International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 3 | ISSN : 2456-3307

A Review on Requirement Engineering in Rapid Application Development

P. Maheshwaran, Rahul Kumar, S. Rajeswari, Dr. Jitendranath Mungara
Information Science Engineering Department, NHCE, Bengaluru, Karnataka, India

ABSTRACT

Time shortage has led to a lot of changes in the process of software development. Rapid Application Development is one of the most widely used strategies for Software Development with these limitations in place. But with these developments there have been cases of failures in projects and the success rate has fallen. This is a serious damage when it comes to the IT industries. The system proposed improves the Requirement Engineering phase of the paper and improves the development strategy by a lot of paces. This system is effective in terms of time, cost and is well managed in terms of reducing errors and risks.

Keywords: Software Development, Rapid Application Development, Requirement Engineering Phase.

I. INTRODUCTION

Software development is an expensive process due to the entity of risk that appears as a part of all software development processes. These risks arise when there are constraints to creating high quality software within a definite time. Understanding these constraints and clearly understanding the problem statement from all viewpoints is an important part of SDLC. These risks are magnified in RAD to high time constraints and reduced clarity in the understanding the problem. The consequence of these risks in any SDLC is the failure of the project. A systematic understanding of the problem domain, powerful risk management tools and standardised practices help to minimize failures. The analysis of risk is to be done at every stage in the SDLC to minimize the costs such as correction, redeployment, etc. of the project.

II. METHODS AND MATERIAL

The first stage in any SDLC is requirement engineering and is a crucial step as this helps to understand the problem domain and provides a consolidated view of the expected product. It deals with problems, goals, constraints and functions of the real world. Requirement engineering is not a single step process, but indeed consists of 5 major activities [2]

1) Requirement Elicitation – Involves collection of requirements from various stakeholders. The key objective of this step is to retrieve the problem statement from the customer

2) Requirement Analysis – Involves analysing the requirements from the previous step for completeness and consistency

3) Requirement Specification – This provides blueprint for software development process. It control both the process of specification and validation

4) Requirement Verification and Validation – This step results in the proof that the declared requirements meet the customer's needs.

5) Requirement Management – This is the last step that involves the management of changes in the requirements: adding, deleting, and updating.

CSEIT172375 | Received : 13 May 2017 | Accepted : 07 June 2017 | May-June-2017 [(2)3: 742-746]

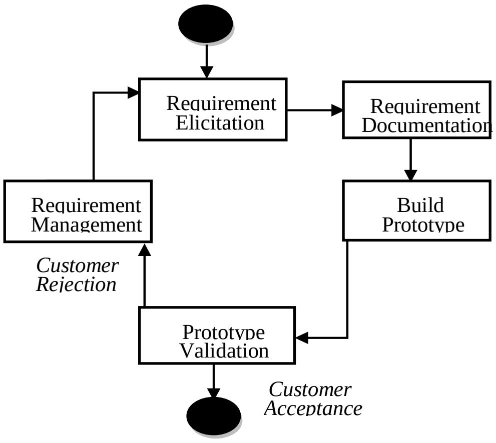

Figure 1. Existing System

Rapid Application Development is a software life cycle that permits organisation to deliver a product faster while reducing cost and time. This mainly uses prototyping model that produces faster unit results and collects response from the customer for further developments

According to James Martin, RAD consists of the following four stages [3]

1) Requirements Planning – Requirements are taken from customers and plan all requirements.

2) Design – The preparation of design of system by analysing the requirements ensuring that the design meets the customer requirements

3) Construction – Design and Construction work in parallel. Construction involves implementation of the design.

4) Cutover – This is the final stage where testing is done. It ends in deployment and maintenance of the software product.

RAD also leads to the following which are referred to as the silent features [2]

1. Cost Effective
2. Time Effective
3. Fast SDLC
4. Life cycle consists of 60 to 90 days
5. Achieves high customer satisfaction
6. Good Risk Management
7. Reduce Developer's Risk

The Requirement Engineering phase can be restructured (to suit RAD) as proposed by Shoaib Hassan, Usman Qamar, Muhammad Arslan Idris as follows [2]

1. Requirement Elicitation by JAD(Joint Application Development)
2. Analysing the requirements and Conflict Management
3. Requirement Prioritisation
4. Requirement Documentation
5. Building Prototype
6. Validation by Customer
7. Requirement Change Management

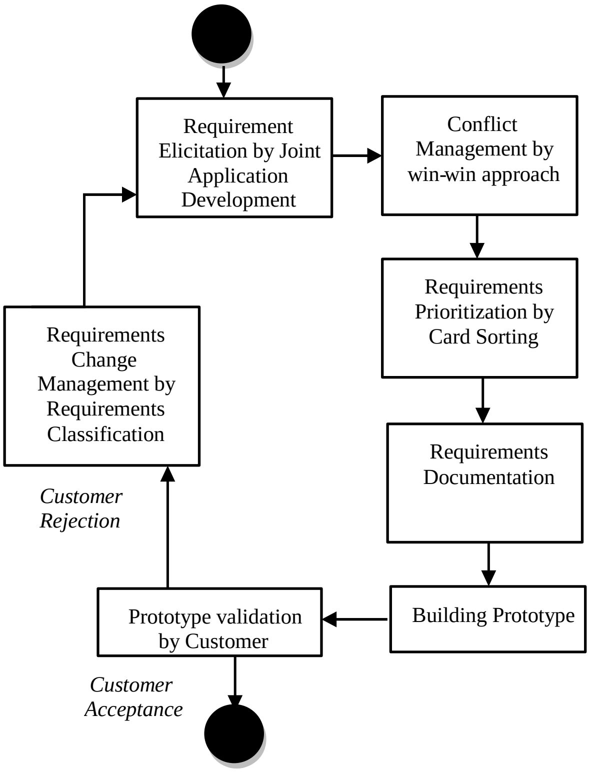
Requirement Elicitation by Joint Application Development
Conflict Management by win-win approach
Requirements Prioritization by Card Sorting
Requirements Change Management by Requirements Classification
Requirements Documentation
Customer Rejection
Prototype validation by Customer
Building Prototype
Customer Acceptance

Figure 2. Modified System

A. Steps in Proposed Requirement Engineering Model

1) There are many techniques for requirement elicitation, but we suggest JAD (Joint Application Development) for eliciting requirements in rapid application development's life cycle. JAD is workshop of 20 to 30 team members. Stakeholders from all domains are present in JAD. In JAD, each stakeholder defines his requirements. JAD covers all the requirements from different domain stakeholders.

2) After eliciting requirements by JAD, there is a high need of analysing the requirements. Analysing the

requirements is necessary for ensuring requirements completeness and consistency.

3) After removing all the conflicts of requirements, there is a high need of prioritizing the requirements. Traditional requirement engineering model for rapid application development has not any requirement prioritization mechanism. But there is a high need of requirement prioritization in this life cycle. Because we want to make a prototype. And in prototype, we firstly show the major functionalities of the system. For this, we should prioritize requirements.

4) After prioritizing requirements by card sorting, there is a high need of documenting those requirements in a well-mannered way. Documentation makes an agreement between stakeholder and organization. There are many techniques for requirements documentation like decision table based specification, software requirement specification, natural language specification etc. but we suggest software requirement specification for documenting requirements.

5) Prototype building is main task of rapid application development's life cycle. Due to prototype development, it is often called prototyping model. Prototype is an executable model that reflects the original system. In prototype development, we firstly build major functionalities of the system. At this step, requirement prioritization gives us benefit of providing major functionalities. So using JAD, win-win approach and card sorting will help us to make better prototype which reflects the real customer needs.

6) Validation is most important step in any development life cycle. In RAD, prototype validation is very necessary for success of final product. In RAD when we do validation of our prototype, customer check the prototype with documented requirements. If prototype matches with requirements then prototype will valid and we move towards our design phase. But if customer rejects prototype and ask for enhancements in prototype then we move forward to requirement management. This step is same in traditional model and our proposed model.

7) If customer rejects prototype then we move to this step. If customer wants changes or enhancements in prototype then we follow this step. No specific technique was defined for requirement management of rapid application development based projects in traditional model. There are many techniques for requirement management but we have proposed requirements classification model.

8) In requirement classification technique, we categorize our requirements in three categories. These three categories are less likely to change requirements, fixed requirements and most likely to change requirements. When customer change requirements then we check requirements by our change requirements repository. If changed requirement category matches with repository then we change our requirements otherwise we ignore this step. It is simplest and customer's oriented technique for requirement management.

III. RESULTS AND DISCUSSION

After proposing the model, we need to implement that model for ensuring either this model will be useful or not. For the validation of our model, we choose survey technique. We made survey form regarding requirement engineering techniques and filled it from different software professionals to know their opinions about requirement engineering techniques. Their answers help us to validate our proposed model. Survey forms have filled from 50 experienced software professionals of 20 different software organizations. Their answers have validated our proposed model. Out of 50 software engineers, 36 engineers are agree or strongly agree on using joint application development technique as requirement elicitation technique for rapid application development's requirement engineering process. 8 engineers gave opinions as neutral on joint application development technique. And 6 software professionals were disagree on joint application development technique. Approximately 72% software professionals are in the favour of joint application development technique. 16% software engineers gave opinion as neutral. And 12% software engineers are disagreeing on using joint application development as requirement elicitation technique for rapid application development. We can see results by following results.

Table 1. SURVEY RESULTS OF REQUIREMENT ELICITATION TECHNIQUES [2]

Step #1	**Techniques**		**Results**	
		Strongly Agree/ Agree %	**Neutral %**	**Strongly Disagree/ Disagree %**
	Joint Application Development	72%	16%	12%
	Interviewing	66%	18%	16%

Requirement Elicitation	Brainstorming	58%	32%	10%
	Questionnaire	42%	50%	8%
	Prototyping	38%	46%	16%
	Reuse Requirements	30%	12%	58%
	Scenarios	24%	30%	46%
	Focus Group	18%	16%	66%
	Ethnography	12%	68%	20%
	Social Analysis	8%	54%	38%
	User Centred Design	6%	16%	78%

Table 2. SURVEY RESULTS OF REQUIREMENT ANALYSIS TECHNIQUES [2]

Step #2	Techniques		Results	
		Strongly Agree/ Agree %	Neutral %	Strongly Disagree/ Disagree %
Requirement Elicitation	Conflict Management	76%	18%	6%
	Card Sorting	70%	28%	2%
	State Machine	56%	28%	16%
	Fault Tree Analysis	48%	18%	34%
	Viewpoint Based Analysis	42%	8%	50%
	Goal Oriented Analysis	26%	32%	42%

Table 3. SURVEY RESULTS OF REQUIREMENT MANAGEMENT MODELS [2]

Step #3	Techniques		Results	
Requirement Management		Strongly Agree/ Agree %	Neutral %	Strongly Disagree/ Disagree %
	Requirements Classification Model	68%	8%	24%
	Ince's Change Model	52%	12%	36%
	Olsen's Change Model	38%	22%	40%

Table 4

AVERAGE RESULTS

Model	Techniques		Results	

Functions	Traditional Requirement Engineering Model of RAD	Proposed Purified Requirement Engineering Model for RAD
Requirement Elicitation by JAD	NO	YES
Conflict Management by win-win Approach	NO	YES
Requirement Prioritization by card sorting	NO	YES
Requirement Documentation by SRS	YES	YES
Requirement Validation by Prototyping	YES	YES
Requirement Management by classifying requirements	NO	YES

		Strongly Agree/ Agree %	Neutral %	Strongly Disagree/ Disagree %
Purified Requirement Engineering Model for Rapid Application Development	Joint Application Development for Requirement elicitation	72%	16%	12%
	Conflict Management	76%	18%	6%
	Requirement Prioritization by Card sorting	70%	28%	2%
	Requirement Classification for Management	68%	8%	24%
	Cumulative Average of all proposed steps	71.5%	17.5%	11%

Table 5
COMPARISONS

IV. CONCLUSION

We purified requirement engineering model by specifying joint application development for requirement elicitation in our proposed model. We also introduced conflict management and card sorting in requirement analysis steps in our proposed model for ensuring requirements completeness and prioritization. We also introduced requirement classification technique for managing requirements in a well-mannered way. Results show that our proposed model is more cost and time effective than traditional model. It achieves maximum satisfaction level of customer because it is customer oriented model. Our proposed purified requirement engineering model purified each step of traditional model. It clears the picture of requirement engineering process to software developers and encourages the participation of all stakeholders. From above discussion, we conclude that requirement engineering is most important factor for the success of any project and our proposed purified requirement engineering model will remove all the difficulties that we were facing while during requirement engineering of rapid application development based projects.

This proposed methodology provides a better structure to Requirement Engineering for the RAD methodology as opposed to the traditional method. This incorporates the change process and modification as proposed by validations with the customer. This provides a complete understanding of the problem domain and also reduces errors on behalf of the engineering team.

V. REFERENCES

[1]. Sergey M. Avdoshin, Elena Y. Pesotskaya, Software Risk Management. In: IEEE, 2011.

[2]. Shoaib Hassan , Usman Qamar , Muhammad Arslan Idris, Purification of Requirement Engineering Model for Rapid Application Development. In: IEEE, 2015.

[3]. Ian Sommerville, Software Engineering, Pearson Education, 8th Edition (2011).

[4]. B. Prashanth Kumar, Y. Prashanth, Improving the Rapid Application Development process Model. In:IEEE, 2014.