REQUIREMENTS ENGINEERING FOR MOBILE-COMMERCE APPLICATIONS

Georg Herzwurm
Dresden University of Technology; georg@herzwurm.de

Sixten Schockert
University of Cologne; sixten.schockert@uni-koeln.de

Ulrike Dowie
Infineon Technologies AG; ulrike.dowie@infineon.com

Michael Breidung
Dresden University of Technology; michael.breidung@mailbox.tu-dresden.de

ABSTRACT
M-commerce (mobile commerce) as understood in this paper is the buying and selling of goods and services on a mobile platform. Planning m-commerce applications leads to new challenges for requirements engineering methods, even against the background of UMTS. The domain is completely 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 m-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 m-commerce applications. Following we explain the main principles of a special QFD variant for m-commerce, called Continuous QFD (CQFD). Finally, we evaluate the suitability of CQFD for planning m-commerce applications.

1 INTRODUCTION

This paper investigates in what way Quality Function Deployment (QFD) is suitable for planning m-commerce (mobile commerce) applications. M-commerce as understood in this paper is the buying and selling of goods and services on a mobile platform (for other and/or detailed definitions see [9]). 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 m-commerce applications. Following we explain the main principles of a special QFD variant for m-commerce, called Continuous QFD (CQFD). Finally, we evaluate the suitability of CQFD.

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 geared 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. (For a synopsis of an extract of the most common process models see [11]) QFD provides a systematic but more

informal way of communication between customers and developers. Additionally QFD is aimed at a software that presents not all technically possible characteristics but those that customers demand ("fitness for use"). Compared by methodical aspects, QFD comes up as a totally new basic approach of requirements engineering regarding to the "traditional" methods. The entire QFD process is carried out by a cross-functional team, including customer representatives, and an experienced moderator [6].

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. Even, the paper is focussing on the very early stages of a m-commerce project, the principle of distinguishing between requirements and solutions could be enlisted to all stages of the development process, since one could interpret the results of one stage as the requirements of the following stage, and so on. This point should become more clearly in sections "2.3 The sequence of matrices" and "3.6 Life of a CQFD project".

2.1 Important Definitions in QFD

The distinction between "WHY" and "WHAT" manifests itself in the basic definitions of customer requirement and product characteristic, while 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. Table 1 summarizes the most important definitions in QFD.

	Customer need = demand	**Product characteristic = solution** (product/system requirement resp. Specification)	
Definition	Need resulting from using the product: Business need	Characteristics or capabilities of the product, independent of implementation, which in case of high fulfillment give the customers the advantages their requirements imply	
Expression	Customer requirement	Product function	Quality element
Definition	Brief, concise statements put in the customers´ words, about advantages which customers get or could get from using the product	Functional characteristic feature of the product, usually not measurable (creates perceptible output)	Non-functional characteristic feature of the product, possibly measurable during development and before delivery (does not create perceptible output)
Example for a email client	Write emails fast and easily	Enter email text via voice	Short response time

Table. 1: Important definitions in QFD. [6]

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" [13]. So in Software QFD the quality elements used in the classic HoQ in manufacturing are replaced by product functions. Figure 1 gives an example of a Software HoQ for an email client software [6].

Customer Requirements / Weights		Product Functions	In reply include original text for comments	Enter email text via voice	Format email text differently	Add in graphical objects to mail text	Spell check	Sort addresses in address book	Create distribution lists from addresses	Individual configuration of ad-dress imports	Create personal address book	Individual signatures	Manage various users	Manage different email addresses for user	Compress files for attachment	Data import/export internal-external editor	Integrate external editor	Attach different types of files	Support standard mail protocols	Save/group mails in different files	Structure mail files	Filter mails into files according to criteria	Distribute incoming mails to files by criteria	Sort read mails according to criteria	Optional file (garbage) for deleted mails	Move mails between files by criteria	Delete emails according to criteria	Reject mails from certain senders
Write emails fast/easily	7,28		3	9	1	3	3	3	1	1	3	3	1	3	3	3	3	1	1									
Write emails fast to many users	5,94		1	9	1	3	3	3	9	1	3	3	1	3	3	3	3	1	1									
Have overview of incoming mails	5,58																			3	3	9	9					9
Design mailbody individually	5,43		1	1	9	9						9				3	3	9										
Keep using existing address data	4,86									9	9					1												
Find read mails fast	4,64																			9	3	9	3	9	1	1		1
Reply to mails easily	4,38		9	3	1	1					1	1				1		1										
Manage other users' email addresses	4,25							9	3	1	9			1														
Continue reading existing mail	4,17																			9	3	9	3	9	1	9	3	
Have overview of read mails	4,17															3	3		3									
Send mails of as little size as possible	4,05		1		9										9			3										
Delete read mails fast and purposely	4																			1		3	1	3	3	3	9	3
Use well-known editor	3,74			1	1	1										1	9											
Move addresses between email clients	3,56									9	1					1												
Dismiss junk mail (e. g. advertising mails)	3,51																			3	3	3	9	1		3	9	9
Access to different mails from one computer	3,27										3	3	9	9					3	1								
Write emails with correct spelling	3,1			9			9					1				1	3											
	Absolute Importance		77	169	107	97	68	78	73	93	139	106	43	73	76	88	111	79	36	114	54	152	112	95	21	65	80	98
	Relative Importance		3,3	7,3	4,6	4,2	2,9	3,4	3,2	4	6,1	4,6	1,9	3,2	3,3	3,8	4,8	3,4	1,5	4,9	2,3	6,6	4,9	4,1	0,9	2,8	3,5	4,3
	Ranking		17	**1**	7	10	21	16	19	12	**3**	8	24	20	18	13	6	15	25	**4**	23	**2**	**5**	11	26	22	14	9

Figure 1: Excerpt from the Software HoQ exemplified by an email client [6]

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 analysis. 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

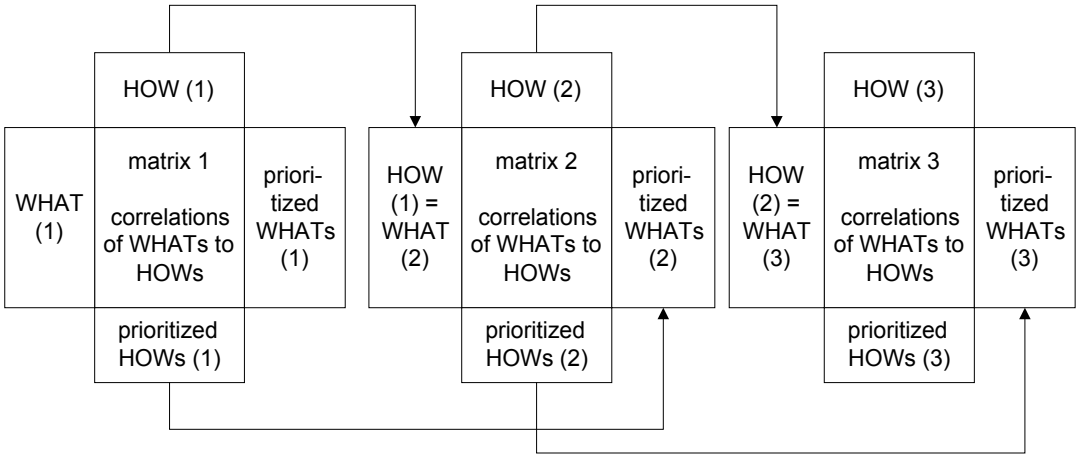

Figure 2: Illustration of a matrix sequence [3]

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 (figure 3).

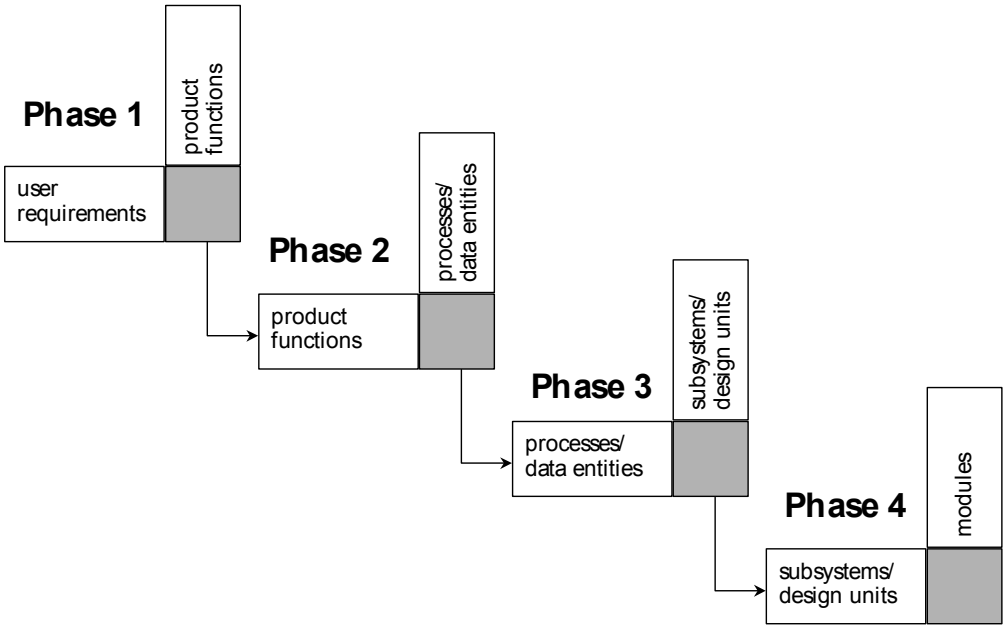

Figure. 3: Zultner's QFD approach [14]

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 fulfilment of customer expectations [5].

However, sequential QFD [14] is not suitable for all kinds of information systems. When - as in m-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 a m-commerce application.

3 CONTINUOUS QFD FOR M-COMMERCE APPLICATIONS

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

3.1 Characteristics of M-Commerce Applications and Consequences for QFD

Developing m-commerce applications lead to new challenges for requirements engineering methods. The domain is relatively new. The technology is rapidly changing. Time and creativity are important success factors [4]. 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”.

Kemper and Wolf characterising the scope of m-commerce application by the three dimensions risk, degree of innovation and speed of development with the values low and high (The values have to be checked situational for the particular systems). Combining these dimensions, one get eight different types of m-commerce applications [8]. All single types of m-commerce applications have there own requirements to the project environment. While the Type 1 have low values in all of the three dimension, it has the least requirements, but those m-commerce projects are rather not to expect. In the opposite, projects with high values in all dimensions. In m-commerce projects, high levels of each of the three dimensions (Type 8) are usually reflected in the several problem fields. The different problem fields reducing the clarity of a development task [9] and leading to the technical and organisational requirements of the project environment. After explaining the problem fields, the paper will show how Continuous QFD is able to handle these problems by considering those requirements.

Unclear Customer Requirements and Product Characteristics (caused by high innovation and risk)

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 short message service (SMS) that is used in m-commerce: something ordinary and well-known today at its premiere as a mobile solution it meant a completely new solution. An other example might be location based services. 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 (caused by high innovation and risk)

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 mobile 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 (caused by high innovation and risk)

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 a 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 customers could not use it 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 (high development speed)

In the mobile world, 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 even high quality can hardly compensate for developing too long.

In QFD terms this means that planning cannot be optimised. The planned product has to be of "good enough" quality [12].

3.2 CQFD Basic Elements

There are three basic elements of CQFD in order to overcome the identified problem fields:

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 m-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 prioritisation 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 iteration loops. 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 prioritisation 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, 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

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 [7]. With EASY-QFD, a non-commercial free web based software tool supporting QFD (http://www.qfdid.de/werkzeuge/easyqfd/easyqfd.htm), QFD teams can collaborate independent from location and time.

Figure 4: non-commercial, web based software tool: EASY-QFD

3.5 Use of Templates

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

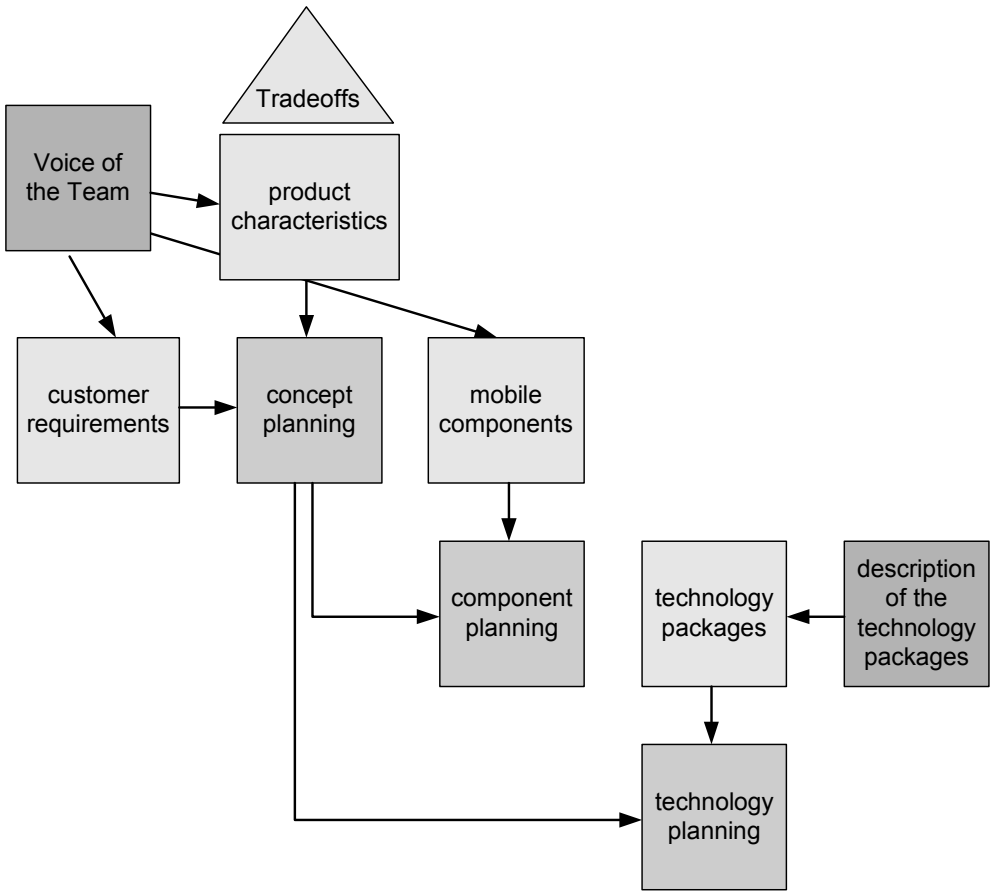

Figure 5: Template for an m-commerce application

Development of an m-commerce application requires three planning steps (figure 5), each represented by a prioritisation 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 grey boxes in figure 5) contain standard values for customer requirements, product characteristics, mobile 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 m-applications usability characteristics such as colour, video, sound, placement, feel, etc. will be 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 (Table 2 shows an excerpt of the concept planning). 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.

customer requirements	*product characteristics*																	
	financial security information	objectives, e.g. price	subjectives, e.g. analyst comments	**searching on the site**	local search	global search engine	**oder book**	**manage my offers, requests, order**	**watch list**	**virtual deposit facility**	**information services**	business information	information about the site owner	new issues	.....	**quality elements**	security	.....
support of information phase																		
information about financial securities		9	9		9	9	3		9	3		9		9				
latest informations about novelties		9	9		9	9	3		9	3		9		9				
support of evaluation phase																		
easy comparision of financial securities		9	9			9		3	3			9		1				
easy comparision of buying conditions		9				9		3				3						
support of buying phase																		
requesting individual orders					9	9	9	9		3			9	3			9	
processing orders							9	9		3			9				9	
support of after sales phase																		
call on service													9					
support of after communication																		
communicate with partners		9	3										9				9	
provide security																		
guarantee confidentiality																	9	
guarantee integrity																	9	
guarantee autenticity and binding nature																	9	
.....																		

Table 2: concept planning template (excerpt)

Component planning, then, sets particularly prioritised product functions as row input against mobile components that form the columns. By mobile component we understand re-usable parts of a mobile application (for example search engine, links, help menu).

In the third prioritisation 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 operating system (e.g., Microsoft Ce, PALM OS, EPOC) and a describing language (such as WML, WBXML). 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 table 3 recapitulate the main differences between CQFD and traditional QFD.

	Traditional QFD	**Continuous QFD**
Process	sequential	repetitive
Implementation	after finishing planning activities (planning determines implementation)	parallel planning and implementation (implementation supports also planning)
Results	completed milestones	incremental provisional results
Planning	activity oriented	time oriented
Time horizon	defined end	continuous

Changes	exceptions (to be avoided)	standard (adaptation intended)
Extent	completeness desirably	focus on essential
Meetings	few, long meetings	many, short meetings
Templates	for matrix chains only	for matrix chains and content of matrices
IT-Support	documentation	documentation and communication (virtual teams)

Table 3: 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 that 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 is 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 (figure 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 fulfils the identified customer requirements as well as possible under the given development conditions and time restriction.

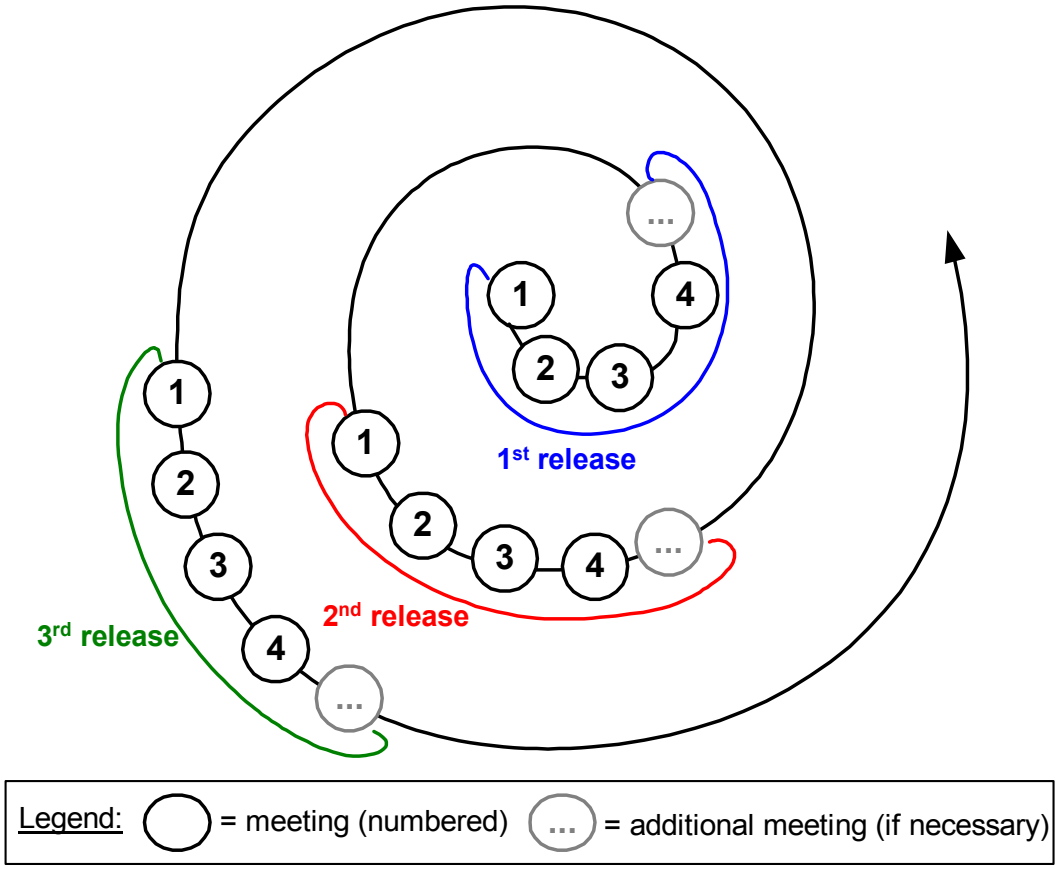

Figure 6: Lifecycle of a CQFD project (according to Boehm's spiral model (Boehm 1988))

4 EVALUATION OF CQFD

To evaluate CQFD from a methodological point of view we confront the demands derived from the characteristics of m-commerce application (see chap. 3) with the expected effects of the CQFD elements.

For this we use a matrix similar to the HoQ notation (figure 7).

legend:
CR = customer requirements
PC = product characteristics

PROBLEMS/Requirements \ SOLUTIONS/Characteristics	**incremental planning & implementation cycles**	- simultaneous collection of CR & PC	- large number of short meetings	- gradually refined weighting	- focus on most important CR/PC	- all matrices developed incrementally	- simultaneous planning and development	**employment of IT (QFD tools & Internet)**	**use of templates**
unclear CR/PC		9	9	9	3	3	9		3
dynamic CR/PC			9	9			↯	3	
uncertain PC (feasibility)							9		
time pressure		9	3		9	9	9	9	9

Figure 7: Evaluation of CQFD

Proceeding in iterations allows for new items to be included in the prioritisation matrices and thus change priorities among development targets at any point. Besides, this provides 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 are likely to be satisfied that planning due to IT support and the use of templates could be completed on schedule. Thanks to simultaneous planning and presentation of weekly builds (weekly implemented pre-products) everyone will be aware of the current status of the product and customer representatives have a chance to give their feedback, which also serves as guideline to focus further development. The weekly builds enhance also visibility and feasibility of the product characteristics. The unclearness of customer requirements and product characteristics are reduced not only by the weekly builds but also by use of templates and incrementally developed matrices.

Because of the early realization activities some design and/or implementation decisions (e.g. technology packages) have to be taken which made reacting to changing customer requirements and product characteristics more difficult (see ↯ symbol in fig. 6).

Not at least, since the CQFD method is already successful used in developing E-Commerce-Application one could expect, after now the method parameters have been adopted to M-Commerce environments, to gain similar advantages within those projects.

5 CONCLUSION

To conclude, 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 fact, that until now only enterprises have experiences with using QFD for e-commerce applications in no phase of our work a general validity of the results was raised. However, even if the results are not approved in terms of statistics, it becomes clear that QFD offers a chance for a fundamental advance towards more customer-oriented m-commerce development.

REFERENCES

1. Arthur, W. B. 1996. Increasing Returns and the New World of Business, Harvard Business Review, 74 (4): 100-109.
2. Boehm, B. W. 1988. A Spiral Model of Software Development and Enhancement, IEEE Computer, 21 (5): 61-72.
3. Cohen, L. 1995, Quality Function Deployment, How to make QFD work for you, Reading, Massachusetts, Addison-Wesley.
4. Dieterich, B., et al: Mobile Business. Märkte, Techniken, Geschäftsmodelle, Gabler-Verlag, 2001
5. Haag, S. T. 1992, A field study of the use on quality function deployment (QFD) as applied to software development, University of Texas at Arlington.
6. Herzwurm, G., Schockert, S., and Mellis, W. 2000, Joint Requirements Engineering, QFD for Rapid Customer-Focused Software and Internet Development, Wiesbaden, Vieweg – Gabler.
7. Herzwurm, G. and Schockert, S 1999. Virtual Product Development. Proceedings of the Fifth International Symposium on Quality Function Deployment and the First Brazilian Conference on Management of Product Development, Belo Horizonte, Brazil, pp. 83-91.
8. Kemper, H.G.; Wolf, E.: "Iterative process models for mobile application systems – a framework-. WOSP 2000: Second International workshop on Software and performance, Ottawa Canada, September 2000
9. Mellis, W., Herzwurm, G., Müller, U., Schlang, H., Schockert, S, and Trittmann, R. 2001 (forthcoming), Rapid Software Development, Aachen, Shaker.
10. Teichmann R., Lehner, F.: Mobile Commerce. Grundlagen, Strategien, Geschäftsmodelle, Springer-Verlag, Berlin, 2002
11. Wolf, Elke: Development of a lifecycle oriented process model for mobile banking application systems (B2C), university of cologne, 2002 (This paper is unpublished, yet)
12. Yourdon, E. 1995. When Good Enough Software is best, IEEE Software, 12 (5): 79-81.
13. Zultner, R. E. 1994. Software quality function deployment - the north American experience. SAQ, EOQ-SC (ed.): Software Quality Concern for People. Proceedings of the Fourth European Conference on Software Quality, Zürich, Swiss, pp. 143-158.
14. Zultner, R. E. 1990. Software Quality [Function] Deployment. Applying QFD to software. QFD-Institute (ed.): Transactions from the Second Symposium on Quality Function Deployment. Novi, Michigan, USA, pp. 132-149.

ABOUT AUTHORS

Prof. Dr. GEORG HERZWURM

is director of the Professorship for Information Systems at Technical University Dresden, Germany and EOQ Quality Auditor. He is the founder and president of the QFD Institut Deutschland e. V., a non-profit organization to improve knowledge, application, methodology and use of QFD in all areas of academics and industry. G. Herzwurm is author of two QFD books and numerous QFD articles as well as a team leader of several QFD projects in different industries. G. Herzwurm is recipient of the 2000 Akao Prize. He was awarded for outstanding contribution to the advancement of QFD.

Contact address:

Dresden University of Technology, Professorship for Information Systems, particularly Information Systems for Service Industries, Münchner Platz 3, 01062 Dresden; Germany

Tel.: +49 (0) 351 / 463-32268; Fax: +49 (0) 351 / 463-32736, Email: georg@herzwurm.de

Dipl.-Wirt.-Inf. SIXTEN SCHOCKERT

born 1969, holds a Masters degree in Information Systems from the University of Cologne, Germany; researcher and lecturer at the department of information systems, system development of Prof. Dr. Werner Mellis at the University of Cologne; member of the QFD Institut Deutschland (QFD-ID); working outside the university as trainer and consultant especially for Software Engineering, customer oriented software development, QFD and project management; co-author of many publications in the area of software management and development especially Software Quality Function Deployment; currently working on his dissertation about the impact of situational factors on Requirements Engineering

Contact address:

Sixten Schockert, Department of Information Systems, System Development, Albertus-Magnus-Platz, 50923 Köln, Germany

Tel.: +49 (0) 221 / 470-5370, Fax: +49 (0) 221 / 470-5386, Email: sixten.schockert@uni-koeln.de

ULRIKE DOWIE

studied economics at the University of Cologne and majored in business computing. Co-developing Continuous QFD was the subject of her final project. After graduating in May 2000, she started working in software administration and R&D controlling for Infineon Technologies AG, Munich, where she is still currently employed.

Contact address:

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

Tel.: 49 (0) 89 /234-26963, Email: Ulrike.Dowie@infineon.com

Dipl.-Wirt.-Inf. MICHAEL BREIDUNG

studied Information systems at the University of Halle/Wittenberg. In the recent few years he was head of the IT-Department of a corporation group, listed at the “New market” of the Frankfurt stock exchange and therefore responsible for the E- and M-Commerce activities of all corporations of the group. Last year he changed to the Dresden University of Technology, Professorship for Information Systems, particularly Information Systems for Service Industries and works there as researcher and lecturer.

Contact address:

Dresden University of Technology, Professorship for Information Systems, particularly Information Systems for Service Industries, Münchner Platz 3, 01062 Dresden; Germany

Tel.: +49 (0) 351 / 463-32735; Fax: +49 (0) 351 / 463-32736, Email: michael.breidung@mailbox.tu-dresden.de

Preferred Mailing Addresses:

Prof. Dr. Georg Herzwurm

Dresden University of Technology,

Professorship for Information Systems

particularly Information Systems for Service Industries

Münchner Platz 3,

01062 Dresden; Germany

Tel.: +49 (0) 351 / 463-32268;

Fax: +49 (0) 351 / 463-32736,

Email: georg@herzwurm.de

Dipl.-Wirt.-Inf. Michael Breidung

Dresden University of Technology,

Professorship for Information Systems

particularly Information Systems for Service Industries

Münchner Platz 3,

01062 Dresden; Germany

Tel.: +49 (0) 351 / 463-32735;

Fax: +49 (0) 351 / 463-32736,

Email: michael.breidung@mailbox.tu-dresden.de