R. Keil-Slawik, H. Selke, G. Szwillus (Hrsg.): Mensch & Computer 2004: Allgegenwärtige Interaktion. München: Oldenbourg Verlag. 2004, S.

Requirements- und Usability-Engineering für Standard-Software: Beispiel StarOffice/OpenOffice.org

Matthias Müller-Prove

Sun Microsystems GmbH

Abstract

Zur Ermittlung der Anforderungen für eine neue Programmversion werden von der Marketingabteilung verschiedene Methoden eingesetzt. Zu nennen wären hier Fokusgruppen, Fragebögen, Interviews, Kundenbesuche (Site Visits) und Konkurrenzanalysen. Obwohl sich die Methoden der Marketing- und der User Experience-Fachleute formell sehr ähneln, sind doch die Fragestellungen der User Experience Sicht sehr viel tiefer gehend. Der Usability-Experte muss sich intensiv mit den Beziehungen zwischen dem Benutzer, dem Computer, zwischen den zu erledigenden Aufgaben und dem Arbeitskontext beschäftigen. Der Methodenkanon erweitert sich folglich um Task-Analyse, Szenarien, User Profiles, Personas und Usability-Testing. Zu diskutieren ist außerdem, inwiefern die ermittelten Anforderungen der Zielgruppe entsprechen. Da man – anders als bei der partizipativen Software-Entwicklung – nur mit einem sehr kleinen Ausschnitt der Benutzergruppe in Kontakt kommt, besteht die Gefahr, dass wesentliche Aspekte der Anforderungen unterrepräsentiert bleiben.

1 Einleitung

Standard-Software sind Programme, die für einen großen Anwenderkreis entwickelt werden. Im Gegensatz zu Individualsoftware, die man in enger Zusammenarbeit mit dem Auftraggeber entwickeln kann, muss Standard-Software die Anforderungen von vielen Kunden abdekken. Die Software muss sich am Markt behaupten und konkurriert dabei mit anderen Produkten, die einen ähnlichen Funktionsumfang besitzen. Das klassische Feld von Standard-Software sind Büroanwendungen: Textverarbeitung, Tabellenkalkulation und Präsentation. StarOffice und sein Open Source Pendant OpenOffice.org decken diesen Anwendungsbe-

reich ab und sind damit unter Linux das führende Office-Paket und unter Windows der bedeutendste Konkurrent von Microsoft Office. Die Unterschiede zwischen StarOffice und OpenOffice.org sind für diese Diskussion derart marginal, dass im folgenden nur noch von OpenOffice.org gesprochen wird.

OpenOffice.org wird maßgeblich von Sun Microsystems entwickelt. Das bedeutet, dass auch die Verantwortung für die Requirements-Analyse und das User Interface und Software-Design bei Sun liegt. Es sein angemerkt, dass OpenOffice.org damit ein untypisches Open Source Projekt ist; in vielen anderen Open Source Projekten ist die Community stärker in die Anforderungsermittlung und das User Interface Design involviert (Benson et al. 2004).

Der folgenden Abschnitt 2 gibt einen Einblick in die verwendeten Prozesse. In Abschnitt 3 werden kurz verschiedene Arten von Requirements vorgestellt und in Abschnitt 4 die Methoden, die vom Produktteam für das Requirements- und Usability Engineering und letzlich für die Gestaltung der Software angewendet werden.

2 Prozesse und Dokumente

Die Entwicklung einer Vollversion dauert in etwa eineinhalb Jahre. Dieser Zeitraum gliedert sich gemäß des *Sun Product Life Cycles* (PLC) in mehrere Phasen, von denen die ersten beiden hier von Interesse sind. Es sind dies die Konzeptphase P1 und die Planungsphase P2. Während der ersten Phase erstellt die Marketinggruppe das Market Requirements Document (MRD) und unter Mitwirkung des User Experience Teams das Product Requirements Document (PRD). Das Ende der Phase ist erreicht, wenn das Product Concept Document (PCD) erstellt ist, das die geplanten Veränderungen in der Software beschreibt. In der darauf folgenden Planungsphase werden die Ideen aus dem PCD verfeinert und genau spezifiziert; vgl. (Vogt 2003). Das folgende Abbildung veranschaulicht die zeitliche Anordnung der ersten Phasen im Product Life Cycle.

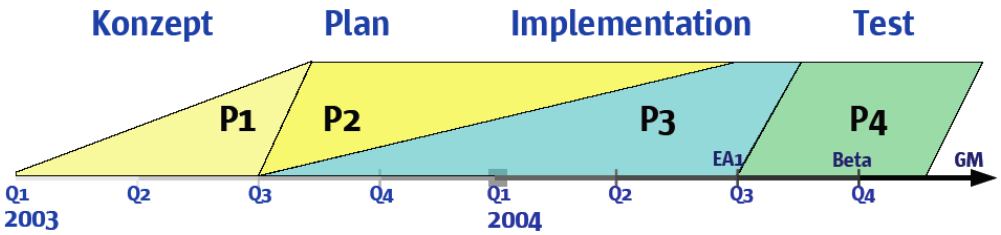

Abbildung 1: Die ersten 4 Phasen des Product Life Cycle in zeitlicher Übersicht

Innerhalb der Planungsphase regelt der *Specification-Workflow* das Vorgehen zur Erstellung der Spezifikationen. Für jede neu geplante Funktion bildet sich ein Team mit jeweils einem Vertreter aus den Bereichen User Experience, Engineering, Qualität und Dokumentation.

Einer dieser vier ist Autor der Spezifikation (kurz „Spec“), während die anderen kritische und konstruktive Berater sind. Die Idee ist, dass eine Spezifikation vom gesamten Team genehmigt wird, bevor die Implementation beginnt. Anderungen auf Papier sind bekanntlich weit weniger aufwändig, als es die entsprechende Anderung im Source-Code wäre. In der Praxis gibt es aber zumeist eine Periode, in der – durch den „Feinschliff“ bedingt – sich das Fertigstellen der Spezifikation und die Implementation überlappen.

Es wird ein Spezifikations-Template benutzt, das aus den folgenden Teilen besteht (OpenOffice.org Specification Project 2003):

- Titel, Status, TaskIDs, Abstract, Team, Approvals, History, Glossary
- Motivation, User Scenarios, Competitive Analysis
- Goals, Requirements and Dependencies
- Detailed Specification
- Future Tasks, Notes, References

3 Requirements

Anforderungen an eine neue Produktversion kommen aus den verschiedensten Richtungen. Im MRD werden allgemeinere Marktanforderungen gesammelt, wie beispielsweise Accessibility – 508-Konformität ist eine Bedingung für den Einsatz der Software in Amerikanische Behörden – oder auch die Kompatibilität mit Microsofts Dateiformaten.

Im PRD listet die Marketingabteilung Benutzungsprobleme auf. Dabei wird zu jedem konkret ermittelten Problem ein Requirement formuliert, für das später eine Lösung gefunden werden soll. Im Falle von OpenOffice.org kommen derartige Anforderungen auch direkt aus der Open Source Community. In jedem Fall ist es aber wichtig, dass das PRD die Lösung nicht vorwegnimmt, da man an dieser Stelle im Prozeß noch keinen Überblick über die neuen Version besitzt. Außerdem liegt die Kernkompetenz des Marketings nicht in der Gestaltung von Software.

Eine dritte Sorte von Requirements kommt von Großkunden und OEM-Partnern, die mit unter sehr genaue Vorstellungen von der Software haben. Die Verhandlungen zu solchen Partnern ist damit wohl mit denen der Individual-Software-Erstellung zu vergleichen.

4 Methoden

Für den Bereich der software-ergonomischen Evaluation unterscheiden Reinhard Oppermann und Harald Reiterer zwischen i) subjektiven, ii) objektiven, iii) Leitfaden-orientierten und iv) experimentellen Evaluationsmethoden (Oppermann & Reiterer 1994). Da die Evaluation der

Requirements-Analyse sehr verwandt ist, werde ich diese Klassifizierung für den hier vorgestellten Methodenkanon übernehmen.

4.1 Subjektive Methoden

Subjektive Methoden beziehen sich unmittelbar auf die Beurteilungen und Einschätzungen der Benutzer. Bei Sun nennt sich dieser Teil des Requirements-Engineering „Voice of the Customer“. Mittels *Online-Fragebögen* werden mehrere hundert Kunden befragt, wie sie die aktuelle Software bewerten und welche Funktionen sie sich für die nächste Version wünschen. Durch den Open Source Ansatz werden solch Fragebögen auch auf der Community-Website eingesetzt. Die Zahl der Antworten ist dabei ungleich größer und beträgt bei einer aktuellen Umfrage bereits mehr als 200.000. Eine dritte Fragebogenvariante betrifft die Usability-Studien im Labor. Über den *SUS-Fragebogen* erheben wir neben den objektiven Daten subjektive Einschätzungen des Teilnehmers über die getestete Software.

Suns Marketing-Gruppe führt desweiteren *Interviews* beim Kunden und Diskussionen in *Fokusgruppen*, die zusätzlich dabei helfen sollen die Erwartungen des Kunden zu ermitteln.

Die Open Source Community formuliert ihre Wünsche auch ganz explizit als *Request for Enhancements* (RFEs) und legt diese mittels Web-Formular als Issues im Task-Tracking-System ab.

4.2 Objektive Methoden

Wir Menschen sind uns nicht immer – oder sogar nur selten – darüber bewusst, was und warum wir etwas tun. Um so schwieriger fällt es uns auch, uns von emotionalen Einflüssen frei zu machen und eine Situation richtig einzuschätzen (Vroon 1993). Objektive Methoden setzen nun anstelle der eigenen Bewertungen des Benutzers die Beobachtung. *Kundenbesuche (Site Visits)* haben das Potential bis dato unbekannte Arbeitsabläufe zu entdecken, da die User Experience Experten den Benutzer im Kontext seiner alltäglichen Arbeitsumgebung begleiten.

Eine weitere wichtige objektive Methode ist das *Usability-Testing*. In einer Laborsituation werden Benutzer an das eigene Produkt, das der Konkurrenz oder an einen Prototypen gesetzt, um damit eine gegebene Reihe von Aufgaben zu bearbeiten. Ein Prototyp muss dabei nicht immer ein Computerprogramm sein. Mit Paper-Prototypen lassen sich sehr kostengünstig verschiedene Designalternativen ausprobieren (Snyder 2003).

4.3 Leitfaden-orientierte Methoden

Im User Experience Team benutzen wir *Peer-Reviews*, um die Qualität der Entwürfe zu verbessern. Ein Kollege begutachtet dabei die Detailed Specification eines anderen Mitglieds des User Experience Teams in hinblick auf Konsistenz zu den Guidelines und zu anderen

Spezifikationen. Per *Cognitive Walkthrough* ergibt sich auch häufig eine Verbesserung des Designs.

4.4 Experimentelle Methoden

Diese Art von Methoden kommen bei der Entwicklung von OpenOffice.org bislang relativ selten zum Einsatz. Informelle Experimente im Bereich des Icon-Design dienen aber gelegentlich dazu, sich zwischen verschiedenen Varianten zu entscheiden.

4.5 Bewertung der Methoden

Den typischen Benutzer von Standard-Software gibt es nicht. Man hat sich also immer der Frage zu stellen, inwiefern die Benutzer, von denen man Daten erhoben hat, repräsentativ für die Zielgruppe der Software sind. Bei gezielten Fragebogenaktionen hat man dieses Problem ganz gut unter Kontrolle; aus einem Adresspool werden bestimmte Kunden ausgewählt und um ihre Mithilfe gebeten. Bei offenen Fragebögen, z.B. auf der Website von OpenOffice.org, ist nie klar, wer diese ausfüllt. Genau wie bei den RFEs bleibt zu vermuten, dass eine aktive Minderheit eine schweigende Mehrheit überstimmen könnte. Positiv ist aber anzumerken, dass Fragebögen recht gut skalieren, sowohl im Stichprobenumfang, als auch in globaler Hinsicht. Bei allen anderen Methoden ist es sehr aufwändig Requirements und Usability-Probleme einer internationalen Benutzergruppe zu ermitteln.

Bei Interviews und Fokusgruppen ist darauf zu achten, welche Rolle die Befragten haben. Hier kann der Unterschied zwischen Kunde, nämlich dem der die Software für eine größer Firma kaufen soll, und einem Benutzer sehr erheblich sein.

Alle subjektiven Methoden haben gemeinsam den Nachteil, dass sie nur eine vermittelte Sichtweise abbilden. Dennoch bieten sich hier sinnvolle Möglichkeiten der Requirements-Analyse, insbesondere wenn man seinen Gesprächspartnern im wortwörtlichen Sinne zuhört. Mark Federman führt das sehr schön aus in seiner Keynote anläßlich der Customer Relationship Management Conference: „Listening to the Voice of the Customer“ (Federman 2001).

Als letztes Bewertungskriterium sollen die Methoden danach aufgeteilt werden, ob sie eher für die Requirements-Analyse oder für das Aufdecken von Usability-Problemen geeignet sind. Zu der ersten Gruppe gehören die gezielten und offenen Fragenbögen, sowie Interviews und Fokusgruppen. Das Usability-Testing samt SUS-Fragebogen und die Peer-Reviews haben ihre Stärken auf der Usability-Seite. Gleichgut in beiden Kategorien sind SiteVisits und RFEs.

Literaturverzeichnis

Benson, C., Müller-Prove, M., Mzourek, J. (2004): Professional Usability in Open Source Projects: GNOME, OpenOffice.org, NetBeans. In: CHI 2004, Vienna. New York: ACM Press. S. 1083-1084 (Extended Abstracts). http://www.mprove.de/script/04/chi/

Federman, M. (2001): Listening to the Voice of the Customer. University of Toronto. http://www.mcluhan.utoronto.ca/VoiceoftheCustomer.pdf

OpenOffice.org Specification Project (2003). http://specs.openoffice.org

OpenOffice.org User Survey (version 1.0.1). http://oosurvey.gratismania.ro

Oppermann, R., Reiterer, H. (1994): Software-ergonomische Evaluation. In: Eberleh, E., Oberquelle, H., Oppermann, R. (Hrsg.): Einführung in die Software-Ergonomie. 2. Auflage. Berlin: de Gruyter. S. 335-371.

Snyder, C. (2003): Paper Prototyping. San Francisco, CA: Morgan Kaufman.

Vogt, P. (2003): Usability im Entwicklungsprozess. In: Heinsen, S., Vogt, P. (Hrsg.): Usability praktisch umsetzen. München: Hanser. S. 66-77. http://www.mprove.de/script/03/hanser/

Vroon, P. (1993): Drei Hirne im Kopf – Warum wir nicht können, wie wir wollen. Zürich: Kreuz.

Kontaktinformationen

Matthias Müller-Prove
Sun Microsystems GmbH
Sachsenfeld 4
20097 Hamburg
Email: mprove@sun.com