***Large Scale Requirement Engineering***

***Reflective Report***

Zhipeng Dong

dzp0713@gmail.com

BTH University

Sweden, Karlskrona

9208111295

**1. Article Selection**

**1.1 First Article**

**Article information:**

Svensson R B, Regnell B. A case study evaluation of the guideline-supported QUPER model for elicitation of quality requirements[C]//International Working Conference on Requirements Engineering: Foundation for Software Quality. Springer International Publishing, 2015: 230-246.

**Article name:**

A case study evaluation of the guideline-supported QUPER model for elicitation of quality requirements

**Motivation for selection:**

The reason we choose this article can be summarized as follows:

1. The published year of the article is 2015, Its publish time is very close to us.

2. Through our previous study of software requirements, we have found there are less quality requirements in the requirements database. The description of these quality requirements is very ambiguous, it is very hard to understand them. But the software quality requirements are very important for us.

3. This article is a good article to help us understand quality requirements and implement quality requirements.

4. This article presents a detail guideline of how to use QUPER model, this model is a good method to help manager to judge the quality requirements from different concept. The different concept includes: reference levels, quality breakpoints, cost barrier and so on.

5. The aim of QUPER model is to help market-driven software products to plan their product release more effectively, and it will also help developers deliver high quality software product to the market as soon as possible.

6. The QUPER model is the only way to solve quality requirements quality and cost constraints, and it is easy to learn and use.

7. The fifth section of this article is: QUPER Case Study Evaluation. I found there are lots of evaluation results lack of necessary relevant data support, its accuracy is suspected. So, I want to use the QUPER model in practice.

**Implementation Plan:**

From the release planning assignment, I found there are many quality requirements in the requirements document. And most of them are abstract and difficult to understand. So, I will use the guideline-supported QUPER model with some quality requirements in the requirements document of the online educational platform system.

The implementation plan is divided into the following steps:

Step1: Select several quality requirements from our requirement document, record their name and briefly description.

Step2: Identify candidate quality requirements with considered relevant features. We need to identified the features for each selected candidate quality requirements.

Step3: Define a scale and a measurement unit for each quality requirement which I have selected.

Step4: Identify reference levels for each quality requirement based on our own products or some similar products from competitors.

Step5: Elicit quality breakpoints. There are three different breakpoints, they are utility breakpoint, saturation breakpoint and differentiation breakpoint. We elicit the quality breakpoints according to the determined market expectations.

Step6: Estimate the cost in terms of the cost barriers, we plan to use two cost barriers for each quality requirement.

Step7: Run these quality requirements on the QUPER model and summary the result.

Before using QUPER model we need to deep understand our project. Our product is an online educational platform system. Its main user groups are students and teachers. The target market is Sweden university.

**Execution:**

I will follow the implementation plan to performed QUPER.

Step1: Select several quality requirements from our requirement document, record their name and briefly description.

The quality requirements I have selected are:

|  |  |  |
| --- | --- | --- |
| Requirement Number | Name | Briefly description |
| No.7 | Heterogeneous System Access | The product shall be accessible from common computer-and operating system platforms. |
| No.13 | Quick and Easy Overview of Relevant Information | It shall be easy to find relevant information. |
| No.128 | File compatibility | As a teacher assistant, he/she can choose which file types are acceptable for submit by the student. |

Step2: Identify candidate quality requirements with considered relevant features.

|  |  |  |
| --- | --- | --- |
| Requirement Number | Name | Relevant features |
| No.7 | Heterogeneous System Access | system should be running in different systems, such as Linux, Windows. |
| No.13 | Quick and Easy Overview of Relevant Information | Basic information operation of the system, such as courses info, discussions info. |
| No.128 | File compatibility | Upload different kinds of files, such as pdf, docx. |

Step3: Define a scale and a measurement unit for each quality requirement which I have selected.

|  |  |  |
| --- | --- | --- |
| Requirement Number | Scale | Measurement unit |
| No.7 | Support system number | One system, two systems, three systems… |
| No.13 | Time | Minutes |
| No.128 | Support file types number | One type of file, two types of files… |

Step4: Identify reference levels for each quality requirement.

|  |  |
| --- | --- |
| Requirement Number | Reference levels |
| No.7 | Level 1: support 2 different kinds of systems (windows and Linux)  Level 2: support 3 different kinds of systems (windows && Linux && IOS)  Level 3: support 4 different kinds of systems (windows && Linux && UNIX && IOS) |
| No.13 | Level 1: 100 minutes  Level 2: 80 minutes  Level 3: 60 minutes |
| No.128 | Level 1: support 2 different kinds of files.  Level 2: support 4 different kinds of files.  Level 3: support 6 different kinds of files. |

Step5: Elicit quality breakpoints. There are three different breakpoints, they are utility breakpoint, saturation breakpoint and differentiation breakpoint.

|  |  |
| --- | --- |
| Requirement Number | Quality breakpoints |
| No.7 | UTILITY: Windows, Linux (According to the reference levels and market)  SATURATION: windows && Linux && UNIX && IOS  DIFFERENTIATION: ALL |
| No.13 | UTILITY: 100 minutes.  SATURATION: 80 minutes.  DIFFERENTIATION: 60 minutes. |
| No.128 | UTILITY: pdf, doc/docx, txt (According to the reference levels and market)  SATURATION: pdf, html/htm, doc/docx, sxw/odt, rtf, txt  DIFFERENTIATION: pdf, html/htm, doc/docx, sxw/odt, rtf, txt |

Step6: Estimate the cost in terms of the cost barriers.

|  |  |
| --- | --- |
| Requirement Number | Cost barriers |
| No.7 | OUR PRODUCT: WINDOWS, IOS, UNIX, LINUX  RATIONALE: Accept other minority systems |
| No.13 | OUR PRODUCT: 60 minutes  RATIONALE: New features come in |
| No.128 | OUR PRODUCT: support pdf, doc/docx, txt  RATIONALE: Accept other minority kinds of files. |

Step7: Run these quality requirements on the QUPER model and summary the result.

Result:

All the abstract quality requirements become more readable and can be easy to understand. So that, we can gain the market competitiveness from the quality of our product.

**Lessons Learned:**

1. Quality requirements is a very important part of requirements engineering domain. Because meet the market requirements is an important guarantee for the success of the product. If the product has a high-quality level, its market competitiveness will become more and more larger.

2. In fact, lots of quality requirements are abstract and unable to understand. So, in order to understand the abstract description of quality requirements, we need to use some quantized techniques, QUPER model is a very effective method.

3. By executing QUPER model in our own product, I known how to develop quality requirements and how to compare the quality requirements to other competitors.

4. QUPER model not only can help company or manager to understand the quality requirement more, but also can help the company or manager to gain the market competitiveness from quality of a product, which is also increase the success rate and reduce the risk of project.

5. By executing QUPER model in our own product, I known the importance of cost barriers, reference levels and quality breakpoints in QUPER model. And I know that thinking market competitiveness is an important process when we using QUPER model.

**Reflection:**

In order to implement the QUPER model, I decided to choose some other articles to learn. I selected two articles, they are “Introducing support for release planning of quality requirements - an industrial evaluation of the QUPER model” [3] and “Setting quality targets for coming releases with QUPER: an industrial case study” [4]. These two articles are related to the QUPER model, one is an industrial evaluation, another is an industrial case study.

By learning these two articles, I know the cost dependence in the implementing process of QUPER is very important. It will help us to better understand the cost dependence between quality requirements in my implementation process.

Another important aspect, when we consider the software quality requirements, we need to contact the related market to consider them. When we set the quality targets, we not only need to consider the related market requirements, but also need to consider the competitive products.

We have found abstraction is a huge problem when we analyzing the quality requirements. From these two articles, we know that, we could describe the quality requirements by answering three questions. These three questions are: "When is the quality level good enough?", "When is the quality level a competitive?", "How to judge the quality level?", and QUPER model is a good model to help us answer these questions.

For QUPER, a very important thing is that breakpoints change over time. In order to achieve your goals and maximize benefits, the three views are a good indication.

**1.2 Second article**

**Article information:**

Gorschek T, Wohlin C. Requirements abstraction model[J]. Requirements Engineering, 2006, 11(1): 79-101.

**Article name:**

Requirements abstraction model

**Motivation for selection:**

The reason we choose this article can be summarized as follows:

1. With the development of the software industry, large scale requirement engineering has become more and more popular. How to make requirement management team quickly and accurately understand the requirements becomes more and more important. The RAM will help the implementing team to easily understand the requirements without any chaos.
2. The RAM helps the requirement team to easily manage, prioritize, roadmap and release planning process.
3. RAM will allow all the requirements to be compared to the product strategy to ensure compliance with management set goals.
4. The RAM placed all the requirements on different levels based on the given guidelines. So, each requirement will have a hierarchy from goal to component levels description.
5. The RAM provides a detailed analysis of requirements, it will reduce the risk of project failure.

In order to easily make requirements in an abstract model and breakdown into different levels, I selected this article. I want to see how the RAM works.

**Implementation Plan:**

RAM is carried out in three steps.

They are:

1. Specify: This is the elicitation phase, in this phase we will get an overview of the requirements. And all the requirements are gathered from the stakeholders. There are four attributes to describe the requirements. The four attributes are specified in this step. They are:
   1. Description: Describes the central essence of the requirements.
   2. Reason: Explain why the requirement is specified.
   3. Risk/ Restrictions: Describes the restrictions and risks of the requirements.
   4. Title: The name of the requirement, the requirement should be able to clearly description and the essence of the requirement should be reflected.
2. Place: In this phase, I will analysis how to placing the requirements at the right abstraction level. There are four abstraction levels exist. They are:
   1. Product Level: This level is goal – like in nature.
   2. Feature Level: This level is the features that the product supports.
   3. Function Level: This level is functional requirements and non-functional requirements.
   4. Component Level and finally: This level is all requirements have to be broken down to function level/
3. Abstraction: This is the third step, in this step, the requirements are broken down to detailed requirements. And new requirements will be created in this phase.

We decide to select 5 requirements that are at the most basic level and use this model to place the requirements on each level. Then I will follow the guide to breakdown and abstract the requirements. I want to implement this technique to understand how it works.

**Execution:**

The requirements we selected:

|  |  |  |
| --- | --- | --- |
| Requirement Number | Requirement Name | Requirement Short Description |
| No.145 | User Interface | As an assistant teacher, I would like the system to have user interface that is easy to learn and understand. |
| No.136 | Add, Delete, Update function | This system should have added function, delete function and update function. |
| No.75 | Sign Up for Participants | Through this, the user can register an account in the system. |
| No.27 | Login/Logout | The user must login before able to access the product. Unauthorized users should not have access to product functionality. |
| No.19 | User Interface Language | The user interface language is Swedish or English. Will be extended with other languages. |

Execution is done in three steps wherein the steps are described clearly below:

Step1: Specify: Specify the requirements that have been listed below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Requirement Number | Requirement Name | Description | Reason | Risk/ Restrictions | Title |
| No.145 | User Interface | As an assistant teacher, I would like the system to have user interface that is easy to learn and understand. | A good user interface is an important guarantee of customer satisfaction. | If the user interface design is unreasonable, the users’ satisfaction will be reduced. | User Interface requirement. |
| No.136 | Add, Delete, Update function | This system should have added function, delete function and update function. | In order to ensure the accuracy of the database info, as well as the stability of the database size, the database could be added, delete, update. | May add useless system info, and some important info may be deleted. | Add, Delete, Update function. |
| No.75 | Sign Up for Participants | Through this, the user can register an account in the system. | In order to distinguish different users, each user should have a specific account. | The size of the databases limits the number of registered users. | Sign Up for Participants function. |
| No.27 | Login/Logout | The user must login before able to access the product. Unauthorized users should not have access to product functionality. | In order to ensure the security of this system, the user must first register to use. | When the number of registered users reaches a certain number, the user name duplication problem will occur. | User Login/Logout function. |
| No.19 | User Interface Language | The user interface language is Swedish or English. Will be extended with other languages. | The user interface language is not just support one language, will meet the requirements of different language lovers. | When the various languages are converted, the system response time may slow down. | User Interface Language function. |

Step2: Place. When the requirements are specified we need to place the requirement at the right

level of abstraction.

There are four abstraction levels exist. They are:

Product Level: This level is goal – like in nature.

Feature Level: This level is the features that the product supports.

Function Level: This level is functional requirements and non-functional requirements.

Component Level and finally: This level is all requirements have to be broken down to function level.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Requirement Number | Requirement Name | Product Level | Feature Level | Function Level | Component Level |
| No.145 | User Interface |  | √ |  |  |
| No.136 | Add, Delete, Update function |  |  | √ |  |
| No.75 | Sign Up for Participants |  | √ |  |  |
| No.27 | Login/Logout |  |  | √ |  |
| No.19 | User Interface Language |  |  |  | √ |

Step3: Abstraction: In this step, the requirements are broken down to detailed requirements.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Requirement Number | Requirement Name | Product Level | Feature Level | Function Level | Component Level |
| No.145 | User Interface | The user interface need to be easy understand, this is a created requirement. | Own user interface. | User need to use username and password to access to their interface, this is a new requirement. | The system should have user interface, this is an original requirement. |
| No.136 | Add, Delete, Update function | This is the security of the system. | Manager could add, delete and update info. | All kinds of info can be adding, delete and update. | The system should support add, delete and update operation. |
| No.75 | Sign Up for Participants | The system need to have a special account to access. | Each account includes a username and a password. | The account must be unique, this is the security of the system. | The account information corresponds to the database. |
| No.27 | Login/Logout | This is the security of the system. | The correct username and password are the prerequisites for logging in. | The username and password must be unique. | The account information corresponds to the database. |
| No.19 | User Interface Language | This is the  usability  of the system. | This system supports multiple languages. | This is an original requirement. | Multiple languages can be switched to each other. |

After placing the requirements at the right level, all the requirements are abstracted. we need to detail the requirements enough so that there are corresponding entries for them at each level of abstraction. Each requirement should be attached with other attributes, such as requirements source, owners, dependencies.

**Lessons Learned:**

1. By implementing the RAM model, I felt that requirements abstraction is necessary as it will give a clear idea to implementation team.
2. The first two steps of this process seem to be pretty easy but the last step is the most crucial task. It was difficult for the author only for 5 requirements and we can imagine that it will be a very difficult challenge for companies dealing with large scale or market driven projects where requirements range at the orders of thousands.
3. From this article, I know that under the market driven requirements engineering environment, the requirements flow is continuous and comes from different stakeholders.
4. I think the abstraction of the requirements is a very important process in the requirements analysis phase, and the RAM provides a guide for requirements engineering researchers.
5. RAM provides four levels of abstraction; each level has a very clear description. By implementing this model, I understand the reasons, benefits and possible risks for each requirement.

**Reflection:**

In order to implement the requirements abstraction model, I decided to choose some other articles to learn. I selected two articles, they are “Suitability of the Requirements Abstraction Model (RAM) Requirements for High Level System Testing” [5] and “Industry evaluation of the requirements abstraction model[J]. Requirements engineering” [6]. These two articles are related to the RAM, one is an industrial evaluation, another is an industrial case study.

By learning these two articles, I know that the opportunity to create new requirements is reduced when moving from component level to product level. Through my practice, I agree with it, why the component level requirements are very hard to link the product strategy? The reason is most of component level requirements are come from developers. And component level requirements are very detailed. So, the component level requirements are very hard to link the product strategy.

Through analyzing two companies who are using RAM, we have found using RAM will need more effort and cost, but the product quality will also be improved. So I think the RAM can be used in industry.

Also through my experience, requirements are used to prioritize without having abstraction levels. So, it may be a challenge as some time one requirement looks important and other requirement importance is hidden, so this RAM model helps to place the requirements and work up on it.

**2. Reference**

[1] Svensson R B, Regnell B. A case study evaluation of the guideline-supported QUPER model for elicitation of quality requirements[C]//International Working Conference on Requirements Engineering: Foundation for Software Quality. Springer International Publishing, 2015: 230-246.

[2] Gorschek T, Wohlin C. Requirements abstraction model[J]. Requirements Engineering, 2006, 11(1): 79-101.

[3] Svensson R B, Olsson T, Regnell B. Introducing support for release planning of quality requirements—an industrial evaluation of the QUPER model[C]//Software Product Management, 2008. IWSPM'08. Second International Workshop on. IEEE, 2008: 18-26.

[4] Svensson R B, Sprockel Y, Regnell B, et al. Setting quality targets for coming releases with QUPER: an industrial case study[J]. Requirements Engineering, 2012, 17(4): 283-298.

[5] Muhammad N. Suitability of the Requirements Abstraction Model (RAM) Requirements for High Level System Testing[D]. Blekinge Institute of Technology, 2007.

[6] Gorschek T, Garre P, Larsson S B M, et al. Industry evaluation of the requirements abstraction model[J]. Requirements engineering, 2007, 12(3): 163-190.