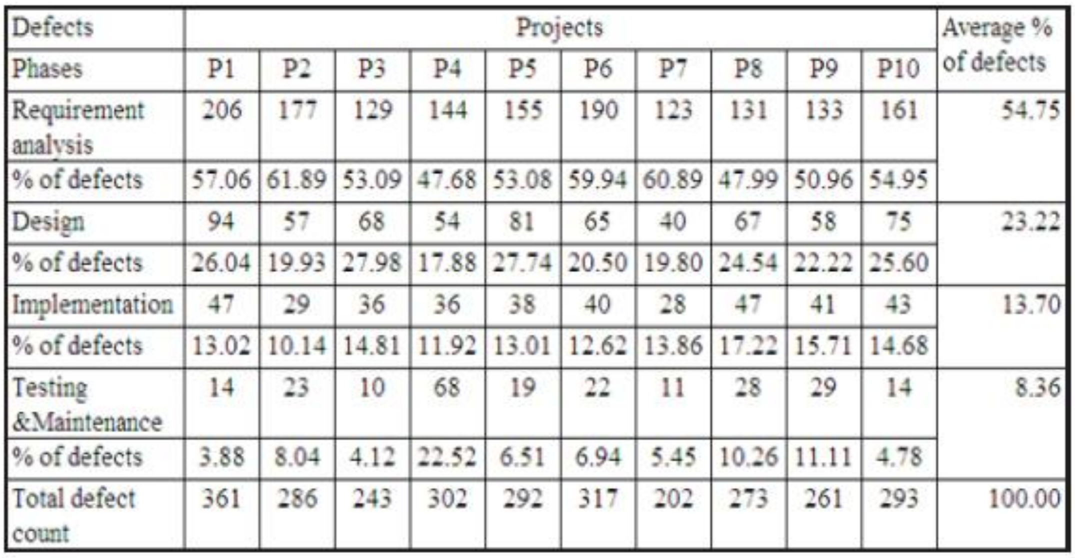
Even when users can express their needs, requirements engineers find it difficult to write them down in a precise way without designing the solution at the same time. The result is that the real demands and the written requirements do not match. For this reason, it is important for stakeholders to check that requirements meet their demands (Ref.[1] – Jamaludin Sallim, 2005).

The processes involved in RE include domain analysis, elicitation, specification, assessment, negotiation, documentation, and evolution. Getting high quality requirements is difficult and critical. In software-intensive systems, the achievement of qualities—such as performance, availability, security, and modifiability—is dependent on the software architecture. Quality should be measured from the early stages of software building or else the development can end up with software that fulfills the requirements but fails to satisfy the customer. High Quality Requirement documents are a must for successful software projects. All process phases directly and indirectly depend on the requirement document (Ref.[2] – Chander Diwaker, Kavita, Ajay Jangra & Shikha).

Requirements initially collected from customers or clients are termed as base requirements. Requirement creep or volatility is a term for adding requirements, once the project has started. Addition of requirements requests arise either from customer or from development team. However, it is well observed fact in industries that majority of additional requests come from customer end. It is worth to recall here that project plans are based on estimated number of base requirements and henceforth the project manager accordingly estimate the various other resources. Addition of requirements causes alterations in estimation and effects project plan. Therefore, requirement volatility is undesirable factor in project development process.

Requirements engineering phase is one of the major areas of research. If security requirements are introduced early in development cycle, ROI on the net profit of the business of company ranges between 12 to 21%. Further, according to the survey report, rework cost of defects due to security issues has consumed an annual economy of $59.5 billion. This research involves a deep study made on several projects developed in a leading CMMI level 5 and ISO certified product based Software Company as a case study.
The following table depicts a sample of complex projects which are developed in Java and .Net platform. These projects were engineered using iterative V model during software development process. (Ref. [3] – Dr.V Suma, B.R Shubhamangala, Dr.L Manjunatha Rao)

Table No 1 : Table of Defect Count during Software Development Process



Defects	Projects										Average % of defects
Phases	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	
Requirement analysis	206	177	129	144	155	190	123	131	133	161	54.75
% of defects	57.06	61.89	53.09	47.68	53.08	59.94	60.89	47.99	50.96	54.95	
Design	94	57	68	54	81	65	40	67	58	75	23.22
% of defects	26.04	19.93	27.98	17.88	27.74	20.50	19.80	24.54	22.22	25.60	
Implementation	47	29	36	36	38	40	28	47	41	43	13.70
% of defects	13.02	10.14	14.81	11.92	13.01	12.62	13.86	17.22	15.71	14.68	
Testing &Maintenance	14	23	10	68	19	22	11	28	29	14	8.36
% of defects	3.88	8.04	4.12	22.52	6.51	6.94	5.45	10.26	11.11	4.78	
Total defect count	361	286	243	302	292	317	202	273	261	293	100.00

Based on the experimental results, it is recommended that software practitioners should be made aware of process patterns and uses them during the requirement analysis phase. As the requirement analysis phase is a major and key activity in software development practice, an improvement in this phase in terms of product and process would have a direct and significant impact on the quality of the resulted software project and application. (Ref. [4] – Ahmad Estabraghy, Darren Dalcher)

Formal methods were born and have grown up in those years of structured analysis and design mainly for justifying, better understanding, and more precise and rigorous use of techniques in parts of a “water-fall model” of a development process. So we have theories of formal specification, verification, refinement, decomposition and composition. These have helped in improving the quality of software systems developed so that they are more correct and safer to use. On the other hand, formal methods have inherited the same disadvantages from the informal use of the “water-fall model” of the structured analysis approach, and they suffer even more seriously from these disadvantages as a specification of the whole system at any level, e.g. the requirement level, in a formal notation is not understandable to most system engineers, not to mention about formal verification. This may be one of the main reasons why the use of formal methods cannot be scaled up and widely accepted in large scale software developments. (Ref. [5] – Xiaoshan Li, Zhiming Liu, Jifeng He)

The key to software development depends on the result of requirement analysis. Traditional analysis method starts from function. But users usually do not know computer or what the software developers really need, it is very difficult for them to describe the future system function. At the same time, most software developers don’t know business processes of users and where to do research and require, so difficult to summarize the requirements of system. This paper proposed a requirement analysis method based on events, which is creating event table of system, namely, collect all the events that system may be encountered and build events table, which constitute the system requirement information. (Ref. [6] – Zhang Zhaoyin, Li Yanfang ,Chen Chao)

The insufficiency of requirement analysis is the important factor for a failure software development. If the defects of the requirement analysis were delayed to the test procedure to solve, the cost of correct would be dozens or hundreds times of the correct cost in the requirement, or resulted to the failure of the development. Therefore, to eliminate the defect in the requirement analysis is important to reduce the risk of the software development for the successful development. (Ref. [7] – Zhifeng Zhang, Yuxi Liu)

With the globalization of software development, software requirement elicitation is also becoming distributed. The distributed requirement elicitation achieves larger elicit basis, and therefore leads to more accurate requirement analysis result. However, distributed elicitation also gives rise to more requirement inconsistencies and makes researches of requirement inconsistency more significant. *A new requirement inconsistency analysis method is presented to solve the problem of inconsistent requirement description in the process of requirement analysis.* This method is still fragile to complex requirement inconsistency because of the immaturity of composite ternary logic operations. We will continue the research especially on the composite ternary logic operations in our future work. (Ref. [8] – Zhang Yikun, Yin Peng, Cui Duwu, Xia Hui)

Among those domain requirements elicitation and specification approaches, there are mainly three types. One is the traditional method. Domain requirements are presented in natural language. The main drawback of this method is that there are gaps between domain users and requirements engineers. Domain users can’t communicate the requirements in the format that requirements engineers want. This will be the main source of generating inconsistent and ambiguous requirements. The second method is the scenario-based method. Domain users can take part in elicitation conveniently, and the requirements are the true reflection of the real system. But the disadvantage of it is the problem how to guarantee a set of scenarios that are the complete requirements of the system. The overlap of different scenarios is the source of requirements inconsistency. The third solution is knowledge-based methods. The methods have been researched hotly, but there are few sound achievements yet. In the third methods, people usually use feature-based and ontology-based methods to research. However, they have some disadvantages and limitations. For example, the feature-based method concentrates on commonality and variability analysis having a shortcoming in reasoning logic, while ontology-based method focusing on representing domain concepts and relations of concepts lack of distinguishing concepts to common or variation sets.

In this paper, mutual exclusion is the prediction relation of different sub problem domains. In practice, subjects may crosscut many aspects and modules of a domain so that intersection often exists. Our future work is to improve the approach to deal with more complex situations, to combine the domain model with other kinds of models to specify domain requirements better, and to develop a propose to map the domain model into reusable architecture and components. (Ref. [9] – YaJun Liu, Lisha Gao, XiaoJie Feng)

In software development, it is important to mediate various concerns coming from user experience (UX) designers and application developers. In Agile User-Centered Design (Agile-UCD), there is a special role called specialist who is dedicated to implement application features as well as to monitor user experiences. However, the specialist normally has difficulty in linking user tasks to be accessed via a user interface (UI) into application feature entities. In addition, the specialist may also have some unsettled usability risks that might result in the failure of meeting certain usability criteria and passing acceptance tests.

Even though our approach has successfully resolved the three main problems of the Agile-UCD in requirement specification phase, there is a need for modeling training to seamlessly apply the approach into practices. In addition, even though this approach satisfies the visual transforming and expressing the user tasks into application features, some of the user tasks are hardly ever supposed to convert right application feature of entities. (Ref. [10] - Sang-Hyun Lee, In-Young Ko, Sungwon Kang, Dan-Hyung Lee)

III. ISSUES WITH REQUUIREMENT ENGINEERING

- RE is the first and most important process of development. All other processes depends on RE. If defects are left out through this phase the product further developed will not conform to the user requirement and cause problems during implementation.
- Even if the defects are found to implementation, during further phases of development or testing, it takes much larger cost, effort and time to resolve the defect. This cost, effort or time varies proportionately with the delay in detecting the defect.
- It has been found through experience of development that most of the software have the highest percentage of defects is traced back to the requirement analysis phase.
- The defects might be arising due to incomplete information provided by the user, misinterpretation on the part of the analysts, assumptions made, ambiguity in specification of process details, etc. Requirement engineering is required to be thoroughly performed in order to avoid such defect.
- Methods for RE are not completely reliable. Most of the methods complicated and difficult to understand for user of project. Apart from this, in majority of cases, the work in requirement analysis is performed manually without the aid of any automated tools.
- Major complications arise in ERP package in manufacturing enterprises due to the huge number of processes to be handled, complicated formats of product data, detailed analysis of data, too many possible deviations in the processes apart from the regularly adopted way of work, large number of process parameters affecting the data handling operations, etc.

IV. CONCLUSION

- ERP systems and similar data management software fail due to improper requirement engineering.
- Requirement engineering is the crucial phase of system development life cycle.
- The suitability of the existing RE methods for their applicability in development of ERP and data management software has to be studied.
- More systematic methods should be developed with the purpose of eliminating or reducing problems in RE.
- The methods should be simple enough to be used in small and medium sized companies, involving less complexity and skill on the part of the analysts.

REFERENCES

[1]. Jamaludin Sallim, Requirements Engineering for Enterprise Applications Development: Seven Challenges in Higher Education Environment, *World Academy of Science, Engineering and Technology 4* 2005.

[2]. Chander Diwaker, Kavita, Ajay Jangra & Shikha, An Efficient Framework for Requirement Engineering, *International Journal of Computer Science & Communication Vol. 1, No. 2, July-December 2010,* pp. 317-320.

[3]. Dr.V Suma, B.R Shubhamangala, Dr.L Manjunatha Rao, Impact Analysis Of Volatility And Security On Requirement Defects During Software Development Process, *International Conference on Software Engineering and Mobile Application Modelling and Development (ICSEMA 2012) Year: 2012 :* Pages: 1 - 5, DOI: 10.1049/ic.2012.0145.

[4]. Ahmad Estabraghy, Darren Dalcher, A Controlled Experiment to Investigate the Effect of 'Process Patterns' on the Quality of Requirement Analysis, *IEEE 1-4244-1031-2/07/$25.00©2007.*

[5]. Xiaoshan Li, Zhiming Liu, Jifeng He, Formal and Use-Case Driven Requirement Analysis in UML, *IEEE 0-7695-1372-7/01 $10.00 0 2001*.

[6]. Zhang Zhaoyin, Li Yanfang ,Chen Chao, Software Requirement Analysis Research Based on Event-driven, *IEEE 978-0-7695-3930-0/09 $26.00 © 2009 : DOI 10.1109/IFCSTA.2009.66.*

[7]. Zhifeng Zhang, Yuxi Liu, Application of Active Learning Strategy and Formalization Method in Requirement Analysis, *IEEE Symposium on Robotics and Applications(ISRA) 978-1-4673-2207-2/12/$31.00 2012*.

[8]. Zhang Yikun, Yin Peng, Cui Duwu, Xia Hui, A Method of Requirement Inconsistency Analysis, *IEEE : 31st Annual International Computer Software and Applications Conference(COMPSAC 2007)0-7695-2870-8/07 $25.00 :* 2007.

[9]. YaJun Liu, Lisha Gao, XiaoJie Feng, An Ontology Modeling Methodology in Requirement Analysis, *IEEE : 3rd International Conference on Biomedical Engineering and Informatics (BMEI 2010) 978-1-4244-6498-2/10/$26.00* ©2010.

[10]. Sang-Hyun Lee, In-Young Ko, Sungwon Kang, Dan-Hyung Lee, A Usability-pattern-based Requirements-analysis Method to Bridge the Gap between User Tasks and Application Features, *IEEE 34th Annual Computer Software and Applications Conference 0730-3*157/10 $26.00 © 2*010 IEEE : DOI 10.1109/COMPSAC.2010.39*.