MITIGATION STRATEGIES TO MAJOR CHALLENGES FACED IN MARKET DRIVEN DEVELOPMENT SCENARIO

Ranjith Engu[1] and Sairam Vakkalanka[2]
[1]Ranjithnot4u@gmail.com
[2]Sairam.vakkalanka@gmail.com
[1&2] School of Computing, Blekinge Tekniska Högskola,
371 79 Karlskrona, Sweden.

ABSTRACT

This report explains about three major challenges that are faced in market driven requirement engineering with feasible solutions. Also, discusses the circumstances and implications organizations face due to these challenges. Validation and analysis of the solutions is provided with our own experiences from industry and literature study, discussing the merits and demerits.

Keywords

Market driven requirements engineering, Market driven development, MDRE, Requirements volatility, Value Estimation method

1. INTRODUCTION

As a business becomes more successful, the client base tends to grow larger and becomes more demanding [12]. For instance, it may focus on larger customer group, may require online access to order and require a new avenue of access to the system [12]. Area where development organizations produce a software to a market rather to an specific customer is known as market driven requirement engineering which is also known as MDRE [10] [29].As it focused to a whole niche market, it becomes a straight target to overcome many challenges like Time to market, Gap between market staff and developers, Release planning, Requirements volatility and writing understandable requirements. Some efforts have been made by researchers to address some of these challenges by proposing new techniques and models [10].In this environment there are no definite set of users for a product. "Here there are only imagined group of users who may fit into the profile of an intended product user"[11].Major difference within the characteristics of market driven environment and customer specific market occurs while eliciting the requirements from the users which are generally managed through marketing, user groups and technical support[11].As there is an increased demand for packaged software which is aimed at large set of customers many organizations are showing interest in MDRE which is also known as Commercial-Off-the-Shelf (COTS)[14].

Activities in market driven requirement engineering are not only limited to development instance but play a vital role in every phase and part of product management [13]. Major hindrances to engineer the Market driven requirements are discussed in the following sections providing the circumstances in which these problems arise, their implications and the solutions to these challenges.

2. MAJOR CHALLENGES FACED IN MARKET DRIVEN DEVELOPMENT :

2.1 Requirements Volatility C_1:

Requirements volatility is defined in many ways such as “frequent changes in requirements”[15] and “the ratio of number of requirement changes to the number of requirements for a given period of time” [15]. It is a known fact that, the desires of humans change rapidly according to the situation and needs. In this context, the requirements of the customer change from time to time according to the needs and situation. For example the usage of two mobile phones was a hectic problem some years ago; in order to fulfil the needs of the customer dual sim mobiles were released. In MDRE, as the product is developed based on the requirements of large customer groups (market), organizations have to face many changes in the requirements which may or may not fulfil the needs of the customer. These requirements changes are inevitable but organizations should manage the requirements according to the changes and changes must be done to the existing requirements in order to satisfy the customer.

2.1.1 Circumstances in which C_1 occurs:

In MDRE the requirements are gathered from different sources and the organizations focus on different types of customers. Different customers have different requirements, in order to fulfil all the requirements organizations have to carry out a detailed analysis of these requirements, and project these requirements into the product to satisfy the customer. From the studies [1][2][4], it is clear that requirements volatility is one of the major problem faced by many organizations during the development process. There are many factors that contribute to the changes in the requirements. Some of the factors are listed and explained below.

- Customers using the beta versions of the product may have a desire of something new in the product.
- The situation in which the system is used changes.[15]
- Continuous flow of customer requirements.[18]
- Competitors products (changes in technology).[18]
- Changes in organizational goals and policies
- Needs of the customers are triggered from current releases.
- Poor domain knowledge.

❖ **Customers using the beta versions:**

Most of the organizations are now a day’s releasing the beta versions of the product to check whether the features in the full version of the product satisfies the customer requirements with this beta products customers provide the missing features and additional requirements with which actual product is released .These missing and additional requirements are added to the requirements document for the development of the original product. Taking a real time example, Microsoft has released the beta version of internet explorer 8 and provided a feedback form to its customer. This process of development is very expensive as the requirements are modified and implemented and it’s a time consuming process.

❖ **The situation in which the system is used changes**

The behaviour of the product may change according to the situation with which customer may be left with no satisfaction. In order to overcome this kind of situations organizations have to plan the requirements such that it satisfies the customer in every situation. For example car battery gets freeze in the winter which poses a problem to start the engine; it is the responsibility of the organization to develop a battery which works fine in any season.

❖ **Continuous flow of customer requirements:**

Satisfying the needs of every customer is impossible in market driven development Different customers have different needs these changing requirements are induced into the product and released in different versions by the organizations.

❖ **Needs of the customers are triggered from current releases.**

When a customer uses the currently released product the missing features are identified by the customer either based on design, functionality or quality of the product. These requirements should be changed in the next release of the same product or different version of the product through which customer satisfaction can be gained. Though the previous missing requirements are fulfilled with the current release of the product there is a situation where the customer requires more requirements in that product. For example

❖ **Changes in organizational goals and policies.**

In order to meet the customer requirements sometimes the goals and the policies of the organizations are changes by the management. This change management leads to the changes in the requirements of the project. For example the timelines of the project are changed according to the requirements gathered and in order to attain the customer satisfaction the goals of the organizations are changed sometimes.[16]

❖ **Competitor Products:**

When a product is developed, the functionalities of the products are imitated by other companies [17] along with some new features added to it through which there is a chance of losing the grip of market on certain product. The requirements have to be changed in accordance to the market needs and new requirements have to be added which overcomes the competitor product.

Though the above factors cause the changes in the requirements but due to these changes the overall project development will be affected.

Not only these there are many causes for the requirements changes, As explained in the literature [25] there are internal and external factors in an organization which causes changes in the requirements. Some of them are stated below:

- *Internal Factors*
    - Requirements Overload
    - Software (price changes) and hardware
    - Requirements diversity (the extent to which stake holders disagree among themselves deciding on requirements).
    - Poor communication between users and development team.
    - Less Feedback from other SDLC phases.
- *External Factors*
    - Government regulations.
    - Market competitors.

2.1.2 Implications of C_1 :

It is a difficult task for the organization to manage all the requirement changes. Planning the changes in the requirements plays a vital role as the requirement changes are not often constant and tend to change not only at the beginning of the development process but also throughout the lifecycle. If these changes are frequent then they may produce significant project uncertainty. These requirement changes affect the project cost, time to market and quality of the product [16], for this reason organizations following MDRE release the products in different versions correcting the major and minor errors in the product. These changes are made for the product to sustain in the market for a long time. Considering a real time example of tata motors pvt Ltd which is an Indian car manufacturing company has released nano car according to the requirements of the customer but in order to provide the quality product they have made some changes to the requirements with which the cost of the car has been increased along with the delay in time to market. Not only may this, but the changes in the requirements result in developing innovative products and also failure of the innovative products or services. For example, evolution of computers from traditional desktop computer to the present surface computer from Microsoft is a success. On the other hand the MMS service developed by Nokia is not a success though it is result of innovative requirement.

2.2 Handling Different Levels of Abstraction (C_2):

In market driven requirement engineering, requirements are often arrived in various forms and shapes to the organizations. As requirements are generated from multiple sources like internal and external, understanding of these requirements becomes a challenge to the organization [1].Generally requirements are gathered from a competitor analysis, market surveys, customers, partners, engineers and from management. Finding indirect requirements from the bundle of the requirements collected is a tough job to the team members in the organization. Levels of Misunderstandings increases as the elicitation of requirements from sources vary. Mere chance of obtaining high level requirements which are suitable for the representation of product feature and requirements with very less description is also present. For example, a requirement which is of high level is useful to market personnel but may not be useful to a developer which in turn provides wrong estimates [12].Requirements gathered are compared according to their abstraction level. Each requirement is segregated into different abstraction level based on its importance and description. Requirements which are of same level can be compared to each other and remaining are omitted out. Organisations have mechanisms to handle requirements and place them in various abstraction levels [12].”Requirements on different levels of abstraction and at varying levels of refinement are considered by some as crucial input to the development in order to get a better understanding of what should be developed and why”[1].

2.2.1 Circumstances in which C_2 occurs:

In MDRE the requirements are gathered from different sources and these requirements may contain little or no information about them which becomes a difficult task for the project manager to understand and implement the requirement. According to [27] the requirements can be classified into direct and indirect requirements where indirect requirements contain the information collected through the marketing surveys by the marketing department and idea-like requirements where as direct requirements can be everything from existing and/or potential customers to updates for added functionality proposed by engineers working on the product.

These direct and Indirect requirements consists requirements at different levels of abstraction which will be very difficult to study and understand the requirements (such as to which release the requirement should be added, how it should be compared with other requirements etc.,). [27]

2.2.2 Implications of C_2:

In MDRE the requirements are collected from various sources and have continuous inflow of requirements. These requirements that are gathered contain the information or data of different abstraction levels where some of the requirements are not specified in detail. Requirements must be selected by comparing them with organizational strategy, product strategy and requirements estimates (where each requirement is compared whether the requirement meets the organizational strategy or product strategy, if the requirement meets the criteria then the requirement is taken or else it is rejected) .The requirements at different abstraction levels cannot be compared with other requirements because the information or data regarding the requirements is abstract. Without comparing the requirements prioritization becomes difficult.
This again affects the release of the product; if requirement is not prioritized the requirement cannot be matched to the release of the product (Which feature of the product should be included to nth release of the product).
If the requirements of different abstraction levels are included then it affects the SDLC and the organizations have to face challenges such as requirements prioritization and release planning which is a heavy task to solve these challenges.

2.3 Writing Understandable Requirements (C_3):

Writing Understandable Requirements is one of the major challenging issues when market driven requirements engineering is concerned. In market driven requirements engineering, the product is developed by focusing on different kinds of markets or a niche market [10]. In this case the product must be developed by keeping the market in mind. In market driven requirements engineering the contact with the end users are limited when compared to bespoke development where the software is developed for a single customer. In bespoke the customer is well known and the requirements are gathered and elicited with the help of negotiation between the developer and the customer but when the market driven development is considered this principle cannot be applied since the customer is not well defined. In the market driven requirements engineering product is developed for different kinds of customers like for example novice, experts, users etc, where requirements tend to change frequently for every release of the product into the market. So requirements writings or documentation has become a major challenge in the field of market driven product development [10]. These requirements are sometimes collected indirectly within the organization for example the marketing department can provide most valuable requirements according to the market needs, also at the same time internally the organization can also come out with new requirements for the market. As a result of writing understandable requirements the Key customers can be identified. Key customers are very important in market driven development since these are the important people who will only cover the large market. When the key customer is identified the product can be developed by keeping complete effort with respect to them [10].

Moreover, Market-driven projects tend to have vaguely stated requirements and an informal mode of expression and delivery, whereas customers for customer-specific projects may express their requirements much more fully and formally [10].

Market-driven projects have to deal with requirements that are expressed in documents that are less authoritative and monolithic. Development staff, usually from a separate marketing group, may obtain sketchy statements of requirements from users of existing or competing systems, but they cannot obtain definitive requirements from outside their organization.

Because there are no identifiable customers, Requirements misunderstandings occur more often in market-driven projects [10].

For a project to be successful, requirements need to be accurate and well documented. Complete and accurate requirements are the foundation for any Successful project developed using various systems using a formal method. A requirement is said to be understandable if it is [6][7][5]

- Traceable
- Which has quality
- Which can be captured easily

2.3.1 Circumstances in which C_3 occurs:

Defects can be introduced in all phases of software development, from requirements to software maintenance. It is desirable to intercept defects as early as possible in the development process, well before traditional software testing begins. The reliance on informal statements of requirements leaves the software development process vulnerable to a large class of potential defects caused by ambiguity and incompleteness in requirements [1].

NASA every year spends millions on designing and building spacecrafts for various missions and dependence on software has increased tenfold, and with the increasing dependence on software comes the risk that bugs can lead to the loss of a mission [1].In the current development process of robotic space Missions at NASA’s Jet Propulsion Laboratory (JPL), usually the set of requirements for a project are developed and maintained informally in the form of Word documents. The result of this lack of formality is a set of requirements that can have significant gaps and ambiguities [1]. Furthermore, the relation between requirements and testing becomes informal.

It becomes hard to understand project requirements if inadequate communication and tacit assent to a customer demand occur [6]. Requirements become unclear when product managers are unable to convey the desired meaning of what they exactly require and when requirements are fragmented for solution design [6].Understandable requirements are not evolved when they are[5][6][7]

- Fragmented
- Incomplete
- Un modifiable
- Ambiguous
- Inconsistent
- Not traceable
- Improperly Documented
- Unstructured
- Written by less experienced practitioners
- Lack of Communication
- Less domain knowledge
- Less Descriptive.

2.3.2 Implications of C_3:

Writing understandable requirements becomes mandatory as it is the first step in the Software requirement specification document. If requirements are not written well then effect will be on whole system which results in the following:

- System may deliver the product late, as confusion may prevail while understanding requirements [5].
- Contrary to expected, developing software may cost more because of redefining requirements again when needed [5].
- System may become unreliable with unclear requirements [5].
- If requirements are vaguely described it directly affects the service and design of the product.
- Disrupting normal operation, the regular system may cause serious errors and crashes which contain faulty requirements.
- Maintaining costs are charged which is a mere waste for vaguely written requirements [5].
- Allowances of rigorous analysis of the system, specification and implementation properties are not done [7].

3. SOLUTIONS TO THE CHALLENGES IDENTIFIED (S - C)

3.1 Solutions addressing Requirements Volatility (S – C_1):

There are different ways through which organizations can overcome the requirements volatility challenge. Many researchers have found some solutions to stabilise the requirements. The requirements are of both types which are very rapid and also creep.[17] These types of requirements can be stabilised by using some techniques which helps the organization to overcome the challenge of the frequent changes in the requirements.

According to the studies from [16, 18, 19] it is known that these change of requirements can be managed using some analysis and techniques which the authors have proposed in their research. The first basic step for effectively managing the changes in the requirements is to classify and characterise the nature of the requirements changes following with the requirements prioritization technique. The unnecessary requirement can be avoided in the analysis phase of the project development.

3.1.1 Rejection of requirements (S_1 - C_1):

As in MDRE the requirements are gathered from many sources which contain useful and useless requirements, if the requirements that don’t contribute to the project are chosen then it affects the cost, time and overall performance of the project. By using the prioritization technique the number of requirements that are entering into the project can be reduced based on the intensity and impact on the overall project performance.[17]

In the research study done by the authors in [19], they have found that there are different types of change requests which have inadequate information about these change requests. In order to overcome the limited and in sufficient data about the change requests the authors have used the card sorting technique which is established method for knowledge elicitation. This card sorting technique was implemented with the help of software practitioners in order to draw the results on the way the software practitioners categorize the change requests and to gather more information about the kind of classification they produces.[19][16]

3.1.2 Prototyping (S_2 – C_1):

As most of the organizations produce new technological and functional product the success rate of the product cannot be identified until and unless the product is viewed by the customers or users. In order to know the expectations and requirements of the user regarding the functionalities and the features the prototyping model or pilot experiment is very much useful. In this experiment the prototype of the product is designed and projected on screens or through any advertisements where the features of the products are roughly documented in order to check whether the product meets the customer requirements. If it meets the customer requirements then the actual product development will be started[17].The best example for this, now a day we have heard about the product called pomegranate which has wide variety of sophisticated features which none of the product in the market possess. Though the product is not real, the features of the products are shown with which a feedback is collected on that product and if the circumstances permits the product may be developed. This is one way of stabilizing the requirements, though there will be an inflow of requirements, the upcoming requirements can be faced by the organization as the product is highly sophisticated according to its features.

3.1.3 Built and Fix Model (S_3 – C_1):

In this model the first version of the software is built, as the new requirements occur during the project the developed project is modified according to the new requirements and this is repeated until there is stability in the requirements or the customer is satisfied.[25].

3.1.4 Measuring the rate of Requirements change (S_3 – C_1):

By measuring the requirements change there is a chance of exploring the impact and cost of these changing requirements [17]. The requirements can be measured using the function points. The function point is a synthetic metric derived from five external attributes of software systems: inputs, outputs, inquiries, logistics, files and interface.[17] by using these function points the size of the project is estimated and the changes in the project can be seen with which the unnecessary requirements can be avoided into the project development.

Not only these but there are some other methods and tools which can be used to facilitate the speed with which changes can be processed [17]. These methods are listed below.

1. JAD(joint application Design)
2. RAD(Rapid Application Development)
3. Requirements Inspection
4. QFD (quality function deployment.

Though these methods do not reduce the rapid changes of requirements but helps to increase the speed of filtering the requirements.[17][16]

3.2 Solution addressing Handling Different Levels of Abstraction (S – C_2):

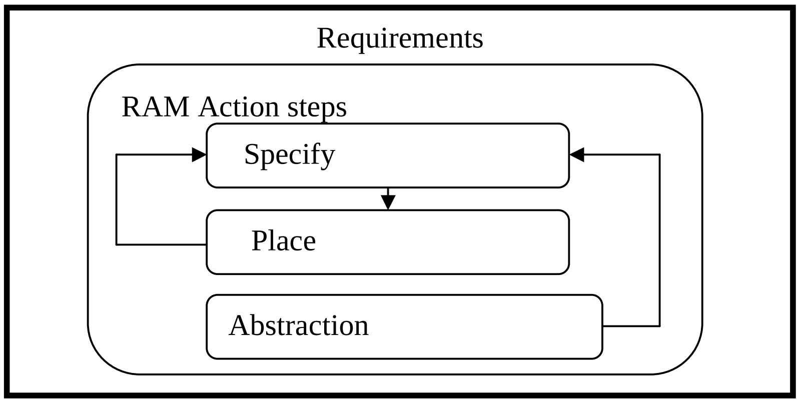
Requirements Abstraction Model (RAM):

Evolution of RAM:

Requirements should be specified on various abstraction levels depending upon their usage but not on the type of requirement [28].Due to the increased pressure on the product developed organizations which should handle the bundle of requirements which are of various abstraction levels, development of RAM was proposed. Central motivation behind the development of RAM is the way how the incoming requirements were handled which are of various abstraction levels and giving a model to the product management. Requirements abstraction model was developed in various stages. Initially RAM went through major validations and was deemed enough to test in an industry. Final RAM was evolved as a result of several validations.

RAM Structure and Process:

Basic model is explained below in which changes can be tailored by the organizations specific to the product and their company [27].Requirements engineering using this model involves three basic steps namely Specify, Place and Work up.



Step 1:

1. **Specify:**

In this step elicitation of raw requirements and enough information about it is gathered. Overview of the requirement to understand it by the product manager is done in this step in which attributes of the requirements are specified. Specification of the requirements mainly depends upon the four attributes namely Description, Reason, Risks and Title.
Description about the requirement should be no more than the 5 sentences and should have an internal description which contains the central essence of the requirement. Requirements should be in broad strokes which explain the requirement in a clear way.

2. **Reason/Benefit/Rationale:**

In this attribute benefit of the specified requirement is to be mentioned and reason behind the requirement is to be specified .Benefit in the context of the user/ product should be mentioned according to them.

3. **Restriction/Risks:**

Description is attached to the requirements in which risks which are associated with it are described. This attribute constitutes the negotiation space in the requirement. Restrictions and risks of the requirements are mentioned prior so that they are helpful when requirement selection is done.

4. Title:

This should reflect the contents of the requirement in an abstract way. Title is not permitted to be in a detailed way. Description of the title should be no more than five words. Other attributes can also be associated with the requirements to understand them in detail. Attributes like State, Version, Date of creation, Dependency, Requirement owner, requirement source, last changed etc can be specified with requirements in order to understand and trace them quickly [27].

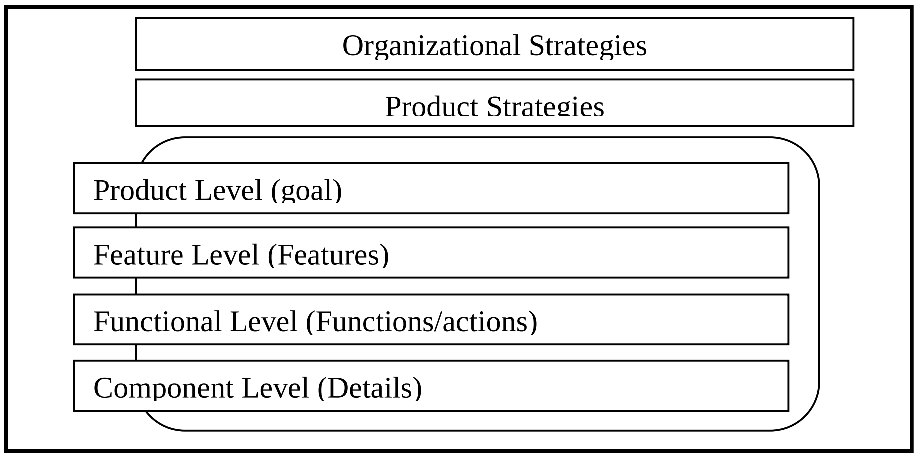
Step 2:

1. Place:

Placing requirements in any of the abstraction levels present in the RAM model is the main aim of the second step. The structure of the RAM model consists of four abstraction levels like Product level, Feature level, Functional level and component level. The overall structure of the RAM model is shown in the figure below with all the levels.

2. Product level:

Requirements in this level are considered to possess a nature which describes the goal of the product. It is considered to be the most abstract level in the whole structure. Requirements placed in this level are very brief and can be compared directly to the product strategies and indirectly to the organizational strategies [2].


Organizational Strategies
Product Strategies
Product Level (goal)
Feature Level (Features)
Functional Level (Functions/actions)
Component Level (Details)

3. Feature Level:

Features of the requirements which product supports are placed in this level. Abstract description of the feature is defined in this level rather than listing the functions that are needed fora product to support a feature [27].

4. Functional level:

This level is considered to be the repository for the functional requirements actions which are possible for a user to do are listed in this level. Non functional requirements area also mentioned in this stage requirements placed here are needed to strive harder to be testable and unambiguous [27].

5. Component level:

Being the last level In Requirement abstraction model, it contains requirements which are described in depth with the information that helps to solve things. Requirements which are elicited through internal sources are mainly placed in this level. This level has a possibility of breaking down functional level requirements in more detail and set limits to a requirement which is in functional level.

Step 3:

Abstraction (Work up):

This is the final step in the RAM model. In this model abstraction of requirements are done according their abstraction level[27].Either work up requirements or breaking the requirements is done depending upon the placement of the original requirement. Work up procedure involves creating new requirements which are also called as work up requirements or creating new requirements. According to many situations and several reasons this process takes place. First reason as mentioned above, as the models goal is to compare every requirement with the product strategies [27].Thus all requirements are needed to be abstracted up to the product level. These procedures are explained in the form of rules 1 and 2 below.

Rule 1: No requirement may exist without having a connection to the product level:

This rule can be succeeded in two ways. One is by creating new work up requirement and other is by linking requirement in the question to the existing requirement which is on level upper adjacent to it. In both the cases the requirement is abstracted upward and compared to the product strategies [27].

Rule2: All requirements have to be broken down to functional level.

For any requirement to be detailed enough it should be broke down enough to act as a design. To place the requirement in a correct abstraction level it needs to be broken down and brought to functional level. Breaking down the requirement from a higher level to lower level is done in this step. This is done to satisfy the original requirement to the extent of giving a good foundation for the initiation of realization of the requirement. Creation of several work-up requirements is done by rule2.

3.3 Solution addressing Handling Different Levels of Abstraction (S – C_3):

To understand requirements specified in the SRS, priory they should be well defined, traceable and which can be easily captured. Several methods are discovered till date to capture, trace and to write requirements which has a good quality in it. Value estimation method is described below to know how the requirements are traced and implemented.

Value Estimation Method for feature oriented requirements:

Traceability is helpful to check requirements consistency, to analyze requirements conflicts and for change impact analysis. Organisation faces heavy burden while maintaining these complex systems and traceability links as there would be numerous artefacts internally and externally [2].Method known as value estimation method is proposed for tracking feature oriented requirements which uses one of its feature as an intermediate catalysis which generates traceability links by prioritizing the levels and by analyzing value of an each feature.

Identifying externally visible features of a product and organizing them into a model is the activity done by feature model, which acts as a mediator which generates the requirements traceability links [2].

Feature model consists of 4 steps to distinguish the external visible characteristics:

- Feature Identification
- Domain planning
- Feature organization
- Feature Refinement

In this model from user requirements, feature oriented requirements are derived in the first step. And then they are traced back to group user requirements to features to know the traceability links which connects the artefacts to features. Then each feature is prioritized based on the risk, effort and its value. Based on granularity of artefacts those are prioritized as Low, Medium and High [2]. Links of traceability are refined according to their priority.

Process Overview:

This process comprises of five distinct phases like requirements definition, feature modelling, feature prioritization, requirement linking and traceability links evaluation.

Requirement definition:

In this process, requirements collected have to be normalized in the first step in order to map them to the various types of artefacts. As many requirements may be unclear and abstract these are to be analyzed and then mapped [2]. Using a dictionary which is a glossary of standard terms, atomic requirements are identified. By following functional and non functional approaches requirements which ate atomic are classified to which a unique ID is assigned. Final step of this phase contains the list of atomic requirements with their unique identification [2].

Feature modelling:

In this phase, generation of a feature diagram is done by categorizing identified target system and its features. Differences in a category are found out to list the features with commonality and variability [2]. To design the feature diagram, these features are used and relationships between these features are considered at this phase.

Feature Prioritization:

Determination of importance to the features identified is done this phase. Granularity level increases proportional to the increasing level [2]. First stakeholders assess the requirements based on their risk, effort and its value. Classifications of features are done based on the estimation. In the second step features are classified according to the granularity. Finely grained traceability links and features which are ranked based on priority levels are obtained as results in this phase.

Requirement Linking:

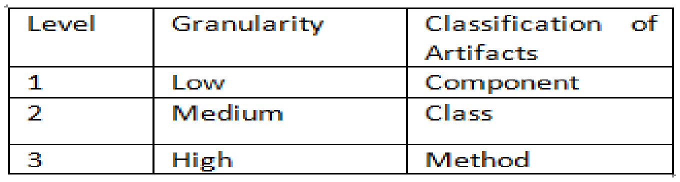
In this phase, artefacts are assigned to its related feature. Usage of features is done to link artefacts and requirements. By granularity level implementation items are fractionated and then traceability links are established [2]. These links corresponds a relationship among the feature, artefacts and requirements. Requirements which are important are traced with links which the final result of this phase.

Traceability links evaluation:

Obtained traceability links are utilized for change impact analysis, consistency checking and requirement conflict analysis. These links can be used repeatedly by frequent refining process [2].

Priority level:

To generate requirements traceability links three kinds of granularity levels are proposed which are shown in table below by which all stake holders assess the features. [2] By this method we can obtain finely grained traceability links.



Level	Granularity	Classification of Artifacts
1	Low	Component
2	Medium	Class
3	High	Method

4. MERITS AND DEMERITS

4.2 Strengths and weaknesses of Rejection of Requirements:

Pros:

- Unnecessary requirements are rejected at the early phases of SDLC.
- Customer Satisfaction for the product can be achieved through selecting the needed requirements.
- Project performance can be improved along with quality in the product.

Cons:

- Best suited for small organizations and for the projects in academia because they carry out small projects where the requirements are scalable to certain extent and when the requirements are heavy this does not produce good resulting while selecting the requirements.

4.2 Strengths and weaknesses of Prototyping:

Pros:

- The use of prototypes reduces the requirement changes for about 10 percent to 25 percent [17].
- Efficient way of stabilizing requirements.
- Requirements can be refined according the customer needs.

Cons:

- If this is carried out there may be a situation where the company has to face requirements overload for the project
- Implementing this is quite expensive and will increase the project’s cost.
- Not every customer requirements is satisfied but the focus is made on a group of customers which fulfils their requirement.

4.3 Strengths and weaknesses of Built and Fix model:

Pros:

- No waiting time to implement new requirements.
- It works effectively when implemented for small projects where there is no requirements overload.
- Project is carried out iteratively and incrementally and includes the release of the product.

Cons:

- It is not suitable for large projects.
- Modifying the Software many times is very expensive and increases the cost of the project.
- The number of Releases of the product increases with which company has to face the release planning challenges and it may also affect the schedule of the product development.

4.4 Strengths and weaknesses of Requirements Abstraction Model (RAM):

Pros

- As all the requirements are compared to the product strategies, violation of the goals set by the organization is not done. It provides an assurance to see that requirements don't violate the goals set by the management. It also makes the resources free as it has the possibility to dismiss the requirements as early as possible [27].
- Better decision support can be achieved by the professionals from the management as the requirements can be followed through several levels of abstraction which gives a better chance to understand each requirement.
- As each and every requirement is broke down to an abstraction level there is a mere chance to test each and every requirement. These requirements are widely used for initiating the project and can easily realize them.
- By work up strategy requirements are compared to various levels of abstraction which is also a prerequisite to effective prioritization and release planning [27].

Cons

- "Risk in this model is that product strategies have to be present and explicit in order for this to work" [27].

In near future RAM v.2.0 may evolve which is usable and a light weight model that covers requirements engineering activities which are performed during product planning and management and also in post projects which addresses most of the issues identified in the industry. New model may incorporate features like explicit support for requirements estimation, risk analysis, prioritization and packaging of requirements into development instances [27].

5. VALIDATION & ANALYSIS

From different research studies it is known that requirements volatility is a major problem associated in the project development lifecycle.[20][17][16] Different types of customers have different needs organization have to be careful while collecting the requirements such that the requirements gathered should be documented without any missing requirement and without ambiguity. If any of the requirements are not documented or contains ambiguity the requirements must be changed during the development process which is not a cost effective process and time consuming. Though organizations focus on the requirements, it is a challenging task to satisfy the customers with the requirements, as the requirements from the customers are inevitable. Through this study we have come across some solutions in order to stabilize the requirements which help the organization to face the requirement changes and trace the correct set of requirements in order to develop a quality product and satisfy its customers. Also, through this study we have identified RAM model as a solution to mitigate the changes in the levels of abstraction in the requirements. This model is adaptable and example driven in nature which can be adapted by any product or organization. It was designed keeping developers and product managers in mind. This model helped in raising number of issues early instead of leaving uncertainties. This model acts also like a risk mitigation tool. After the validation results show us that mainly problems were raised due to infrastructure that is required to support RAM but not directly with Requirements Abstraction model.RAM can be modified over time which reflects the current situation of a development organization and product life cycle [27]. Also, to handle the problem of proper documentation of requirements, in various areas researches has been carried out related to requirements traceability. Techniques like constraint networks, hyper text, Integration documents, requirements traceability matrices are provided in many studies which only track the requirements but value of the requirements are not considered. In this model value as well as cost benefit factor is also discussed which lacks in other techniques. Model like Value based requirements tracing which also supports in tailoring requirements and tracking through conducting grouping requirements using hierarchy is proposed but lacks cost estimation parameter concept which is obtained and evaluated in this study [3]. Another study by Riebischi proposed feature models which enables to lessen the efforts to maintain the requirements traceability but it lacks to generate valuable traceability links as it just traces the relation between problem domain and solution domain [4]. Effort and cost for precise traces are more expensive than the less precise traces. Cost and effort can be reduced if unimportant requirements are not taken into account or omitted. In our view this method may be advantageous as this Meta model uses features which are the major characteristics of the product, which is a vital concept in the software product line. By using feature it not only acts as an intermediate to solve the gap between the requirements and artefacts but also helps to manage requirement variability. It also structures the links traced out between requirements and elements of solutions like source code, test cases and objects. Understanding requirements does not only mean to trace them but also defining them in a clear state in order to become easy for the practitioners to use them. Many other models like model based testing process to determine traceability, win-win spiral model, Requirement Ontology to improve quality in the requirements are proposed to acquire a well defined requirement which is useful for a software project in a long run[8][9].

6. CONCLUSION:

In this report three major problems in MDRE were discussed. Though many tools, methods and models were developed to solve the issues and fix them, MDRE environment makes them difficult to implement and successfully retrieve the solutions. To our analysis these methods still need to improve in many ways to refine the requirements and its changes. By a closer view upon the challenges we come to know that challenges in MDRE are closely inter related, therefore solutions should be chosen in a corrective manner so that it addresses the issue corresponding to the related challenge. Hence, this report would certainly help in understanding the major challenges faced in industry and solutions which could mitigate these challenges.

REFERENCES

[1] M.H. Smith and K. Havelund, “Requirements capture with RCAT,” International Requirements Engineering, 2008. RE'08. 16th IEEE, 2008, pp 183–192.

[2] S. Ahn and K. Chong, “A value estimation method for feature-oriented requirements tracing,” in Convergence and Hybrid Information Technology, 2008, ICHIT'08, International Conference on, pp. 159–162, 2008

[3] M. Heindl and S. Biffl, “A case study on value-based requirements tracing,” Proceedings of the 10th European software engineering conference held jointly with 13th ACM SIGSOFT international symposium on Foundations of software engineering, 2005, pp. 60–69.

[4] M.Riebisch, “Supporting evolutionary development by feature models and traceability links,” 2004

[5] Ian Sommerville,”Requirements Engineering an Overview”,Lancaster University, UK

[6] S. Fricker, T. Gorschek, C. Byman, and A. Schmidle, “Handshaking with Implementation Proposals: Negotiating Requirements Understanding,” IEEE software, vol. 27, 2010, pp. 72–80.

[7] Hsia, P, Davis A.M, Kung D. C, "Status report: requirements engineering," Software, IEEE, vol.10, no.6, pp.75-79, Nov1993, doi:10.1109/52.241974

[8] D.V. Dzung and A. Ohnishi, “Improvement of Quality of Software Requirements with Requirements Ontology,” Quality Software, QSIC’09 9th International Conference on, 2009, pp. 284–289

[9] B. Boehm, H. In, “Identifying quality-requirement conflicts,” Software, IEEE, vol. 13, 2002, pp. 25–35

[10] A. Gomes and A. Pettersson, “Market-Driven Requirements Engineering Process Model–MDREPM,” Karlskrona: Blekinge Institute of Technology, 2007.

[11] Karlsson, \.G. Dahlstedt, B. Regnell, J.Nattoch Dag, and A. Persson, “Requirements engineering challenges in market-driven software development-An interview study withpractitioners,” Information and Software technology, vol. 49, 2007, pp. 588–604.

[12] E.F. Ecklund Jr, L.M. Delcambre, and M.J Freiling, “Change cases: use cases that identify future requirements,” Proceedings of the 11th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, 1996, pp.342–358.

[13] M. Khurum, K. Aslam, and T. Gorschek, “MERTS – A method for early requirements triage and selection utilizing product strategies”, APSEC 2007

[14] E. Carmel and S. Becker, “A process model for packaged software development,” IEEE Transactions on Engineering Management, vol. 42, Feb. 1995, pp. 50-61.

[15] Davis, A.M, Nurmuliani, N, Sooyong Park, Zowghi D, "Requirements Change: What's the Alternative?", Computer Software and Applications, 2008, COMPSAC '08, 32nd Annual IEEE International , vol., no., pp.635-638, July 28-2008-Aug.1-2008, doi: 10.1109/COMPSAC.2008.216 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4591637&isnumber=4591503

[16] McGee S, Greer D, "A Software Requirements Change Source Taxonomy", Software Engineering Advances, 2009, ICSEA '09, Fourth International Conference on , vol., no., pp.51-58, 20-25 Sept. 2009 http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5298458&isnumber=5298193

[17] Jones, C, "Strategies for managing requirements creep," J. Computer, vol.29, no.6, pp.92-94, Jun 1996

[18] Nurmuliani, N, Zowghi, D. Powell, S, "Analysis of requirements volatility during software development life cycle," Software Engineering Conference, 2004, Proceedings 2004 Australian , vol.no.,pp.28-37,2004

[19] Nurmuliani, N.; Zowghi, D, Williams, S.P, "Using card sorting technique to classify requirements change," Requirements Engineering Conference, 2004. Proceedings.,12th IEEE International , vol., no., pp. 240- 248, 6-11 Sept. 2004.

[20] Thakurta R, Ahlemann F. Understanding Requirements Volatility in Software Projects – An Empirical Investigation of Volatility Awareness, Management Approaches and their Applicability Methodology, 2010:1-10.

[21]L. Karlsson, Å. Dahlstedt, G., J. N. o. Dag, B.Regnell, and A. Persson, "Challenges in Market-Driven Requirements Engineering - an Industrial Interview Study," in Eighth International Workshop on requirements Engineering, Germany, 2002,pp.101-112

[22]Andrigo Gomes, Andreas Pettersson, “Market-Driven Requirements Engineering Process Model – MDREPM”, MSE-2007-06, pp.141, 2007

[23]Jonas Bergström, Andreas Dahlqvist, “BESMART - a framework for shifting from BESpoke to MARkeT-driven requirements engineering”, MSE2007-24, pp. 119, 2007

[24]Karlsson L, Dahlstedt ÅG, Natt J, Regnell B, Persson A, “Challenges in Market-Driven Requirements Engineering - an Industrial Interview Study”, Requirements Engineering, 2002

[25]Sudhakar M, “Managing the Impact of Requirements Volatility Master Thesis”, 2005, Changes, 2005, 1-42. https://goldpractice.thedacs.com/practices/mr/index.php

[27]Gorschek T, Wohlin C. “Requirements Abstraction Model”, Requirements Engineering. 2005; 11(1):79-101. http://www.springerlink.com/index/10.1007/s00766-005-0020-7.

[28]Gorschek T, “Requirements Engineering supporting Technical Product Management”, Blekinge Institute of Technology Doctoral Dissertation Series, No 2006:01, 2006, ISSN 1653-2090, ISBN 91-7295-081-1 http://gorschek.com/doc/c_v_files/tgo_thesis_final.pdf

[29]Sairam Vakkalanka, Ranjith Engu, “Requirements Triage – Challenges and Solutions”, International Journal of Software Engineering & Applications (IJSEA), 3(2), 41 – 58, 2012

Authors

Ranjith Engu, completed Masters in Software Engineering at Blekinge Tekniska Högskola, Karlskrona, Sweden in December 2011.

Sairam Vakkalanka. pursuing masters in software Engineering at Blekinge Tekniska Högskola,Karlskrona, Sweden.