Diffusion of Open Source ERP systems development: How Users are Involved

Björn Johansson

Department of Informatics, School of Economics and Management, Lund University, Sweden
+46 46 222 80 21
Bjorn.johansson@ics.lu.se

Abstract. Open Source receives high attention among organizations today, and there is also a growing interest on Open Source Enterprise Resource Planning (OS ERP) systems. Open source development is often considered having a high level of involvement from stakeholders in adopting organizations. However, it depends on for the first what is meant by stakeholders, but also what is meant by involvement. In the area of ERP development stakeholder involvement is defined as to what extent users of the system are involved in the development of the standardized software package. The way this is done differs between different vendors but it can be summarized as dealing with management of requirements. In this paper we explore how requirements management is done in development of OS ERP. To do this, we use a theoretical base on requirements management in the ERP field from which we investigate stakeholder involvement in four organizations and the development of it's respectively OS ERP system. The basic question asked was: how are end-users of OS ERPs involved in the development of OS ERP. From the investigation we present a general picture of the requirements management process in the OS ERP area. The main conclusion is that end-users are not involved to the extent first expected and when comparing with proprietary ERP development a tendency towards a similar approach to requirements management in OS ERP development was discovered.

Keywords: Requirements Engineering, Enterprise Resource Planning, ERP, Open Source, Standard systems, user involvement.

1 INTRODUCTION

Many organizations implement enterprise resource planning (ERP) systems from the view that they will obtain an advantage over its competitor (Verville et al., 2007), by for instance getting better and faster access to information stored. However, there are organizations that are resistant because ERP implementation takes time and costs a lot of money, while there is also a high risk that it may fail (Daneva, 2007; The Standish Group, 1995; Verville et al., 2007). This has created an interest in developing ERPs in the open source (OS) area. The open source ERP development could be seen as an alternative to traditional ERP proprietary systems available today.

Rapp (2009) suggests that OS have grown large and states that it continues to grow strongly as more and more organizations become interested in how they can benefit from OS in their organization. However, the same thing can also be stated about ERPs. But, the question is then if combing open source and ERPs is that a doable and beneficial combination for organizations. It can be claimed that if it should be, development probably needs to be done with closeness to its users. An interesting question to ask is therefore to what extent and how users are involved in OS ERP development.

According to Fitzgerald (2006) open source development differ from “ordinary” software development since the first phases of planning, analysis and design usually has no clear borders, and is performed by a single developer or a small group of developers. This is explained in a survey by Zhao and Elbaum (2003), showing that a large proportion of projects in OS are developed by self-interest and only a small part are developed in response to different organizational needs. Lemos (2008) states that OS ERP is a growing market, and that there is a growing commercial interest in it. Fitzgerald (2006) argues that since OS projects will be more commercial, a greater level of structure and control in development is needed. Sommerville (2007) supports this by observing that in development of large systems, such as ERPs, the high complexity is a major problem. Young (2001) provides the following view of requirements management in software development: "We often record requirements in a disorganized manner, and we spent far too little time verifying what we do record. We allow change to control us, rather than establishing mechanisms to control change. In short, we fail to establish a solid foundation for the system or software that we intend to build ".

From this discussion the question which is discussed in this paper is: How are end-users involved in the development of Open Source ERPs? To answer this, a description of four OS ERP developer organizations and their respectively OS ERP system are presented (section 2). Section 3 then presents an analysis on how requirements management is done in the four organizations. The final section then presents some conclusions on the question how end-users are involved in development of OS ERP and some future research directions in that area.

2 FOUR ERP DEVELOPER ORGANIZATIONS AND THEIR OS ERP SYSTEM

This section reports from an investigation done in four organizations that have the commonality that they all develop what is marketed as open source ERP systems. Data was collected in semi-structured interviews with representatives from the organizations. Respondents were selected from the perspective of having high level of knowledge about how development is done in respectively organization. In addition to the interviews, data was gained by investigating documents from respectively organization’s websites. The interviews were tape-recorded and then transcribed. Follow up questions was asked in order to clarify some uncertainties. Investigated organizations were: Project Open, Night Labs, Openbravo and Open Source

Strategies, and the interviewee has the following role: Founder, CEO, product manager, and product manager.

2.1 Project Open

Project Open develops an OS ERP system with the same name. They also offer various types of consulting services relating to adaptation of the system within organizations (beyond normal support).

Project Open's development is highly controlled and are usually based on customer requirements and direct funding from customers. The main source for requirements is individual customers. However, they also take into account requirements from potential customers. Project Open has a limited scale of development that starts from self-initiated projects without involving a customer. These projects are based on experience from past client projects. Some requirements also arise internally from own ideas, and to a limited extent requirements are gained from the community around Project Open. Prioritization of stakeholders was described as:

"Well, you know everybody who basically finance and sponsor development, will be considered in the first place." (Founder of Project Open)

The majority of requirements are gathered in meetings with representatives of the customer and users related to a specific customer project. During these meetings, they usually organize a workshop at the customer organization place for development of requirements with users. Requirements are also partly obtained from the community forums on sourceforge.net, but, since the community does not provide any funding, this source of requirement is not of high priority. At the moment Project Open also works with building a repository of requirements from four other projects, and plans to post them on a website to avoid having to start from scratch with future similar requirements. During customers' workshops, cost estimates on requirements are created. After this Project Open develops and uses prototypes to visualize requirements which they later on discuss with the customer. It usually leads to decisions about how to continue. If there is a need for requirements prioritization due to various constraints, the customer always has the last word. In situations where they feel they need more feedback about a specific requirement within a short period of time, they tend to repeat the workshop activities, as described by the founder of Project Open in the following way:

"In some very rare cases we sometime would need to organize like a second workshop to dive deep into particular questions that have come up during this stage and only then these are required - when we talk about more complex extensions, configuration of workflows and so on."

Project Open visualize requirements through prototypes, besides this they also describe them in conjunction with a form of use cases. This is done to make sure that requirements are understood correctly, and to make it possible to have feedback on requirements from the customer. During the requirements engineering process changes in requirements are managed through an iterative process. During requirements engineering activities documentation of requirements are done in

parallel, by formulating requirements in a specific document. This documentation is then used as a manual for the system.

2.2 Night Labs

Night Labs is an organization that develops OS products and offers consulting services in the OS software development field. They develop an OS ERP system called JFire. Night Labs provide their own support and adjustments of their own developed systems.

Software development by Night Labs is divided into different projects, each project having its own requirements management process. Requirements are gained by talking to the customer. This is the major source of requirements, the idea is to have customers to describe requirement from the context of usage. Requirements are also obtained from a community forum where developers and users can discuss and also provide feedback on requirements. Requirements are also gained from discussions within Night Labs. They also analyze requirements in competing systems. Night Labs put great importance to investigate whether requirements are specific to a specific customer or if it is possible to adjust them to fit more customers. Recycling of requirements is done by analyzing requirements of modules that they already have developed for a specific customer. They then investigate whether it is possible to implement the system and making it useful for another customer, with or without changes. The CEO of Night Labs says:

"Of course it happens sometimes, that we think: 'Ah, this is very specific´ then we implement it specifically and then maybe later someone else comes with more of the same thing, and only sometimes minor changes and then of course we go back and take the previous project, and analyze what can we pull out".

Night Labs state that dependent on the development situation there is to some extent different demand on management of requirements. An individual project often gathers a larger amount of requirements in the beginning. After receiving some feedback the major changes primarily relates to contextual changes because it was misunderstood from the beginning. If several projects run in parallel, it could mean that requirements to a higher extent will be gained during the projects because of overlaps between the projects' requirements. It is also stated that requirements continuously comes from the community. Once Night Labs have gathered a variety of requirements for a project, they try to investigate how the requirements can be implemented and provide feedback to customers aiming at resolving uncertainties. At Night Labs they argue that customers usually do not provide adequate information about their demands so they could implement requirements directly. The CEO states that:

"In this situation we basically have most of the requirements analysis at the beginning and then we have during the project only analyses of maybe misunderstanding and adapting to the requirements, to fit the needs of the customer better".

To handle this situation they therefore develop use cases which they use when discussing requirements with customers. These use cases are then used to verify that

all requirements are consistent with what the customer wants and expects. This activity takes place regularly. The CEO, however, said that there often is confusion and it is important to work closely with the customer throughout the process to quickly ensure that they are on the right track. Night Labs also present prototypes consisting of an early version of implemented requirements for the customer, to obtain feedback confirming that their understanding of the requirement is correct. If constraints arise regarding implementation of a number of requirements within the same project, Night Labs conduct priority activities focusing on deciding on what requirements to be implemented first. They present the situation for the customer, after which the customer should select the items to be prioritized. However, sometimes they need to do the requirements prioritization by themselves. The prioritization is then based primarily on the basis of what benefits the project most and what implemented requirement benefits the largest number of users. The CEO says:

"So we have to invest from our own money, so we have to decide if this is a benefit for the project in general, for us as a company, whether it benefits" a wide range of users or whether if it is a very, very individual thing".

Developed use cases have the roles as both contract and documentation between Night Labs and the customer. Regarding communication within the community forums this communication is automatically saved and stored on a wiki. Customers may self-prioritize requirements they consider most important in their projects but Night Labs has to interact and say what is possible, considering all necessary constraints. The CEO describes this in the following way:

"Of course we make constraints, if he says: 'I want something' and we know that for this something, something else needs to be done and he says: 'Ah, but this I want later' then we have to tell him: 'Yeah, sorry, but you can't build the roof without first building the house'."

Night Labs in a similar way involve the community by constantly showing were in the process of implementing a requirement they are and collecting feedback from the community in that matter. Changes on requirements are handled in what is termed as "Change Request". These changes are handled differently depending on what stage the project are in when they are discovered. If changes occur before implementation has started they are according to Night Labs not a big problem. But, if there is a change in requirements that have already been implemented, it is a bigger problem. Night Labs then solve this problem by dealing with it in the next version of the system.

2.3 Openbravo

Openbravo is an organization that develops the OS ERP system Openbravo ERP. Special customization and other services around Openbravo ERP are done by partners who are scattered around the world.

Depending on type of development there is a variation on how requirements are gathered. Requirements come from partners, the community related to Openbravo, as well as internally. Partners are seen as the main source of requirements for future

development of Openbravo ERP. Requirements management is done through close contact with the originator of the project and is constantly ongoing. All sources, however, plays a role, a product manager at Openbravo describes it as:

"Our Requirements are driven of what the partner's need, but also" in a large part by what we believe need to be done to strategically move the product forward and provide a best of breed platform for developers."

If there is a partner involved in future development of requirements, this is discussed in meetings and/or forums with the partner. Another way of requirements gathering is by monitoring community forums. After the specific context for the requirements is described, requirements are visualized by various forms of prototyping in order to produce feedback, and to confirm that the resulting requirements are correct. These are then presented either for customers or for the whole community, through meetings, forums and blogs. The prototypes are mainly based on requirements from external stakeholders. However, on some occasions, prototypes are developed from internally gained requirements; based on what developer at Openbravo think users want to see in the system. The prototypes in that case also aim at generating feedback from potential users. The product manager gave the following description of the process:

"We actually start our requirements gathering from the perspective of what will the user see, what interface, what screen the user will be presented with and the workflows, and we work backwards from there."

Prioritization of requirements is sometimes done by a partner. This is the case if the partner provides money for specific functionality to be implemented. If so, there is a discussion with the partner in order to reach an agreement over what should be done. But, in most cases requirements prioritization are based on what they believe is best for users, however, sometimes they let users vote on what requirements should be implemented. Openbravo is working on what they call its backlog which is a list of requirements and its priority. This backlog consists of three pillars: the first pillar is "maintain" that aims at constantly keep Openbravo far ahead among the competitors, which is described by the product manager as:

"One is that we want to maintain our competitive edge and also that with our partners."

The second pillar is "Delight" which aims at increasing ease of use for users of Openbravo ERP. Described by the product manager as:

"That is we want to develop features and functionality that would delight our users (...)".

The last pillar is "Monetize" which aims to provide an economic value to the product, which is described by the product manager in the following way:

"These are features that allow Openbravo and its partners to make money."

When a partner is involved a specification of the requirements are developed before implementation begins. The specifications are posted on a forum where feedback can be obtained and were Openbravo developers are able to direct questions about the requirement to the partner. Documentation of requirements continues the whole time during the project and is done in parallel with development. Openbravo is aware of the fact that requirements changes and therefore has Openbravo, according

to the product manager, a flexible approach to managing change. However, when implementation has started changes become more complex. Changes are handled through the forum, where developers and other stakeholders, according to the product manager, can have a dialogue about change:

"And we have a very interactive approach in defining the requirements for our customers, and while that is going on, any change is permissible, until the point that we start development (...)".

2.4 Open Source Strategies

Open Source Strategies is an organization that develops Opentaps and whose purpose is to promote OS development in general. They also provide consulting services, mainly related to adjustments of Opentaps, but also on evaluating, testing, certification, training in development and integration of the software.

The main sources for requirements are, according to a product manager for Opentaps, users of the system and the Opentaps related community. However, some requirements are gained internal, during own development when the system is used by themselves. Opentaps also have customer projects where customers are responsible for presenting requirements. These customer projects have the highest priority. Gathering of requirements are in part concentrated to the start of the project. The major part of requirements is collected at the beginning by discussions with the customer and its users. All requirements from the collection add up to what the product manager termed as an issue-tracker. However, the product manager adds that new requirements arise during the project, primarily through the monitoring of community forums, but also internally during the time they develop the system. The gathering of requirements is done in various ways, for instance by being contacted by users who produce their own needs; these are then discussed to develop a stringent set of demands. The product manager said:

"Then basically we would go to the user and discuss exactly what they're looking for, and we would put it usually on an issue tracker to describe what their requirements are."

Sometimes Opentaps create prototypes that are presented to the community and based on these prototypes experience on requirements is gained. In addition Opentaps sometimes also use scenarios describing the context as they have understood it during discussions with users. They then use the scenarios to achieve feedback on requirements and its context. Reuse of requirements is also an aspect of their requirements management, which is described by the product manager in the following way:

"In this way it's a bit like a city: each building in the city represents the requirements of its residents at one point, and the buildings are reused and modified over time to meet new requirements".

At Opentaps changes in requirements are handled by introducing changes in the system, which then undergoes testing and thereafter are documented. This may be due to the iterative process they use, which the product manager describes as:

"It is just an iterative process of the system being used of people and shown to people and their feedback gets incorporated and the system is modified."

Requirements are undergoing discussions and various tests with the customer to ensure that requirements are correctly understood. Stakeholders in the community forums also provide feedback through the various scenarios presented in the forums. The requirements prioritization that takes place depends on various factors based on whether it is a customer project or not. For customer projects, the customer has the final say regarding what requirements should be prioritized. If not, prioritization of requirements is done from the number of users that Opentaps believe would have the function, or depending on the number of users who have contacted them and asked for a specific function. The product manager said:

"So for example, a lot of people will come to us and say 'you know we would like this" and that, if it is something we think will benefit a lot of users we will give it a higher priority".

Documentation of Opentaps then takes place after the functionality has been implemented. This documentation is then used as user manuals and to preserve requirements for future use. This is described by the product manager in the following way:

"Usually after the feature has been implemented, for the purpose of allowing other people to use the feature and also documenting the requirements for later use".

3 ANALYSIS AND DISCUSSION

The following section highlights both similarities and differences in how end-users are involved in development of the four organizations respectively OS ERP system. The description above is analyzed and discussed from literature around software requirements under the following headings: 1) requirements elicitation, 2) requirements analysis and negotiation, 3) requirements documentation, 4) requirements validation, and 5) requirements changes.

3.1 Requirements elicitation

It is possible to distinguish similarities and differences in how the different organizations collect requirements. Three of the organizations have a more concentrated collection of requirements at the start of the project. However, there is then an ongoing less extensive requirements elicitation in all cases. Openbravo differs from the other organizations since it has a continuously requirements gathering process where they constantly repeat their activities to gather requirements. The requirements management process in all four organizations suggest that they take into account changes on requirements related to its context, just like Hickey and Davis (2003) as well as Kotonya and Sommerville (1998) state being necessary. This is also in line with the statement from Eriksson (2008) who argues that it is difficult to gather all requirements at one single point. All interviewees claim that the organizations focus a lot on customers when gathering requirements, just like Davis (2005)

describes as the key stakeholders in a system development project. However, it is only Open Source Strategies, which explicitly says that they interact directly with users when they discuss requirements. The other organization seems to more indirectly discuss requirements with other stakeholders and not directly involve end-users. It is also stated that requirements are gained internally within the organizations, except in the Project Open case. All organizations use community forums to develop standards to some degree, however, the extent of their importance as a source for requirements differ between the different organizations. Project Open does not involve the community in the requirements gathering to any large extent, while others take more account of different communities. There is no clear idea of what types of stakeholders that are involved in the community forums. But, it is possible to speculate that it probably involves different groups of stakeholders, including users. This may explain why organizations do not directly approach the users to gather requirements. The community may also consist of developers and other stakeholders. It can be assumed that a high interest in the system is needed for any stakeholders to be active in the community. Therefore, there is possibly a mismatch between various stakeholders involved on the forums. The prioritization of stakeholders is consistent within the organizations, since all are taking the greatest account of the one that provides funding. Group Meetings is also a regular feature of the organizations, however, Project Open uses workshops instead. Both activities were based on individual group meetings with discussions between stakeholders and developers. Furthermore, it is possible to distinguish differences in what activities and technologies that they choose in their requirements elicitation process. These differences can be explained from Jiang et al. (2005) who argue that the choice of techniques may be different because of stakeholders' knowledge, or from Hickey and Davis (2003) as well as Zowghi and Coulin (2005), whom argue that selection of technologies is based on the context for the requirements. Reuse of requirements is done only in three of the organizations. Recycling is most pronounced in Project Open, which has built storage of requirements from previous projects, to avoid repeating the process to develop similar requirements. This is in line with Robertson and Robertson (2006) who argue that recycling can be the basis for new requirements. Night Labs only reuse requirements from already implemented modules, however, despite that changes are usually required to be made on the requirement. Their reuse demands insights into parts of the system. Reuse of Open Source Strategies takes place over time, without any specific plan. In Openbravo reuse is not a widespread activity, the reuse of requirements that is done is done primarily through the reuse of code. Table 1 provides an overview of requirements elicitation in the different organizations.

Table 1. Requirements elicitation

Organization	**Project Open**	**Night Labs**	**Openbravo**	**Open Source Strategies**
Description	Requirements gathering begins with a high number of requirements gathered from the customer at the beginning of the project. The community plays no significant role in gathering of requirements.	Requirements gathering take mainly place in the beginning of customer projects with customers and their users, but there is also an ongoing gathering of requirements from the community.	Requirements gathering take mainly place in the beginning of customer projects with customers and their users, but there is also an ongoing gathering of requirements from the community.	Requirements gathering take place in an ongoing fashion. However, the majority of requirements are collected, at the beginning of projects.
Activities/ Techniques	Workshops, prototypes, community forums.	Group meetings, use cases, community forums.	Group meetings, prototypes, community forums.	Group meetings, community forums.
Reuse of requirements	Building up a stock of ready-made requirements from previous projects that must be used when having related demands, to avoid the need to repeat the process. However, changes are often required.	Re-use of requirements is done by analysis of previous projects with similar requirements. Requires usually small changes.	No reuse of requirements.	Some reuse exists when requirements change. Previous requirements can be the basis for new demands.
Where do demands come from?	Customers, potential customers, internal, community.	Customers, internal, community, market analysis	Partners, community, internally	Customers, users, community

3.2 Requirements analysis and negotiation

The use of prototypes is one way among the organizations to analyze requirements. The prototypes are presented to various stakeholders to identify problems. This is consistent with the view that Kotonya and Sommerville (1998) give on the analysis and negotiation phase. However, Night Labs differ since they do not use prototypes. Within all organizations there are also other activities designed to create discussion about requirements. Three of the organizations are following group meetings, while Project Open uses workshops to discuss the same issue. Both activities promote discussion of requirements but they differ to some extent. Workshops are seen as a more formal and structured activity than group meetings, however, both types of discussion activities are designed to arrive at agreement on requirements. All organizations use a systematic approach to analyze requirements and describe the context for requirements, but also to find out if they have developed legally ok requirements. This is consistent with one of the main activities in the analysis phase that Sommerville and Sawyer (1997), Kotonya and Sommerville (1998) as well as Wiegers (1999) chooses to emphasize, claiming that this are distinctive part of the analysis and negotiation phase in requirements management. It is common for developer to prioritize customer requirements in the projects they are involved in. The organizations involved in internal projects gives priority based on what would benefit a major part of users. This statement is consistent with Karlsson (1996) who describes, the importance of stressing customer satisfaction when prioritize requirements. Prioritization in Openbravo is mainly based on what they call "Monetize", which basically means that requirements that allow Openbravo and its partners to make a profit should be prioritized. The negotiating process in the described organizations is consistent with the structure Kotonya and Sommerville (1998) present in which a discussion, prioritization and agreement are included. Table 2 gives an overview of how the analysis and negotiation of requirements are done at the various organizations.

Table 2. Requirements analysis and negotiation

Organization	**Project Open**	**Night Labs**	**Openbravo**	**Open Source Strategies**
Description	Analysis of requirements is done through discussions with the customer, as part of the workshops conducted when gathering requirements.	By holding discussions with stakeholders sorting out confusion on requirements. Takes place when a variety of requirements has been developed.	When a prototype is produced, they present it to the community and partners who provide feedback.	Discusses and shows the customer functions in a prototype, collect feedback. This is done after a set of demands is developed.

Organization	Project Open	Night Labs	Openbravo	Open Source Strategies
Activities / Techniques	Prototypes, workshops	Group Meetings	Group meetings, prototypes, community forums and blogs	Group meetings, prototypes.
Prioritization of requirements	Customer focused, the customer prioritizes the requirements of the projects.	Customer prioritizes their requirements. Night Labs has a large number of parameters which they use when they prioritize in their own projects.	Partner priority, when a partner is involved and pay. Otherwise priority of requirements is based on the three pillars.	Customers prioritize requirements for their projects. OpenTaps prioritize according to what benefits the most users.

3.3 Requirements documentation

Requirements documentation in Project Open, Night Labs and Openbravo is done in an ongoing fashion during development projects. The ongoing documentation is a way of dealing with changes among requirements and to be more flexible. The documentation process in Open Source Strategies differs, and documentation is done first after requirements have been implemented. The aim of the documentation varies between the organizations. However, all organizations except Openbravo are using the documentation as a base for user manuals. Night Labs documentation is also used as a contract between them and their customers. Openbravo are using the documentation as a support to developers. The way documentation is done also differs to some extent, however, they all document requirements in what they call a requirements specification. Night Labs in addition also document requirements in use cases, which is in line with what Eriksson (2008) describes as one way of using use cases. An overview of requirements documentation in the organizations is shown in Table 3.

Table 3. Requirements documentation

Organization	**Project Open**	**Night Labs**	**Openbravo**	**Open Source Strategies**
Description	Documentation is ongoing and consists of specifications.	Documenting everything on the system that runs. The record is done by use cases.	Documentation is ongoing with the project. Specification of projects linked to a discussion forum.	The documentation is done after requirements have been implemented in projects.
Purpose	Documentation is used as a manual for the system.	Documentation serves as the contract for the customer and help for the user.	To provide support for the developer to communicate changes.	To ensure the requirements and provide assistance to users.

3.4 Requirements validation

All organizations except Open Source Strategies use prototypes to get feedback on requirements they have developed. Project Open does not address the community in relation to feedback on developed requirements, while the rest do so. Even if all, except Open Source Strategies, use prototypes the way they use prototypes differs. What differs is the way they present a specific prototype. Three of the organizations have validation activities that are repeated during the requirements engineering process, which is in line with Wiegers (1999) as well as Loucopolous and Karakostas (1995) description of doing validation activities throughout the entire requirements engineering process. However, Open Source Strategies differs since they perform this activity only at the end of the development project. All organizations have a discussion with the customer aiming at validating requirements. The community is also involved in validation of requirements, except in Project Open, which consistently excludes the community from the entire requirements management process. A common feature of organizations is that they perform the various activities in order to check whether they have got the requirements right, which is consistent with the view that Kotonya and Sommerville (1998), and Sailor (1990) describes as one characteristic of the validation phase. Table 4 provides an overview of how organizations deal with requirements validation.

Table 4. Requirements validation

Organization	**Project Open**	**Night Labs**	**Openbravo**	**Open Source Strategies**
Description	Collect feedback on requirements and discuss it with the customer in an ongoing fashion.	Collect feedback and discuss it with the customer and the community regularly during the project.	Partners are closely involved in the development, and through frequent contact with the community, feedback on requirements is discussed frequently.	Through discussion with the customer and the community. This is done at the end of the requirements management process.
Activities / Techniques	Review, prototypes, use cases.	Community forums, reviews, prototypes, use cases.	Community forums, reviews, prototypes.	Community Forum, Review, test-case.

3.5 Requirements changes

All organizations are working continuously to cope with changes that occur during the requirements engineering process, which is consistent with the picture Kotonya and Sommerville (1998) give as necessary when dealing with requirements. This suggests that all organizations are aware of that the context of requirements is changing, as several authors point out (Davis, 2005; Eriksson, 2008; Kotonya and Sommerville, 1998). All organizations are taking account of changes if they occur before the development is initiated, while there are clear discrepancies in how they work to manage these changes later in development. It is possible to distinguish the two organizations, Labs Night and Openbravo, from the other since they have distinct techniques to manage changes and change requests. Project Open and Open Source Strategies does not have an explicit technique to manage change. However, they deal with changes by having an iterative approach. Table 5 gives an overview of the process of requirements changes in the organizations.

Table 5. Requirements changes

Organization	**Project Open**	**Night Labs**	**Openbravo**	**Open Source Strategies**
Description	Changes occurring early in the development project are taken into account more or less directly. After development has started only minor changes are considered.	Changes are considered if related requirement not have been implemented, otherwise changes are managed in the next version of the system. Change request is used to manage changes.	Changes are generally allowed before development starts. Then it's up to the one who request the change on requirement to take decisions. The forums are used to manage changes.	The work is done in an iterative process that allows that changes in requirements can be handled. Changes are implemented directly in the system.

Summing up the description above which was based on the interviews, but also on the analysis of similarities and dissimilarities also presented above, give the following summary. Regarding requirements elicitation, it can be stated that concentrated requirements gathering takes place at the beginning of the projects. But, there are also an ongoing requirements gathering during development of the projects. Involved stakeholders are customers, community, and internal stakeholders. Customers of the developed product are the prioritized stakeholder group. The major technique used for requirements elicitation is group meetings and workshops, and community forums. There is also to some extent a reuse of requirements from previous projects.

During the analysis and negotiation phase group meetings and workshops, and prototypes are used to analyze requirements. In customer projects the prioritization is done by the customer. In internal projects the prioritization is done from the perspective of what will be most beneficial for the biggest part of users. Negotiating activities are done during the analysis of requirements. All companies consult stakeholders, both through structured discussion activities, and through more informal approaches.

Requirements documentation is ongoing in many parts of the requirements management process. The documentation is then often used as a user manual or act as a base for the user manual. To a high extent the documentation builds on requirements specifications.

Requirements validation is an ongoing process, Reviews of requirements and prototypes are used to validate requirements. In the process the community and users are involved.

Regarding requirements changes is this done continuously until the development is initiated and has started.

3.6 Conclusions and future research

The research presents a representation of requirements management processes in Open Source ERP development. From this some conclusions can be drawn on the question asked in the paper, which are: How are end-users involved in the development of Open Source ERPs?

Requirements elicitation is done as a concentrated collection of requirements in the beginning of the development project, followed by a concurrent less extensive requirement collection during the project. There is also a reuse of requirements from previous projects. From the question on end-users involvement it can be concluded that users are involved but not to a higher extent than in other standardized software package development. There is a tendency that end-users are involved to a higher extent than in proprietary ERP development. However, if critical reflect on this it can be said that this tendency is to a very high extent similar to involvement in customization/configuration of proprietary ERP software package.

There are a number of different ways in which stakeholders are involved. The most common techniques consist of personal meetings with customers, aiming at creating discussions about requirements. Community forums are used to stimulate dialogue about context of requirements and produce feedback on specific requirements. Prototypes are used to show stakeholders how work is progressing and to ensure that the requirements corresponding to stakeholder needs. Requirements are then visualized to users and analyzed through close meetings in which they also are negotiated upon.

It can also be concluded that to some extent are end-users involved in prioritization of requirements, however, interesting to notice is that prioritization to a high extent builds on what the developing OS ERP organizations see as beneficial for them. In other words, if the customer pays for implementation of the specific requirement or if they see that they will earn money from the implementation in another way, for instance gaining a competitive advantage in relation to competitors in the OS ERP space, the specific requirement will be prioritized.

The research shows that there is a structured requirements management process in the development of open source ERPs. Future research on how this process is related to the often stated benefit, flexibility of Open Source ERPs, would be interesting to conduct.

Acknowledgement

I would like to thank my former students Alexander Persson and Oscar Sjöcrona at department of Informatics, Lund University for their assistance with helping me collecting the data for this research.

References

Daneva, M. (2007). Understanding Success and Failure Profiles of ERP Requirements Engineering: an Empirical Study, 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO 2007).

Davis, A. M. (2005) Just enough requirements management: where software development meets marketing. New York: Dorset House Publishing.

Eriksson, U. (2008) Requirements engineering for IT-systems (in Swedish: Kravhantering för IT-system) (2nd ed.). Lund: Studentlitteratur.

Fitzgerald, B. (2006) The Transformation of Open Source Software. MIS Quarterly, Vol 30, No. 3.

Hickey, A. M. and Davis, A. M. (2003). Elicitation technique selection: how do experts do it?, 11th IEEE International Requirements Engineering Conference.

Jiang, L., Eberlein, A. and Far, B. H. (2005). Combining Requirements Engineering Techniques - Theory and Case Study, 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems, 2005. ECBS '05.

Karlsson, J. (1996). Software requirements prioritizing, the Second International Conference on Requirements Engineering.

Kotonya, G. and Sommerville, I. (1998) Requirements engineering: Processes and techniques. Chichester: John Wiley & Sons.

Lemos, R. (2008) Open Source ERP grows up. CIO.com http://www.cio.com/

Loucopoulos, P. and Karakostas, V. (1995) System requirements engineering. Berkshire: McGraw-Hill.

Rapp, J. (2009) Increased Interest for Open Source (in Swedish: Ökat intresse för Open Source.) Logica, http://www.logica.se

Robertson, S. and Robertson, J. (2006) Mastering the requirements process (2nd ed.). Upper Saddle River: Addison-Wesley.

Sailor, J. D. (1990) System Engineering: An introduction. In I. R. H. Thayer, & M. Dorfman (Eds.), System and software requirements engineering: 35-47. Washington: IEEE Computer Society Press.

Sommerville, I. (2007) Software Engineering (8th ed.). Harlow: Pearson Education Limited.

Sommerville, I. and Sawyer, P. (1997) Requirements engineering: a good practice guide. Chichester: John Wiley & Sons Ltd.

The Standish Group. (1995). CHAOS Report http://www.projectsmart.co.uk/docs/chaos-report.pdf

Verville, J. J., Palanisamy, R., Bernadas, C. and Halingten, A. (2007) ERP Acquisition Planning: A Critical Dimension for Making the Right Choice. Long Range Planning, Vol 40, No. 1.

Wiegers, K. E. (1999) Software Requirements. Redmond: Microsoft Press.

Young, R. R. (2001) Effective Requirements Practices. Boston: Addison-Wesley.

Zhao, L. and Elbaum, S. (2003) Quality assurance under the open source development model. The Journal of Systems and Software, Vol 66, No. 1.

Zowghi, D. and Coulin, C. (2005) Requirements Elicitation: A survey of techniques, approaches and tools. In I. A. Aurum, & C. Wohlin (Eds.), Engineering and managing software requirements: 19-46. Berlin: Springer.

AUTHOR BIOGRAPHIES

Björn Johansson holds a PhD and a Licentiate degree in Information Systems Development from the Department of Management & Engineering at Linköping

University and a Bachelor degree in Business Informatics from Jönköping International Business School. He defended his doctoral thesis “Deciding on Sourcing Option for Hosting of Software Applications in Organisations” in 2007. Currently he works as an Associate Senior Lecturer at the Department of Informatics at School of Economics and Management, Lund University. Before that he had a Post Doc position for three years at Center for Applied ICT at Copenhagen Business School, within the 3gERP project (http://www.3gERP.org). He is a member of the IFIP Working Group on Diffusion, Adoption and Implementation of Information and Communication Technologies (IFIP TC8 WG8.6), the IFIP Working Group on Enterprise Information Systems (IFIP TC8 WG8.9) and the research networks: VITS Work practice development, IT usage, Coordination and Cooperation and KiO Knowledge in Organizations.