Quality Planning for E-Commerce Applications

Prof. Dr. GEORG HERZWURM
Professur für Wirtschaftsinformatik, insbesondere Informationssysteme im Dienstleistungsbereich, Technische Universität
Dresden, Bundesrepublik Deutschland

SIXTEN SCHÖCHERT
Department of information systems, system development, University of Cologne, Germany

ULRIKE DOWIE
Infineon Technologies AG, IT FIN SAP A, Munich, Germany

ABSTRACT

E-commerce (electronic commerce) as understood in this paper is the buying and selling of goods and services on the
Internet, especially the World Wide Web. Planning e-commerce applications leads to new challenges for requirements
engineering methods. The domain is relatively new to both developers and users. The technology is rapidly changing. Time
and creativity are important success factors. This paper presents a special variant of Quality Function Deployment (QFD) for
building e-commerce applications that considers these conditions. For this, demands on the design of a QFD project are
derived on the base of general characteristics of e-commerce applications. Following we explain the main principles of a
special QFD variant for e-commerce, called Continuous QFD (CQFD). Finally, we evaluate the suitability of CQFD for
planning e-commerce applications in a small case study.

1 INTRODUCTION

This paper investigates in what way Quality Function Deployment (QFD) is suitable for planning e-commerce (electronic
commerce) applications. E-commerce as understood in this paper is the buying and selling of goods and services on the
Internet, especially the World Wide Web (for other and/or detailed definitions see [1]). The focus of the paper lies in the early
development stages of a software project i.e. eliciting and defining requirements. For this, demands on the design of a QFD
project are derived on the base of general characteristics of e-commerce applications. Following we explain the main
principles of a special QFD variant for e-commerce, called Continuous QFD (CQFD). Finally, we evaluate the suitability of
CQFD for planning e-commerce applications in a small case study.

Existing Requirements Engineering methods pursue the primary goal to transform user requirements that exist in a natural
language into formal specifications to serve as basis for design and implementation. The most techniques used are primarily
gear towards building data, function, and process models and thus focus mainly on technical aspects of the information
systems to be developed. Customers who are not familiar with these models created for specialists and don’t want to invest
the necessary high learning efforts, however, only comprehend such models to an unsatisfactory degree which leads to
constant communication problems with software developers. QFD provides a systematic but more informal way of
communication between customers and developers. Additionally QFD is aimed at software that presents not all technically

possible characteristics but those that customers demand (“fitness for use”). The entire QFD process is carried out by a cross-functional team, including customer representatives, and an experienced moderator [2].

2 QFD BASICS
To avoid that (presumably) brilliant and (supposedly highly) complicated technical solutions miss the market’s needs QFD strictly distinguishes between requirements and solutions. This means not only separating “WHAT is to be realized” from “HOW is it to be realized” technically. QFD already starts one level earlier, namely with the distinction between WHAT and the reasons “WHY something is to be realized”. Only the answers to this WHY describe the customer benefits in the form of advantages when using the product. This opens up potential for new ideas and thus paves the way for alternative, innovative solutions.

2.1 Important Definitions in QFD
The distinction between “WHY” and “WHAT” manifests itself in the basic definitions of customer requirement and product characteristic, where the demanded implementation independence of the latter describes the separation of “WHAT” from “HOW” (e.g. in the form of architecture components). Besides, in QFD one has to distinguish between functional characteristics (product functions) and non-functional characteristics (quality elements) of a software product. Tab. 1 summarizes the most important definitions in QFD.

<table>
<thead>
<tr>
<th>Definition</th>
<th>Need resulting from using the product: Business need</th>
<th>Characteristics or capabilities of the product, independent of implementation, which in case of high fulfillment give the customers the advantages their requirements imply</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression</td>
<td>Customer requirement</td>
<td>Product function</td>
</tr>
<tr>
<td>Definition</td>
<td>Brief, concise statements put in the customers’ words, about advantages which customers get or could get from using the product</td>
<td>Functional characteristic feature of the product, usually not measurable (creates perceptible output)</td>
</tr>
<tr>
<td>Example for a email client</td>
<td>Write emails fast and easily</td>
<td>Enter email text via voice</td>
</tr>
</tbody>
</table>

Tab. 1: Important definitions in QFD [2].

2.2 The House of Quality (HoQ)
The best known instrument of QFD is the so-called House of Quality (HoQ). The HoQ is a matrix which analyzes customer requirements in detail and translates them into the developers’ language. Put in reverse order: product characteristics are being prioritized to form development targets by means of correlating them to weighted customer requirements.
The HoQ is the framework of most of the prioritization matrices used in QFD. Since QFD has its roots in the manufacturing industry, the product characteristics in the HoQ originally correspond to measurable quality characteristics. But the product software is identified not by its physical characteristics but by its behavior. Put differently: “Software [...] is valued not for what it is, but for what it does” [3]. So in Software QFD the quality elements used in the classic HoQ in manufacturing are replaced by product functions. Fig. 1 gives an example of a Software HoQ for an email client software [2].

<table>
<thead>
<tr>
<th>Customer Requirements / Weights</th>
<th>Product Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write emails fast/easily</td>
<td>7,29</td>
</tr>
<tr>
<td>Write emails fast to many users</td>
<td>3,94</td>
</tr>
<tr>
<td>Have overview of incoming mails</td>
<td>3,56</td>
</tr>
<tr>
<td>Design mailbox individually</td>
<td>3,43</td>
</tr>
<tr>
<td>Keep using existing address data</td>
<td>4,86</td>
</tr>
<tr>
<td>Find read mails fast</td>
<td>4,84</td>
</tr>
<tr>
<td>Reply to mails easily</td>
<td>4,38</td>
</tr>
<tr>
<td>Manage other users email addresses</td>
<td>4,25</td>
</tr>
<tr>
<td>Continue reading existing mail</td>
<td>3,17</td>
</tr>
<tr>
<td>Have overview of read mails</td>
<td>3,17</td>
</tr>
<tr>
<td>Send mails of as little size as possible</td>
<td>4,05</td>
</tr>
<tr>
<td>Delete read mails fast and purposely</td>
<td>4</td>
</tr>
<tr>
<td>Use well-known editor</td>
<td>3,74</td>
</tr>
<tr>
<td>Move addresses between email clients</td>
<td>3,56</td>
</tr>
<tr>
<td>Access to different mails from one computer</td>
<td>3,27</td>
</tr>
<tr>
<td>Write emails with correct spelling</td>
<td>3,11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Absolute Importance</th>
<th>Relative Importance</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>77</td>
<td>105</td>
<td>97</td>
</tr>
<tr>
<td>86</td>
<td>73</td>
<td>94</td>
</tr>
<tr>
<td>130</td>
<td>106</td>
<td>93</td>
</tr>
<tr>
<td>42</td>
<td>75</td>
<td>68</td>
</tr>
<tr>
<td>75</td>
<td>39</td>
<td>58</td>
</tr>
<tr>
<td>98</td>
<td>81</td>
<td>91</td>
</tr>
<tr>
<td>119</td>
<td>39</td>
<td>115</td>
</tr>
<tr>
<td>79</td>
<td>36</td>
<td>110</td>
</tr>
<tr>
<td>114</td>
<td>58</td>
<td>102</td>
</tr>
<tr>
<td>112</td>
<td>96</td>
<td>113</td>
</tr>
<tr>
<td>21</td>
<td>65</td>
<td>89</td>
</tr>
<tr>
<td>80</td>
<td>35</td>
<td>48</td>
</tr>
<tr>
<td>98</td>
<td>22</td>
<td>39</td>
</tr>
</tbody>
</table>

Fig. 1: Excerpt from the Software HoQ exemplified by an email client [2].

The numbers 0, 1, 3 and 9 in the cells represent the degree of correlation between the satisfaction of a customer requirement (lines) by a product function (columns). The importance of an individual product function is given by the sum of the multiplications of each requirement’s weightiness and the corresponding degree of correlation. Moreover, the present data allow for numerous other quantitative and qualitative analyses. For example, a consistency analysis can be carried out: blank lines (i.e. a customer requirement without correlation) indicate that product functions are missing respectively have been overlooked, and blank columns (i.e. a product function without correlation) hint at the possibility that an unnecessary product function has been defined.

2.3 The Sequence of Matrices

The instrument used to carry the prioritized information from the matrices used in QFD through all of the development process, is deployment in the form of several matrices linked with respect to vertical output and horizontal input. This means that the columns in one matrix become the next matrix’ rows in order to then be correlated again to more detailed information in the columns which then will serve as the next matrix’ row input and so on (fig. 2).
There exists no standardized matrix chain for QFD in software development. For example in Zultner's QFD approach the product functions that have been prioritized in the HoQ are then turned into entities, processes or objects, depending on the development techniques used (fig. 3).

In an analysis of 25 software development projects in five companies (Digital Equipment, AT&T, Hewlett-Packard, Texas Instruments, IBM, CSK) by the University of Texas at Arlington requirements engineering with QFD is rated better regarding all 12 criteria used than requirements engineering with “traditional” methods. One of the reasons for success is indicated to be better communication among the development team as well as between customers and developers and better fulfillment of customer expectations [6].

However, sequential QFD [5] is not suitable for all kinds of information systems. When - as in e-commerce applications - customer requirements are not well-defined and technologies are changing fast waterfall QFD is not appropriate. In the next chapter we will present a special variant of the QFD method that is adapted to the needs of an e-commerce application.
CONTINUOUS QFD FOR E-COMMERCE APPLICATIONS

In this chapter we describe characteristics of requirements engineering tasks regarding e-commerce application development and draw consequences for the design of a QFD project.

3.1 Characteristics of E-Commerce Applications and Consequences for QFD

Developing e-commerce applications lead to new challenges for requirements engineering methods [7]. The domain is relatively new. The technology is rapidly changing. Time and creativity are important success factors [8]. The tasks are less clear because of the unstable environment and the newness of requirements and technical solutions for both developers and customers. We call this kind of development tasks “fuzzy”.

There are several different factors reducing the clarity of a development task [9]:

- Unclear Customer Requirements and Product Characteristics

  When customers are faced with a completely novel product to be developed, they are likely to have little idea of what benefits the product can deliver to them. As an example take the shopping cart that is used in e-commerce: something ordinary and well-known today at its premiere as a web component it meant a completely new solution. Or the statements customers give don’t reveal directly the implied need a product is asked to meet. Product characteristics sought to meet new requirements depend on the developers’ imagination and creativity. Even if an innovation has been found and developed, the optimal extent from the customers’ perspective to which the product function should be realized (e.g., its degree of complexity or range of functional options) may remain unclear.

  In QFD terms this means that neither all matrix input nor certain correlation values can be determined at the start of the project.

- Dynamic Customer Requirements and Product Characteristics

  Dynamic customer requirements are the logical consequence of unclear requirements: once a requirement becomes clear, new requirements will emerge and enlarge the list of existing (well-defined) requirements and shift the existing requirements’ relative weights. For product characteristics, the dynamics result from technological progress, particularly in Internet technology, which continually increases the scope of possible solutions and thus renders new functionalities suited best to meet certain requirements.

  In QFD terms this means that matrix input, related data and correlation values may change during planning and development.

- Uncertain Product Characteristics (Feasibility)

  Several categories of risk are implied when developing new product features. First, their realization may turn out too complicated or not to work at all. Second, they may not be able to be implemented as intended (due to interaction with other components involved in providing certain functionality). And third, once a function is implemented it still requires customers to accept and use it in order to meet the requirement it was developed to fulfill. A practical example of an uncertain technological solution is given by mobile phones’ connection to the internet by means of the Wireless Application Protocol (WAP). At the time the first mobile phone was presented that offered this feature it could not be
used by customers because the technology was not accepted as industry standard and therefore no content providers were available.

In QFD terms the degree of difficulty of realizing product characteristics is not determinable and correlation values may drop to zero unexpectedly, demanding new product characteristics to meet the affected requirements in very short time.

- **Time pressure**

On the Internet, development time is a success-critical factor [7]. A competitive edge requires recognizing and meeting new requirements or presenting a new technological solution first. In these markets not even high quality can compensate for developing too long.

In QFD terms this means that planning cannot be optimized. The planned product has to be of “good enough” quality [10].

### 3.2 CQFD Basic Elements

There are three basic elements of CQFD in order to overcome with fuzzy development tasks:

- **Incremental planning and implementation cycles** make product characteristics feasibly and help to make the requirements more clearly for the customer.
- **Employing Information Technology (IT)** is very important for handling the dynamic, particularly the high number of changes regarding requirements and technical solutions.
- **The use of templates**, containing prepared standard requirements, solutions, etc., and representing a tailored process for e-commerce planning, accelerates the development process and ensures quality.

### 3.3 Incremental Planning and Implementation Cycles

Maintaining a high level of customer satisfaction when customer requirements change after product delivery or improved technologies become available implies continuous adaptation of the product. Therefore, planning with CQFD does not end when a specification document is written or a first version of the product is delivered. Instead, planning carries on as long as the product is in the market. Adaptation to changing requirements and/or product characteristics at any time means that all elements in the prioritization matrices may change during the planning process. These changes are an explicit component of the method and represent no exceptions.

To integrate these changes continuously CQFD dismisses the “do it right the first time” approach. Instead, CQFD proceeds in many iterations. This has two implications: on one hand, decisions taken in an earlier meeting may be subjected to discussion again at a later point, revised and changed. On the other hand the prioritization matrices are being developed incrementally with each iteration resp. each meeting.

There is great number of short meetings following the same procedure, consisting of 5 steps: brainstorming, understanding, sorting/classifying, checking/evaluating, and deciding. The basic difference between meetings is the degree of detail of discussions and (provisional) results. New matrix inputs may occur in any meeting, and evaluation of requirement weights and correlation values becomes more refined with each meeting.
Customer requirements and product characteristics are being collected simultaneously, which is much more appropriate in an unstable environment than the sequential procedure traditional QFD follows. Independently of the degree of detail that the planning has reached at a certain point the focus lies at all times on the most important customer requirements and the product characteristics that are strongly correlated. This is essential when planning is deadline-based and feasibility of product characteristics is uncertain.

In order to carry out the principle of continuous adaptation to changed customer requirements and development conditions into practice planning and development take place simultaneously. Thus, at any time a product is available which provides at least the most important functions. Moreover, early customer feedback can be gathered in order to influence and possibly direct the further development.

3.4 Employment of Information Technologies
As opposed to traditional QFD, CQFD makes extensive use of IT. The Internet makes asynchronous, distributed, economical and structured generation and processing of information possible, independent of the actual meeting's time and its location. To prepare the meetings, the moderator will structure and visualize the gathered information so that discussion and evaluation by the entire team can take an efficient course. Special QFD software tools are being employed before, during and after the meetings for documentation, for weighting and entering correlation values by the team members, and for calculation. The benefits are that the process is continually being documented and that via the Internet this documentation can be read and worked on by all CQFD team members at any time. [11]

3.5 Use of Templates
In CQFD planning is based on a template corresponding to the individual development situation, here e-commerce applications. Templates include standard content items like customer requirements, product characteristics, web components and technology packages. The templates accelerate the process and make the fuzzy planning information more concrete.

Fig. 4: Template for an e-commerce application
Development of an e-commerce application requires three planning steps (fig. 4), each represented by a prioritization matrix: concept planning, component planning and technology planning. Of the three planning steps only concept planning is to be carried out by developers and customers, the remaining two parts are the developers’ responsibility.

The lists the template provides (represented by light gray boxes in fig. 4) contain standard values for customer requirements, product characteristics, web components and technology packages. For each CQFD project these lists have to be adapted by entering new items, changing or completing existing items. For example in consumer oriented e-applications usability characteristics such as color, video, sound, placement, feel, etc. are critical quality elements. The list entries are then completed by related data such as weights, satisfaction values, satisfaction comparison with competing products, etc.

The lists of customer requirements and product characteristics then form the input rows and columns, respectively, for the HoQ called concept planning (fig. 5). The correlations filling this house are to be determined by the team as a whole, a task that is done in various iterations both via the Internet and within the meetings.

![Fig. 5: Concept planning template (excerpt)](image-url)
Component planning, then, sets particularly prioritized product functions as row input against web components which form the columns. By web component we understand re-usable parts of a web application (for example guest book, search engine, links, pop-up help, pull-down menu).

In the third prioritization matrix, called technology planning, especially quality elements are correlated to technology packages in order to determine the set of technologies which best enables the developers to realize the sort and level of quality required by the customers (determined in concept planning). These packages each include a combination of server operating system (e.g., Microsoft NT Server, UNIX Servers such as Solaris or Linux), a data base application (such as MySQL, Microsoft SQL Server, etc.), a kind of server software (e.g., Apache Server, Internet Information Server, etc.), and a script/programming language (such as PHP, Perl, C++, Java-Script, ASP). These sets don’t consist of completely exclusive technologies in each category, but they are all different to some extent, and they present sensible alternatives.

The following tab. 2 recapitulates the main differences between CQFD and traditional QFD.

<table>
<thead>
<tr>
<th></th>
<th>Traditional QFD</th>
<th>Continuous QFD</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Process</strong></td>
<td>sequential</td>
<td>repetitive</td>
</tr>
<tr>
<td><strong>Implementation</strong></td>
<td>after finishing planning activities (planning determines implementation)</td>
<td>parallel planning and implementation (implementation supports also planning)</td>
</tr>
<tr>
<td><strong>Results</strong></td>
<td>completed milestones</td>
<td>incremental provisional results</td>
</tr>
<tr>
<td><strong>Planning</strong></td>
<td>activity oriented</td>
<td>time oriented</td>
</tr>
<tr>
<td><strong>Time horizon</strong></td>
<td>defined end</td>
<td>continuous</td>
</tr>
<tr>
<td><strong>Changes</strong></td>
<td>exceptions (to be avoided)</td>
<td>standard (adaptation intended)</td>
</tr>
<tr>
<td><strong>Extent</strong></td>
<td>completeness desirably</td>
<td>focus on essential</td>
</tr>
<tr>
<td><strong>Meetings</strong></td>
<td>few, long meetings</td>
<td>many, short meetings</td>
</tr>
<tr>
<td><strong>Templates</strong></td>
<td>for matrix chains only</td>
<td>for matrix chains and content of matrices</td>
</tr>
<tr>
<td><strong>IT-Support</strong></td>
<td>documentation</td>
<td>documentation and communication (virtual teams)</td>
</tr>
</tbody>
</table>

Tab. 2: Main differences between traditional QFD and CQFD

3.6 Life of a CQFD project

Product planning is continued in each meeting with brainstorming for new requirements and/or product functions because the product will have to be adapted to new or changed requirements and technologies continually. The next meeting can pertain to the same release and follow soon, or it forms part of the next release’s planning and thus won’t be held until a few months later. Meetings are set up in time before delivery of a new release in order to include all changes (regarding design and functionality) that have become necessary since the last release was delivered. New versions are either developed at regular intervals (following a release plan) or in time to meet a certain “milestone”, i.e. an event where the product is to be presented. A new set of meetings can also originate in the necessity for adaptation which results from changed values in the matrices.

Independently of the release planning, the team members keep the templates that are posted in the Internet/Intranet up to date.
The main source for adaptation are customer reactions once the product or new release hits the market. Consequences affecting the planning input data can be derived from detailed sales information that also include customer data, from customer complaints or other satisfaction indicators, from actions and/or reactions by the competition etc. The templates are adjusted accordingly so that they can serve as basis for the planning of the next release.

The idea behind this continuous approach (fig. 6) that won’t stop adapting the product to customer requirements in between planned releases is to be able to deliver a product at all times which fulfills the identified customer requirements as well as possible under the given development conditions and time restriction.

![Fig. 6: Lifecycle of a CQFD project (according to Boehm's spiral model [12])]({})

4 EVALUATION OF CQFD

The CQFD approach was tested in an e-commerce application case study. The project’s target from the initiator's point of view was to plan and implement a website for a computer dealer whose primary customer group are administrative bodies such as city councils or universities and/or individual departments of these. The team consisted of 3 customer representatives, 3 developers, a sales employee as quality and product manager and at the same time representing customers whose interests were different from the participating customers’, and the company owner as project manager. Two external moderators prepared, conducted and documented the meetings.

After the project CQFD team members were asked whether their expectations on the project and the method used for planning had been fulfilled. All participants agreed that the results including the respective demanded degree of fulfillment, proved planning with CQFD successful in this case study (see fig. 7 for details).
Regarding two issues the expectations of the CQFD team members had not been fulfilled: The Internet based communication and testing platform caused technical troubles for some customer representatives so that an additional plug-in was required to overcome Java compatibility problems. The time effort between the CQFD meetings was underestimate by nearly all team members.

To evaluate CQFD from a methodological point of view we confront the demands derived from the characteristics of e-commerce application (see chap. 3) with the practical effects of the CQFD elements recognized in the case study.

For this we use a matrix similar to the HoQ notation (fig. 8).

---

**Fig. 7: Fulfillment of team members expectations**

**Fig. 8: Evaluation of CQFD**
Proceeding in iterations allowed for new items to be included in the prioritization matrices and thus change priorities among development targets at any point. Besides, this provided team members with enough time for making up their minds about the requirements’ refined weights so that these could be assumed to be very reliable.

The entire team was satisfied that planning due to IT support and the use of templates had been completed on schedule. Thanks to simultaneous planning and presentation of weekly builds (weekly implemented pre-products) everyone was aware of the current status of the product and customer representatives had a chance to give their feedback, which also served as guideline to focus further development. The weekly builds enhanced also visibility and feasibility of the product characteristics. The uncleness of customer requirements and product characteristics was reduced not only by the weekly builds but also by use of templates and incrementally developed matrices.

Of course, not all of our experiences in this project have been positive: Because of the early realization activities some design and/or implementation decisions (e.g. technology packages) had to be taken which made reacting to changing customer requirements and product characteristics more difficult (see symbol in fig. 8). The most valuable criteria for a CQFD assessment are the economical success of the web site and the user’s satisfaction. However, due to business policy reasons, the web site isn’t online yet. Therefore no web metrics or comments from users of the site are available for judging the results.

5 CONCLUSION
To conclude, the case study presented above shows that Continuous QFD is capable of achieving requirements engineering results as postulated by the model’s theoretical background by reacting appropriately to the difficulties fuzzy development tasks bring about.

Due to the small number of enterprises who have experiences with using QFD for e-commerce applications in no phase of our work a general validity of the test results was raised. However, even if the results are not representative in terms of statistics, it becomes clear that QFD offers a chance for a fundamental advance towards more customer-oriented e-commerce development.
REFERENCES


