# The use of Trade-offs in the development of Web Applications

Sven Ziemer and Tor Stålhane

Department of Computer and Information Science
Norwegian University of Technology and Science
{svenz, stalhane}@idi.ntnu.no

**Abstract.** The development of Web Applications has some special characteristics, such as very short Time-to-Market, parallel development and requirements that are not prioritized. This makes it hard to address quality attributes in a systematic way, to prioritize requirements and to resolve conflicts between them. The use of trade-off methods such as QFD will help to prioritize the "right" requirements and to resolve conflicts between them.

## 1 Introduction

In recent years, the World Wide Web has become a popular platform for software application development. One of the reasons for this is the ease of software deployment. When the software has been installed or updated on a web server, it is available for all users within seconds. Today, we are using Web Applications in many domains, and are depending on the proper function of these applications. The same is true for many companies who are using their Web Application in an important part of their business.

It is therefore a growing interest in how Web Applications are developed, and in the development practices used. Especially, there is an interest in how quality attributes are addressed during the development of Web Applications ([11], [12]).

WebSys[1] is a research project whose main goal it is to develop high-quality research competence and concrete guidelines for industrial development of timely and reliable Web-based systems.

The most important factors for the WebSys project are Time-to-Market, reliability and robustness. These factors can have a contradicting influence on each other. E.g., spending time to build a more reliable and robust system will increase the Time-to-Market. Therefore, these factors have to be balanced in a controlled way, making it possible to assess both the delivery time and how reliable and robust the system will be.

This paper presents how Trade-off can be used to resolve conflicts that stem from contradicting requirements, conflicting expectations from the stakeholders,

[1] The WebSys project is founded by the Norwegian Research Council (NFR)

and from the time-to-market requirements. We also show how the Quality Function Deployment (QFD) method is a suitable method for performing trade-offs.

The rest of this paper is organized as follows: Section 2 describes Web Engineering Practices. Section 3 introduces how trade-offs can be used in Web Development, and section 4 shows how Quality Function Deployment can be used in this context. Conclusions are given in section 6.

## 2 Web Engineering Practice

Web Applications are applications that offer contents and services that can be used by a Web browser. This opens for a wide range of Web Applications. There exists several categories of Web Applications ([13], [4], [7] and [3]), organizations can have different perceptions on Web Application Development [4], and they use Web Applications for different purposes. This makes it difficult to give a description of Web Application Development that matches every Web Development project.

Based on some surveys for Web Application Development ([2], [8], [9] and [14]), and on a number of interviews conducted in the WebSys project with Norwegian companies, we will describe some development practices which are special for Web Development. This does not mean that these practices are not found in other software development projects, but practices are found in a more extreme version in Web Application Development.

### 2.1 Rush to market

Web Development projects often have a short Time-to-Market (TTM) requirement. In some cases the development cycle for Web Applications is only some weeks. The short Time-to-Market is maybe the most important characteristic of web application development. All the development practices mentioned later are related to this one. In order to meet tight TTM requirements, development teams uses both evolutionary and parallel development. There may be other reasons for the use of ad hoc development processes (e.g. differences in development culture), but the force of the rush to market is a prominent reason.

An interesting question is, what makes this TTM requirement so important. In our opinion, this lies outside the World Wide Web as such. It is rather a factor stemming from economics and marketing. It is about opportunities that can not be missed, and it is about presence in a competitive marked environment. This is happening when a company is focusing on the short term benefit only, such as accepting to deploy software with poor quality just to keep a market opportunity or to avoid that a competitor gets an advantage. It is also about reducing the economical risks. A small investment is made to implement a new function. This function will be evaluated by the internet users. If the investment pays off – meaning that this functionality gets repeated visits from Internet user – a new, and may be larger, investment can be made to improve the quality of this function.

In order to meet the TTM requirement, development teams are using some of the following practices.

- **Evolutionary development** – The Web Application can be delivered in successive releases. For each release, new functionality will be added, and existing functionality can be improved. This is sometimes called **Web Gardening** [10]. New releases of the Web Application are deployed at a fixed time interval (e.g. every second week). As a consequence of this, to keep the deadline for each release is more important than the content of a new release. Functionality that was expected to be part of one release can easily be assigned to a later release. It is accepted that software is developed without addressing quality issues in the way they should and that this software is released. When problems with quality issues are discovered, the software can be improved in one of the future releases. In this way, the current quality of software becomes less important.
- **Parallel development** – To meet the tight TTM requirements, development teams have to reduce the time for each development cycle. This is done by parallel development. In one iteration a developer can work on several releases. He can work on the implementation for the next release and on the planning for a future release.
- **Ad hoc development process** – Web projects often uses an ad hoc development process. The development teams depend on the skills of the most experienced developers in the team. The development of Web Applications lacks the rigour and systematic approach, quality control and assurance. This contributes to what some authors have called a **Web Crisis** [10].
- **Requirements elicitation** – The requirements for Web Applications are evolving all the time. Prototypes are used to communicate with the customer, and they are also used to validate and refine the requirements. The prototypes can also be released.

The development practices described above cause a number of problems and makes it difficult to use traditional methods for quality control and assurance. The competitive environment of the Web and the rush-to-market makes it difficult to keep a long-time focus. Hence, quality issues will be addressed later in the lifecycle than is the case for traditional software development. The way requirements are dealt with makes it hard to prioritize among the requirements, and using ad hoc processes results in lack of a systematic approach for Web Application engineering and a dependence on the skills and experience of individual developers. Using traditional methods for quality control takes "too much time"[14], they assume that a proper requirement specification has been produced, and that there will be both analysis and design activities. This is not the case, as web applications are developed without a proper specification of requirements, coding starts in some cases even before the requirements are stable, and there are no analysis and design activities.

What is needed are not only methods for quality control and assurance that will work with these development practices. There is also a need for a method

that will help developers to prioritize the "right" requirements and to balance between conflicting requirements. This method must fit into the Web Development practices described here and be feasible for small project teams, since many Web Applications are developed by small project teams.

## 3 Trade-offs and Web Engineering

In the WebSys project we are studying trade-off situations that can occur in a Web Development project. A trade-off situation consists of several elements. First, there are a number of requirements, which possibly are in conflict with each other. Second, we need an analysis of the advantages and disadvantages of the proposed requirements and the costs needed to meet them. Finally, a decision based on the identified advantages and disadvantages has to be made, balancing them to yield the best result possible.

One trade-off situation for Web Engineering projects is the conflict between Time-to-Market and high quality. Both requirements are important, and we have to make a trade-off between "sooner but worse" and "later but better". A web application can loose users if new functionality is first introduced on a competitors Web application. On the other hand, the only mechanism that brings users back to Web sites is high quality [11].

By using trade-offs we want to give small software projects a simple way to have an efficient decision making process. This is an important criteria for us, since we want to use methods that are well known, and easy and intuitive to understand and use.

### 3.1 The nature of Trade-offs

The use of trade-offs will be helpful for Web Development projects with respect to the aforementioned problems. Trade-offs help to create awareness of the conflicts between contradicting requirements, success factors and goals for a project. A trade-off shows the conflicts and the possible solutions that can be used to resolve them. The consequences of the solutions are also shown. Thus, using a trade-off method helps the project to find the best available solution.

This also makes a trade-off a tool for negotiation between stakeholders. The solution to the problem of conflicting requirements and interests from different stakeholders is usually one that make the most stakeholders satisfied. Using a trade-off is helpful in this kind of negotiation because all parties can see the consequences of their choices, and use this to find the proper balance.

Trade-offs can help to find a balance between the short and long term benefits in a development project. In the end the preferred balance depends on the decision-makers. The trade-off method should show were the conflict between short and long term benefits is, and how these two views can be balanced in a compromise.

Trade-offs help a project team to focus on a common goal, since the trade-off analysis described above is a group activity. The pros and cons of conflicting

requirement and solutions are discussed in the trade-off situation. Thus, all the available information is shared among the project team members, improving the communication in the development team and between all the stakeholders.

All trade-offs imply changing one or more goals in the project. In order to do this, we need to make the goal measurable – either directly or indirectly – and relate it to our experience and state of the practice. There are several ways to make the goal measurable. In our opinion, the GQM process [16] is the most efficient way to achieve this. To relate the goal to experience and state of the practice we need three pieces of information:

- What is our standard, average performance related to the goal.
- What is the best we have ever done?
- What is – to our knowledge – the best that is done worldwide?

The purpose of this exercise is to highlight the relationship between the goals and the resources and time necessary to achieve it. The message should be clear: if the trade-off process requires us to go beyond the best we have ever done, the probability of achieving this goal is small. By knowing our own limits, based on experience, we can assure that we keep our project goals realistic.

An important parameter to consider during trade-off is the degrees of freedom available at the current stage of the project. The opportunities for making trade-offs will decrease over time as we make more decisions. At the same time, our understanding of the product, the customer's need and so on increases over time. The paradox is that when we deliver the product, we have full understanding but can not change anything. There is a way out of this – we need to increase our understanding from the start so that we have some information to base the early trade-offs on. This can be general knowledge or company-specific knowledge.

### 3.2 Five questions about trade-offs

The discussion related to the five questions following below will give us a better understanding of how we want to use trade-offs in a development process.

- **When to start a trade-off?** Trade-offs can be started in two settings. Either the trade-off is scheduled or there are situational circumstances that triggers the need for a trade-off. Trade-offs can be planed for situations we know from our experience are critical for the development project. They can also be triggered when the project team runs into problems and conflicts that can not be easily resolved. This is the case when the consequences of some decision is unclear.
- **What are the prerequisites for using trade-offs successful?** A clear understanding of the conflict or problem is needed, in addition to a list of alternatives or solutions, and the consequences attached to them. This helps us to assess the conflict at hand, to assess the consequences of possible solutions and to choose among the solutions with respect to the desired outcome. It is important to look at solutions that can create a consensus.

- **How to perform a trade-off?** For scheduled trade-offs this is straight forward. A member of the project team is assigned as responsible for this trade-off and have to prepare for it (see question on prerequisites for trade-offs). In case of non-scheduled trade-offs this is more difficult. As soon as somebody involved in the project discovers a conflict that he can not resolve himself, he should prepare for a trade-off meeting and be responsible for that trade-off. The trade-off should be performed as a group activity, with the responsible team member acting as a moderator. We want the trade-off meeting to be as informal as possible, so that it can fit into the development practices.
- **What are the results of a trade-off?** A trade-off should result in a decision on what actions to take, and how to resolve the conflict at hand. An important thing about a trade-off is that the project team should have the same focus and at least a common understanding of the rational for the decision made or proposed. In some cases it may not be possible to find a decision, but rather a list of alternative decisions, with the final decision to be taken at some later time. The decision can be made by the same group when the project has gathered more knowledge about the unresolved conflict.
- **How many persons can participate in a trade-off?** There is a limit on the number of persons that can participate in trade-offs as they are described here. We believe that the upper limit is 10 persons, and that the optimal number of persons is four. The reason for keeping the number low is to avoid making trade-offs a formal event. By keeping trade-offs informal they can be used as a way to help speed up the decision process.

The objective for trade-offs in the WebSys project is to find a simple way for the projects to clarify their goal and problems, to support a constructive communication about them, to help to gather all available information about them and to reach a decision based on consensus.

## 4 The use of Quality Function Deployment (QFD)

The WebSys project has no intention of solving the trade-off problems once and for all. The problems and the spread of stakeholders competence are too diverse for that. Instead, we have put together a tool box that developers and project managers can use when they need to perform a trade-off [15]. When selecting trade-off tools for our tool box, we decided that our tools must fulfill the following requirements:

- Be easy to understand for all parties involved.
- Include both qualitative and quantitative information.
- Be applicable in all activities in a web project.
- Be applicable with all development methods and process models.

Quality Function Deployment – QFD for short – is one of the methods in our toolbox. We find it easy to learn and use, and is well suited for the trade-off

situations described above, because of its flexibility. This method is described in [1], see also [5]. The following is a description and discussion of the Quality Function Deployment method, and a discussion of how it can be used in Web Development projects.

### 4.1 Quality Function Deployment

QFD is a method for answering important questions during the requirement analysis, architectural design, technological assessment and implementation planning. We will here only consider the requirements phase, where QFD looks at the three important questions: what, how and how much.

- What – what shall we implement? This is pertaining to the customer's requirements
- How – what method, technique etc. shall we use to realize each requirement?
- How much – how much resources shall we put into meeting the requirements? This will depend on the priority of each requirement and on how good these requirements are met by our competitors.

Traditionally, the QFD matrix uses three symbols to indicate the relationships between the "whats" and the "hows" (strong correlation, weak correlation and conflicting relation). In order to use QFD in trade-off situations, we are using numerical values directly into the matrix for the following situations:

- 9 points, strong positive relation
- 3 points, weak positive relation
- 0 points, no relation
- Correspoinding for negative relations.

Each tool, method and function – the "hows" – get a score that is the sum of the products of its relationship to each requirement. All the information can be shown in a matrix (see figure 1). This method has much in common with the Multiple Attribute Evaluation method [6] used in economy.

### 4.2 QFD for Web Development

There are several factors that make the QFD well suited for trade-offs:

- The data needed – requirement priorities and our current level for these requirements compared to our most important competitor(s). By requiring that these data must be available, we have taken a major step towards a sound basis for trade-offs.
- The importance and difficulty scores, which gives us a good idea of the work needed when using each tool or method and for implementing each function.
- How we relate to our main competitors both with current and new versions of our products.

QFD matrix for method 1:

| | Requirements | Weights | | Tools, methods | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Stakeholders: A = Marketing, C = Maintainance | A | C | Method 1 | Robust Pattern | LoadTest Tool 1 | Browser Test | Total A | Total B | Us now | Goal | Product 1 | Product 2 |
| 1 | Dynamic effects | 5 | 2 | 9 | 0 | -3 | -3 | 15 | 6 | 2 | 4 | 4 | 3 |
| 2 | High Performance | 4 | 5 | -9 | 0 | 9 | 0 | 0 | 0 | 4 | 5 | 4 | 5 |
| 3 | Short TTM | 5 | 2 | -3 | 0 | -3 | -3 | -45 | -18 | 3 | 3 | 3 | 2 |
| 4 | Run on all browsers | 4 | 3 | -9 | 0 | 0 | 9 | 0 | 0 | 3 | 4 | 4 | 4 |
| 5 | Tolerate bad data everywhere | 4 | 5 | 0 | 9 | 0 | 0 | 36 | 45 | 3 | 3 | 3 | 3 |
| | Total Stakeholder A | | | -42 | 36 | 6 | 6 | 6 | | | | | |
| | Total Stakeholder B | | | -60 | 45 | 33 | 15 | | 33 | | | | |

QFD matrix for method 2:

| | Requirements | Weights | | Tools, methods | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Stakeholders: A = Marketing, C = Maintainance | A | C | Method 2 | Robust Pattern | LoadTest Tool 1 | Browser Test | Total A | Total B | Us now | Goal | Product 1 | Product 2 |
| 1 | Dynamic effects | 5 | 2 | 3 | 0 | -3 | -3 | -15 | -6 | 2 | 4 | 4 | 3 |
| 2 | High Performance | 4 | 5 | 0 | 0 | 9 | 0 | 36 | 45 | 4 | 5 | 4 | 5 |
| 3 | Short TTM | 5 | 2 | 0 | 0 | -3 | -3 | -30 | -12 | 3 | 3 | 3 | 2 |
| 4 | Run on all browsers | 4 | 3 | 0 | 0 | 0 | 9 | 36 | 27 | 3 | 4 | 4 | 4 |
| 5 | Tolerate bad data everywhere | 4 | 5 | 0 | 9 | 0 | 0 | 36 | 45 | 3 | 3 | 3 | 3 |
| | Total Stakeholder A | | | 15 | 36 | 6 | 6 | 63 | | | | | |
| | Total Stakeholder B | | | 6 | 45 | 33 | 15 | | 99 | | | | |

**Fig. 1.** QFD Matrix for method 1 (top) and QFD Matrix for method 2

– The description of how each tool, method and implemented function contributes to the delivered product. There are two important pieces of information here; we have to make sure that each line in the table have at least one marked column and the strength of the relationships the "hows".

The main importance of the information shown in the QFD matrix is that it improves the decision process by showing the stakeholders what they can manipulate and the consequences of their manipulations.

Trade-off problems in web development can often be solved by using QFD. Also, the use of QFD can support the requirement elicitation of web projects. When using QFD the way we suggest, every stakeholder can rate the importance of each requirement, which will reveal potential conflicts among the stakeholders. The tool and methods available to fulfill the requirements can be rated according to how they influence the intended requirements and how they influence other requirements. The requirements can be compared to the requirements from a competitor. The whole process will help us to exclude unrealistic requirements, assign a low priority to nice-to-have requirements and to identify the most important requirements.

When conflicts are discovered, the development team can identify advantages and disadvantages, and attach weights to each requirement. This will improve the communication inside the team, and with the stakeholders. All of this can be done without the use of formalized methods, by using ranking and weights in the QFD matrix (also called the House of Quality).

In order to show how this will work, we look at a small scenario taken from a company we conducted an interview with: The marketing director of a company wants to include some dynamic effects into a Web Application to make it more

attractive to the main user group. The maintenance group on the other hand wants to achieve a high performance and to test this in advance. Also, short TTM, and the requirement "run on all available web browsers" are important requirements for the next version of the Web Application. In this situation a couple of "conflicts" have to be resolved:

- There is a value conflict between a marketing issue (dynamic effects) and maintenance (high performance), which can be seen from the different ratings these stakeholders (marketing = stakeholder A and maintenance = stakeholder C in fig 1) associate with these requirements. By comparing the weights of these requirements with two competitors (Product 1 and Product 2), a compromise can be reached, which makes this the best product when comparing it with the competitors.
- Two different methods are available to implement the dynamic effects and there is a possible conflict on which one to use. Method 1 will result in the best dynamic effect (indicated by a "9" in the first QFD matrix), whereas Method 2 will give a not so good, but still acceptable effect (indicated with a "3" in the second QFD matrix). When analyzing the consequences of both methods with respect to the other requirements, it becomes clear that Method 1 will have a negative influence on all other requirements. In this case the development team decides to use Method 2.
- In order to save time, the development team considered removing the requirement "run on all available web browsers". This will remove the need to comply with the browser checklist, and to perform the Browser Test. However, the QFD matrix also shows us that we at the present are trailing way behind our most important competitors – Product 1 and Product 2 – in this matter. The new version was intended to bring us closer to our most important competitor – Product 1. Since the competitors are considered to be better on the "run on all browsers" requirement, but about the same level for TTM, the team decides to keep the requirement.

## 5 Future Work

So far, we have performed interviews with a number of companies to identify trade-off situations that these companies encounter during the development of Web Applications and to study how they are resolved. The next step in our work is to perform case studies with companies using QFD to perform trade-offs. In addition, we will perform student experiments in a software engineering course. The goal of the experiments will be to see if student groups using QFD to perform trade-offs can make more informed decision given the short amount of time available.

## 6 Conclusions

The development of web application is different from traditional software development. Among the most important differences is the strong Time-to-Marked

requirement, which makes it difficult to address quality attributes in a proper and systematic way. The development practices identified for Web development result in problems such as the lack of requirement prioritization, no long-time focus and no systematic development approach with respect to quality attributes.

In the WebSys project we have looked at the decision making process with regard to functional and non-functional requirements, and how conflicts between them can be resolved by using trade-off methods. The method we are proposing is Quality Function Deployment.

We find the use of trade-offs helpful for several reasons: trade-offs can create awareness about conflicting requirements, they can help to negotiate how to resolve conflicts, and they can help us to reach a consensus on the projects goal.

## References


1. Y. Akao. *Quality Function Deployment*. Productivity Press, Cambridge, MA, 1990.
2. L. Baresi, S. Morasca, and P. Paolini. An empirical study on the design effort of web applications. In *Proceedings of the 3rd International conference on Web Information Systems Engineering (WISE 2002)*, 2002.
3. J. Conallen. *Building Web Applications with UML, Second Edition*. Addison-Wesley, 2003.
4. Y. Deshpande, S. Murugesan, A. Ginige, S. Hansen, D. Schwabe, M. Gaedke, and B. White. Web engineering. *Journal of Web Engineering*, 1(1):3–17, October 2002.
5. R. B. Grady. *Practial Software Metrics for Project Management and Process Improvement*. Prentice Hall PTR, 1992.
6. C.-L. Hwang and K. Yoon. *Multiple Attribute Decision Making – Methods and Applications*. Springer Verlag, Berlin, Heidelberg, 1981.
7. G. Kappel, B. Pröll, S. Reich, and W. Retschitzegger, editors. *Web Engineering*, chapter Web Engineering - Die Disciplin zur systematischen Entwicklung von Web-Anwendungen. dpunkt.verlag, 2004.
8. A. McDonald and R. Welland. A survey of web engineering in practice. Technical Report TR-2001-79, Departmend of Computing Science, University of Glasgow, 2001.
9. A. McDonlad and R. Welland. Web engineering in practice. In *Proccedings of the fourth WWW10 Workshop on WebEngineering*, pages 21–30, 2001.
10. S. Murugesan, Y. Deshpande, S. Hansen, and A. Ginige. A new discipline for web-based system development. In *Proceedings of the Workshop on Web Engineering, ICSE99*, 1999.
11. J. Offutt. Quality attributes of web software applications. *IEEE Software*, 19(2):25–32, 2002.
12. L. Olsina, G. Lafuente, and O. Pastor. Toward a reusable repository for web metrics. *Journal of Web Engineering*, 1(1):61–73, 2002.
13. R. Pressman. *Software Engineering – A Practitioner's Approach*, chapter 29. McGrawHill, 2000.
14. B. Ramesh, J. Pries-Heje, and R. Baskerville. Internet software engineering: A different class of processes. *Annals of Software Engineering*, 14(1-4):169–195, 2002.
15. T. Stålhane and S. Ziemer. A trade-off toolbox. Technical report, Department of Computer and Information Science, 2003.
16. R. van Solingen and E. Berghout. *The Goal/Question/Metric Method*. McGraw-Hill, 1999.