WEB APPLICATION DEVELOPMENT AND QUALITY – OBSERVATIONS FROM INTERVIEWS WITH COMPANIES IN NORWAY

Sven Ziemer and Tor Stålhane
Department of Computer and Information Science, NTNU
NO-7491 Trondheim

Keywords: Web Application Development, software quality, development practises, trade-off.

Abstract: In this paper we present our findings from a series of interviews with companies developing web applications, investigating how quality issues are managed when developing web applications in a rush-to-market and competitive environment. Our findings suggest that requirement practises are communication intensive, that companies perceive quality attributes related to a good user experience important, and that companies don't have a clear trade-off situation.

1 INTRODUCTION

Web Applications have become part of our every day life. Hence, the quality of web applications are of interest to all users. We take a special interest into how quality issues are managed in web application development. In this paper we present our findings from a series of interviews with Norwegian companies [1].

The paper is organised as follows: We start with the background for our study in section 2. The findings from the interviews are presented in sections 3, 4 and 5. In section 6 we discuss our findings and conclusions are given in section 7.

2 BACKGROUND

We interviewed representatives from seven Norwegian companies developing web applications. The interviewees had different roles in their companies: IT manager (3 companies), developers (3 companies), and development manager (2 companies). For one company, we had two interviewees. We used a semi-structured interview. The selection of companies to our sample were partly based on their willingness and availability for the interview and on the type of web application they are developing. We were approaching companies developing internet based web applications, that are critical for the companies business, that are used in the marketing strategy of the company and where TTM is important.

The companies selected for the interview differ quite a lot, both with respect to size, organisation and what they are developing. Some stats about the companies are given in figure 1.

[1]The work is port of the WebSys project, which is supported by the National Research Foundation in Norway

3 REQUIREMENTS

Eliciting and specifying requirements are important steps in the development cycle of every software system. How do companies find the right balance between specifying enough details for the requirements and the development time. In our view this is a trade-off between reducing risk and TTM.

We found that requirement practises where depend on the ownership of the web applications: whether the development where done in-house or bespoken.

We asked the companies in our sample the following questions: (1) How are requirements elicited? (2) How are proposals for new requirements and change request managed? (3) How are requirements specified? (4) How are requirements validated?

3.1 Elicitation

Requirements where elicited in a number of ways. The most obvious way are customer given requirements. Four companies had a formal approach of writing requirements, including both companies developing bespoken software. These companies used

Company	Domain	Company Size	Project size	Deployment frequency	Type of development
1	Travel	8	3	Daily on contenct; bug fixes and improvement every 2nd week; user-interface remake every 18 months	In-house development
2	Media	650	3 – 4	Bug fixes weekly; new functionality whenever customer dependent.	In-house development
3	Service	80	4 – 6	2 releases every year; project time for each release is 4 – 6 month´s. Custom releases: project time is 1 – 6 months.	In-house development
4	Travel	20	3 – 5	Minor uppgrades every 3rd week, new services every 3rd month.	In-house development
5	Development for customer	10	1 – 3	Project delivery for customer; typical project time is 1 – 2 month´s.	Development for customers
6	Development for customer	25	1 – 3	Project delivery for customer; typical project time is 1 – 2 month´s.	Development for customers
7	Finance	250	5 – 10	Scheduled releases; typical project time is 4 – 6 month´s.	In-house development

Figure 1: Statistics on the interview sample.

the requirements specification as part of an agreement with the customer. Two of the companies developing in-house software used domain experts to elicit requirements and two where following a development process. The other companies used oral communication and email communication in an informal way.

Requirements where also elicited by collecting user feedback. This can both be error messages and more general improvement suggestions. We found that user feedback was used mostly by companies developing in-house software, where developers have direct contact with the users. Companies developing bespoken software have to discuss the user feedback with the customer to decide what to include.

Two companies used to specify requirements for marketing purposes. These where collected from comparisons with competing applications, from strategic planning to make the web application more attractive to new user groups, and from user satisfaction surveys. Both companies are developing their software in-house.

To sum up, the companies developing in-house seem to have more informal requirement elicitation practises. They rely on oral communication and on the domain knowledge of their developers. Companies developing bespoken software have in general more formal development practises, mainly because of the contractual relationship with the customer.

3.2 Specification

The two companies developing bespoken software produce a written requirements specification, which is then part of the contract between the companies. Requirements are detailed according to the information given by the customer.

Practises found in in-house development are more informal. Most requirements are presented and detailed in oral communication. This can take the form of ”this would be a nice function to include”, or ”do you remember what we did on another system one year ago? That would work fine for our new system too, with some changes”.

As mentioned earlier, user feedback is used to elicit requirements. Both error messages and improvement suggestions are received in written form, by email or chat. But another part of the user-feedback is the hands-on experience resulting from user support. Typically, this information is not documented. The details of the requirements and change requests are in the understanding of the developers, as a sort of tacit knowledge.

When the requirements have to be clarified or harmonised, this will be done either in conversations between two developers or in team workshops. If necessary, requirements will be detailed with some sketches or with a simple user interface mock-up. Some of the companies are producing written notes from such meetings.

When working with written requirements, these were presented as either plain text, use cases or screen shots. One company used the PLanguage language (developed by Tom Gilb, see (Gilb, 1989) and (Gilb, 2004)) to specify their requirements.

3.3 Requirement Validation

When requirements are not specified in details, it is hard to validate them. Therefor, most companies where relying on user feedback for requirement validation. Only two companies from our sample do have a dedicated testing department. We encountered a number of strategies to validate the product:

- *The web site is validated by the users.* New version are developed to fulfil the expectation of the developers. After publishing, the application will be changed in accordance with the user feedback.
- *A prototype is validated by the customer.* Customers can evaluate a prototype of the web application. Dependent on the customers opinion, the web application is completed for deployment and changed according to the users comments or aborted.
- *The web site is validated by a special user group.* The same as above, except that the web application is used by a special user group, such as a group of experienced users or a test panel.
- *Comparison with competitors.* The web application is compared to the main competitors. If any shortcomings are identified, they will be fixed prior to publishing.

4 QUALITY ISSUES

We have gained some insight into how the companies in our sample handled the quality aspects of their products. This was done by asking them questions such as ”What are important quality factors?” and ”What is important for your customers?” Since the companies that we interviewed span a wide range of application areas, we want to understand the spread of opinions and their implications for the development of web systems.

The most important quality factors mentioned were availability and reliability, performance and to give the users a good user experience. By and large, the priorities given by the companies when asked about important quality factors were also reflected when the companies were asked to name important project success factors (see figure 2).

To sum up – the most important factors for a successful web system are availability, performance and the ability to give the user a good experience. This is the user view. In order to achieve this, the developers need to focus on how to build systems that are reliable, scalable and have high usability.

5 DEVELOPMENT PROCESS

In the development process, we want to look at how projects are initiated, estimated and staffed, and how these factors contribute to trade-off opportunities.

5.1 Project Organisation

All the development teams are small – three to five persons. From this it follows that most of the development projects in our sample are small. Once we know this, it is hardly a surprise that all the companies used one form or another of incremental development. The mode of development spans the whole spectrum – from just incremental to using Evo (Gilb, 2004). The latter is a complete and documented development method.

Only two of the companies say that most of its projects are initiated as a result of a request for tender. In most cases, projects are initiated either by a customer request or by product ideas stemming from cooperation between the company and a set of key customers.

When it comes to estimation, the most interesting result is that two out of six of the companies do not do any real estimation at all. Instead, they try to get an idea of what the customer is willing to pay and then they go backwards from there to define an acceptable project. Since quality factors are the requirements that are hardest to define and hardest to test, they easily becomes the factors that are adjusted in order to deliver according to calendar time and budget.

5.2 Trade-off Opportunities

Most of the companies involved had no clear trade-off strategy. One reason for this was that they did not feel the need, since they used an incremental development process. This process gives them ample time to adjust requirements and quality throughout the project and the need for explicit trade-offs is small.

One company said that they did trade-offs between all their important quality factors – performance, scalability, and newness. One company did trade-offs between price and complexity while one said that it usually adjusted the quality factors in order to finish the project within budget.

6 DISCUSSION

Our sample of companies is small and we should not generalise our findings. Still, we consider our findings as an indication that a different set of development practises will evolve when developing software in a rush-to-market and competitive environment.

An observation we made is that all companies seem to be successful. They had a clear understanding of what they perceived as their success factors and managed fairly well to live up to them. Not surprisingly were all success factors in some way related to the user view of the system.

#	Quality	Success factors	Trade-offs	Estimation and planning	Team and organization	Project initiation
1	Availability, good user experience	Usability	–	Expert judgment	Small teams	Environmental changes, customer changes
2	Performance, availability, scalability, maintenance	Scalability, performance	Performance, scalability, newness	What will the customer pay?	Small teams	Customer changes
3	Performance, reliability, user friendliness	Usability	–	Evo	Small teams, some outsourcing to Vietnam	Customer changes, small team from company and key customers
4	Performance, availability, good user experience, scalability	Development time	–	–	–	–
5	Performance, reliability, availability, good user experience	Performance, reliability	Complexity vs. price	What will the customer pay?	–	Invitation to tender, pre-project
6	Reliability	Usability, customer relevance	All quality related requirements	Expert judg-ment and WBS	–	Invitation to tender
7	Data integrity, security	Development time	–	–	–	All development is out-sourced

Figure 2: Quality issues from the interviews.

Informal and oral communication plays an important role when developing web applications in a rush-to-market environment. Most people we spoke with where satisfied with this practise. It also seems that there are few problems associated with this practise. And if used wisely it can be a considerable contribution to the success of the company. One explanation for this can be the ”piece-wise” development strategy of web applications. Developers get early feedback for their work, problems and conflicts in the requirements are detected early, and when necessary the level of detail for requirements can be increased.

The dependence on the developers experience and domain knowledge can be a challenge to this development practise. A lot of information is never documented. This makes a development team vulnerable when people leave the company.

Another challenge is to balance requirements specification and decision making. Most communication is informal and oral. How can different options be assessed when the information available is limited. A simple tool to show both the positive and the negative impact of the chosen technology would have been helpful. We have proposed such an tool in our previous work (Ziemer and Stålhane, 2004).

What surprised us was that most companies did not have a clear trade-off strategy. This does not mean that they did not perform any, but they were not aware of it. It is part of a trade-off to make a decision on how to balance two or more conflicting factors. Not being aware of this means that the decision making is haphazard and that there is no opportunity to systematic improve the result of the trade-off.

7 CONCLUSION

In this paper we have presented our findings from a series of interviews with companies developing web applications in a competitive and rush-to-market environment. Our findings suggest that development in such a environment is communication intensive. Our contribution lies in the documentation of development practises found in a small sample of companies developing web applications. We related these practises to the success criteria identified by the companies them selves and pointed out some future work directions.

In the future we will to look more at the decision making processes applied in web application development. How can good trade-offs be performed with these informal practises.

REFERENCES

Gilb, T. (1989). A planning language (a planguage). In *APL ’89: Conference proceedings on APL as a tool of thought*, pages 169–177. ACM Press.

Gilb, T. (2004). *Competitive Engineering: A Handbook for Systems and Software Engineering Management Using Planguage*. Addison Wesley Longman Publishing Co., Inc.

Ziemer, S. and Stålhane, T. (2004). The use of trade-offs in the development of web applications. In Matera, M. and Comai, S., editors, *Engineering Advanced Web Applications*. Rinton Press.