Research Paper

2002-09-23

How to handle requirements when developing market driven products?

Jimmy Persson (-)
Gustav Evertsson (-)

Blekinge Institute of Technology, Sweden

Index

Introduction

Overall description

The purpose of this document is to show the result of a research about what happens when there is no actual customer in a software project. We will not give a solution to the problems, but we will try to make you aware of their existence and perhaps point out some directions that could help solving at least some issues.

The research was performed during the course "Requirements Engineering" at Blekinge Institute of Technology in Sweden.

Problem description

In the beginning of this research, we discussed in what way we should confront the problem. We decided that the best way probably would be to create a scenario and use it as an example. Therefore we use the made-up company "Larger Than Life".

Scenario

In the beginning of the year 2000 a company called "Larger Than Life", or LTL, was determined to develop a system for managing the storage, orders and billing of a web shop and/or mail order company. LTL had managed to get an investor who was willing to pay for the development cost. In return, LTL had to come up with a result within a year.

Problem

The basic problem is: "How does one handle requirements if one has no customer?"
This statement consists of at least the following questions:

- How does one select a target group?
- How does one find adequate requirements, when not delivered by a customer?
- How is different stakeholders' point of view prioritised?
- Who has the final word about a requirement when there is a conflict amongst stakeholders?
- How does one verify that a requirement is fulfilled?

There are probably more questions to consider, but in this research these are the ones we have chosen.

Stakeholders

Possible stakeholders are:

- Investors
- Developers
- The target group
    - Storage personnel
    - Managers of economics
    - People that handles orders coming by phone or mail
    - System managers (Handling backups etc.)
    - Customer to the company coming through a web shop

Research

Target group

"How does one select a target group?"

A mayor problem when developing a mass-market-driven product is that you don't have a clear view of neither the customer nor the end user. So you need to identify and consult system stakeholders and collect requirements from multiple viewpoints[1]. It is important that the people feel that you listen to them and that their opinion is important. Because you need them to think right from the beginning so you don't have to make any costly changes later in the project. It is also important to plan for conflicts and conflict resolution. Conflicts are inevitable if a system is to serve many people with different expectations and fears[2].

It can take a lot of time to interview and/or study how different people do a certain work. One way to handle this can be to take a very small group or maybe only one person and let them represent a larger group of stakeholders. The benefit of this method is that the requirement elicitation will go much faster. It is also a big advantage to have a person to call during the development process if something needs to be cleared out. A disadvantage can be that the person does not think like the majority of the users so the product will probably not be perfect in every ways.

Find adequate requirements

"How does one find adequate requirements, when not delivered by a customer?"

Market-driven products usually starts with that someone sees a need for a certain type of product. After the initial market analysis to show that the company can make profit from the product is it time to make the requirement specification. A good start can be to write down a goal for the product, and say that if a certain requirement doesn't fit within this goal it will not be accepted. It can be stored in an idea bank for future versions that may have a modified goal.

When developing a product for a market, rather for a single customer, the pressure on short time-to-market is evident. This is important because market always demands change and it is important to release the product before it is too late. It is important to release your product before competing companies release theirs. It is easier to get new customers and to keep them, than it is to convince the customers of competitors that your product is better than theirs.

One way to handle the requirement elicitation process that UIQ-Technology uses is that they have a special marketing department. The employees from this department visit different trade fairs and conferences to talk to different customers and end users. They will in this way get input that is later used to create high level requirements.[3]

There can be some problems when collection high level requirements. The requirement engineering staff must be able to handle requirements from many different sources. These

[1] Nawrocki Jasiński Walter Wojciechowski, Extreme Programming Modified: Embrace Requirements Engineering Practices, 2002.
[2] Op.cit.
[3] Persson Evertsson Nilsson Ohlsson, Case study, 2002.

sources may include customers, marketing, support, testing, usability evaluation and technology forecasting. The requirements can be of very varying quality. A problem can also be that there are a lot of requirements. Therefore it can be hard to find conflicting requirements or requirements that depend on each other. Even when they are found it can be too late and the affect on the prioritising and release plan can be catastrophic. It is therefore a need to find a way to find the requirement relations early, without spending too much time on in-depth analysis. And you need to find them even if the requirement is poorly written or misspelled. Because of the fact that most requirement are written in a natural language, an automated lexical analysis can be helpful in this kinds of situation. This is of course not a replacement for the human judgement but it can be an affective tool when searching for duplicates and interdependencies among the requirements[4].

A Mass-market-driven product is usually released not only one time but many times. It is therefore necessary to have a management plan for the products lifecycle. Two ways of handling this has been developed at Ericsson Radio System and Telelogic. They call the models for RDEM[5] (Requirement Driven Evolution Modal) and REPEAT[6] (Requirements Engineering Process At Telelogic) respectively. The two models have been developed in parallel but completely independently. They are quite similar in many respects. Both RDEM and REPEAT have lifecycle models with different states for each individual requirement in its progress towards release. Both also have a string focus on roles and responsibilities. The general differences behind the two models are that Telelogic REPEAT is more focused on time-to-market, whereas Ericsson's RDEM is more focused on quality. The lifecycle approach makes the requirement management more flexible so new requirements can be introduced when needed and requirements can change based on changes in the market situation. A major problem with this type of models can be the release plan, but also to decide what to include in the current release. Many requirements have dependencies to each other. For Example "X must be done before Y" or "the cost for X decries if Y is done".[7]

When developing packaged software for a market place, the requirement engineering process should be able to invent requirements based on foreseen end-user needs and select a set of requirements resulting in a software product which can compete on the market. A packaged software product, sometimes called COTS (Commercial off-the-shelf) software, is often an integration of components. The product with its components is evolved in releases, with each release including new and improved features that, hopefully, ensure that the vendor stays ahead of competitors.[8]

Prioritisation of stakeholders

"How is different stakeholders' point of view prioritised?"

When eliciting requirements, there will most definitely be split opinions about what specific requirements actually are supposed to do, and how they will do it. There will probably be issues about if specific requirements should be used or thrown away. Somehow there has to

4 Natt och Dag Regnell Carlshamre Andersson Karlsson, Evaluating Automated Support for Requirements Similarity Analysis in Market-Driven Development, 2001.

5 Carlshamre Regnell, Requirements Lifecycle Management and Release Planning in Market-Driven Requirements Engineering Processes, 2000.

6 Höst Regnell Natt och Dag Nedstam Nyberg, Exploring Bottlenecks in Market-Driven Requirements Management Processes with Discrete Event Simulation, 2000.

7 Carlshamre Regnell, Requirements Lifecycle Management and Release Planning in Market-Driven Requirements Engineering Processes, 2000.

8 Regnell, Beremark, Eklundh, A Market-driven Requirements Engineering Process, 1998.

be a prioritisation of stakeholders. What stakeholder is most important to please, and to what extent? Can we compromise?

> "Stakeholders may have divergent interests that pull the system in different directions and which may require negotiation to resolve. Even within a single coherent set of requirements, there can be competing demands on a system. This is especially evident among non-functional requirements, where difficult tradeoffs often need to be made among requirements such as costs, performance, flexibility, usability, etc"[9]

As listed in the "Introduction"-part of this document, there are seven groups of stakeholders, all of which have their own idea of what requirements are important. In the following list, I discuss some of the stakeholders' ideas of what is most important and what impact they may have on the requirements. Notice that the stakeholders listed are those whose opinions are harder to collect, regarding to the situation, than those who are in the neighbourhood (e.g. investors, developers etc.). Hence: The listing will consist of the so called target group.

- *Storage personnel*. These stakeholders think that it is extremely important that it is easy to update the register of products in different ways. If it is not, then the work will be inefficient. Customers to the company may have to wait longer for their products and will therefore be unhappy and take their business somewhere else.

- *Managers of economics.* A system that cannot generate invoices or show statistics and so on is no good and is not worth a penny. Again, there may be delays in deliverance and an unhappy customer as a result.

- *People that handles orders coming by phone or mail.* Here it is very important that the input of orders is efficient and easy, so that many orders can be handled in a short period of time.

- *System managers.* Companies with computers have system managers. These peoples responsibility is for instance to make backups, update systems and so on. They do not want things to be troublesome. Especially so if it is a big company with many things to handle.

- *Customer to the company coming through a web shop.* Nobody will stay at a web shop for long if it takes to much time to get search results or if it takes forever to navigate from one place on the site to another.

Nobody mentioned in the list above cares much for the needs of the others. As long as the specific stakeholder gets what he or she wants, everything is fine. However, it may not be realistic to build this kind of system, which is based on all the requirements that these people say is necessary to include. Some things have to be dropped, at least in the first release, but what should it be? Normally, when there is an actual customer, meetings can be held and discussions will eventually lead to conclusions about what is important and what is not. In our case, where there is no customer but a whole market, there are some issues that may make things slightly more difficult. It is not easy to have a meeting with a whole market, so who to discuss with? Sure, a market analysis could do the trick, but how can we be sure that the result reflects the real world? Not long a go it was a parliamentary election in Sweden. Of course

[9] Yu Mylopoulos, Why Goal-Oriented Requirements Engineering

there were researches of what political party that would win the election. The result said that the Social Democrats would lose, which, as it turned out, was not the case. In fact it was the total opposite. So, can we trust the result of a market analysis enough to use it for prioritising? Another example of an issue that can be troublesome is who has the final word about requirements? This will be discussed shortly. By using a system like "CORE"[10], which is a system for organising information from multiple stakeholders, the situation may become more pleasant. This model let you create views based on stakeholders and their requirements. This will give a clearer picture of what stakeholder should have higher prioritisation. Then real requirements can be extracted from the views. Another approach to solve the problem is to use a model called "house of quality"[11]. This model take in consideration what requirements came from what stakeholder, and further more let you se relationships between requirements. This way it is possible to se if requirements are in conflict or not, and what stakeholders are involved in a certain requirement. Then prioritisations can be made with stakeholders and system requirements, etc, in mind.

The final word

"Who has the final word about a requirement when there is a conflict amongst stakeholders?"

When there are disagreements about requirements amongst the stakeholders, perhaps because the requirements are ambiguous or in conflict with each other, it is a good idea to contact the customer and book a meeting. In the meeting the issues will be discussed and if the different parties cannot come to a conclusion, the customer has the final saying. The customer wants to have it "this way", so that is the way it is going to be. But, in our case, this is nothing but a dream scenario; because in our reality we are not so lucky that we have a customer (remember). Hence the question at issue: "Who has the final word…".

For the scenario where a customer exists, there are ways of dealing with conflicts. For instance, Liu and Yen discuss a model for calculation of the priority, when in conflict[12]. I have not yet seen such a method that is specialised for markets, so it will be a bit trickier to handle the kinds of situations we are talking about. Although it is fully possible to assemble a committee of experts, there is no safe way of telling if what they say will match the real world. If there was a customer, the result would be his responsibility. The risk taken by the developer company (for instance LTL, which is mentioned in the Introduction of this document) is greater by far compared to if they had a customer. If the developer company's idea is no success, then perhaps they have to close down their business. If they had a customer, and the circumstances were the same, it would be the customer who would have to take the fall. How ever, the developing company must approach this problem if they want to win something at all. They will probably have to rely on market researches and expert committees, as well as their own hunch, or gut feeling if you will. One-way would be to wait and see what happens, but that is not to recommend, at least not if one want to profit from the idea. There will be rivals who might not wait, and if there is a market, they will get there first. Then it might be too late.

Is the requirement fulfilled

"How does one verify that a requirement is fulfilled?"

[10] Christel Kang, Issues in Requirements Elicitation, 1992, p 36, 51.
[11] Liu Yen, An Analytic Framework for Specifying and Analyzing Imprecise Requirements, 1996, p 65-66.
[12] Liu Yen, An Analytic Framework for Specifying and Analyzing Imprecise Requirements, 1996, p 67-68.

When the system at last is ready to be tested a new, but old, problem arises. Who can confirm that the system work as it should? Of course, at this point in the project we more or less have all the information necessary to build our system, so what we can do is to make sure that the system is working the way or information says it should. We can for instance use formal proof techniques[13] to actually prove that the system does what it is supposed to. We can make performance tests[14] to see if the response time is good enough. These methods will however not be worth a nickel if we were wrong from the beginning. Since there is no single person, or small group of people to satisfy, it will be hard to get useful critique about the system.

It is difficult to establish accurate requirements. Even if we manage to do so, at the point of discovering a requirement it may be hard to say if it is actually wanted, and more so if we have no customer. The people, who get to test the system, may not see if the requirement is right. They will probably see if it is correct, according to the information about it, but there is nobody to tell if the requirement as such is correct. Perhaps the requirement needs to be changed, or even removed.

[13] Shari Lawrence Pfleeger, Software Engineering Theory and practice, 1998 p 293.
[14] Shari Lawrence Pfleeger, Software Engineering Theory and practice, 1998 p 349.

Conclusion

This section is dedicated for the final thoughts and conclusions of the authors of this document.

Gustav Evertsson

The requirement engineering phase will in some ways change when moving from developing a system for a single customer to a market-driven system. It is important to know this before the developing process start. If you are going to develop more than one product it can be a good idea to make a model of the process used. That way you can use it again in the next project.

Another thing that is important to know is that maintenance is a very important quality attribute. This is because most systems today will not be released only one time, but many. It is just too expensive to develop a product from scratch today.

We have also found during the research for this paper, that stakeholders can be a bigger problem in market-driven system development than in normal development. One reason for this can be that the customers and end-users don't come from the same organisation; normally they are not even from the same country. So you can count on having conflicting requirements when talking to different end-users. I think it seams like a good idea to handle these kinds of situations with an automated lexical analysis. This is especially helpful when the requirement is collected by different people that maybe don't meet each other too often.

Jimmy Persson

The situation and its different issues discussed in this document are obviously not easy to master. There are a lot of risks included, but if the idea turns out to be a hit, and the money start rolling in, then it was probably worth it. On the other hand it could be devastating for the company…

I think that a lot of research on the subject is needed because I have not yet seen any good solutions that cover this situation as a whole. Some methods used in normal customer-developer situations, could probably be used in the market-developer situation as well. Such methods, which all are mentioned earlier in this document, are for example "the house of quality", formal profs, CORE and so on. There are however weaknesses. CORE and "the house of quality" would for instance be good tools for handling conflicting stakeholders and their requirements. When it is time to perform tests, however, there is no single person to ask about the requirements, if necessary, because there will probably not be one or two persons to ask, but a whole market. Now we can se that this will also be a problem when writing a specification for a certain requirement, because we must take the whole market in consideration and find some middle way through all opinions. This situation is much like a treadmill, where you must now A to be able to perform B, but if you do not know B, you cannot perform A, where A and B are arbitrary tasks. Another fact when dealing with a lot of stakeholders is that there will be plenty of requirements to process. Perhaps there will be so many requirements that the situation get out of hands. If this is the case, then some kind of automated lexical analysis, which was discussed above, could be the solution to the problem. The way it narrows out the lot by removing probable double requirements.

Perhaps there is no way of handling the fact that you can not discuss with the whole market, and that there is no customer to tell you that this is right or that is wrong. Perhaps we just have to live with this fact, or maybe somebody (you, your friend, or me etc) will find a general solution. If we, the writers of this document, at least have managed to sow a seed of awareness in your mind, so that when the day comes, you can approach this kind of situation with consciousness and without a clouded mind; this paper was worth the effort.

Source

Literature

Pfleeger, Shari Lawrence, Software Engineering Theory and practice, Upper Saddle River, NJ 07458, Prentice Hall, 1998.

Other sources

Written

Carlshamre, Pär - Regnell, Björn, Requirements Lifecycle Management and Release Planning in Market-Driven Requirements Engineering Processes, Research and Innovation Ericsson Radio Systems Linköping Sweden, Dept. of Communication Systems Lund University Sweden, http://citeseer.nj.nec.com/carlshamre00requirements.html, 2000

Christel, Michael G. – Kang, Kyo C., Issues in Requirements Elicitation, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, 1992.

Höst, Martin – Regnell, Björn - Natt och Dag, Johan - Nedstam, Josef - Nyberg, Christian, Exploring Bottlenecks in Market-Driven Requirements Management Processes with Discrete Event Simulation, Department of Communication Systems Lund University Sweden, http://www.tts.lth.se/Personal/bjornr/Papers/PROSIM00.pdf, 2000.

Liu, Xiaoqing Frank – Yen, John, An Analytic Framework for Specifying and Analyzing Imprecise Requirements, Department of Computer Science, University of Missouri, Rolls, Missouri 65401, Department of Computer Science, Texas A&M University, College Station, Texas 77843, 1996.

Natt och Dag, Johan - Regnell, Björn - Carlshamre, Pär - Andersson, Michael - Karlsson, Joachim, Evaluating Automated Support for Requirements Similarity Analysis in Market-Driven Development, Department of Communication Systems Lund University Sweden, Ericsson Radio Systems AB Linköping Sweden, Telelogic Technologies AB Malmö Sweden, Focal Point AB Linköping Sweden, http://citeseer.nj.nec.com/nattochdag01evaluating.html, 2001.

Nawrocki, Jerzy – Jasiński, Michał – Walter, Bartosz – Wojciechowski, Adam, Extreme Programming Modified: Embrace Requirements Engineering Practices, University of Technology Poland, http://www.cs.put.poznan.pl/jnawrocki/publica/re02-essen.doc, 2002

Persson, Jimmy- Evertsson, Gustav - Nilsson, Peter - Ohlsson, Erik, Case study, Blekinge Institute of Technology Sweden, 2002.

Regnell, Björn – Beremark, Per – Eklundh, Ola, A Market-driven Requirements Engineering Process, Department of Communication Systems, Lund University Telelogic AB Malmö Sweden, http://citeseer.nj.nec.com/regnell98marketdriven.html, 1998.

Yu, Eric – Mylopoulos, John, Why Goal-Oriented Requirements Engineering, University of Toronto, http://www.cs.toronto.edu/pub/eric/REFSQ98.html