Bridging the Gap Between Analysis and Design

Ein toolbasiertes Vorgehensmodell zur Entwicklung Mobiler Systeme

Dr. Dennis Krannich
Universität Bremen
Bibliothekstraße 1
28359 Bremen
krannich@tzi.de

Abstract
Usability-Engineering beschäftigt sich seit über 30 Jahren mit der Analyse und Gestaltung von Computer-Systemen. Während Usability-Testing für stationäre Systeme meist standardisiert ist, stellen Mobile Systeme durch ihre Heterogenität neue Herausforderungen dar. In diesem Beitrag stellen wir ein toolbasiertes Vorgehensmodell vor, um die traditionelle Trennung von Analyse und Design aufzubrechen. Es besteht aus einem Phasenmodell und einem Rapid-Prototyping- und Usability-Testing-Instrument und ermöglicht (a) so früh wie möglich mit der Umsetzung prototypischer Entwürfe zu beginnen und diese hinsichtlich ihrer Usability zu untersuchen, (b) im originären Benutzungskontext und auf den späteren realen Endgeräten zu testen und (c) aussagekräftige Daten für die Evaluation zu sammeln.
Nach der Beschreibung der Kernelemente präsentieren wir die Ergebnisse unserer Evaluation. Die empirischen Studien bestätigten das Konzept und zeigten vielversprechende Möglichkeiten für einen Paradigmenwechsel innerhalb dieses Forschungsfeldes.

Keywords:
/// Mobile Usability
/// Rapid-Prototyping
/// Usability Testing
/// ripcord

1. Einleitung

In den frühen 1980er Jahren – als die ersten Computerprogramme erschienen – waren nur Art und Umfang der Funktionalität von entscheidender Bedeutung, ob eine Anwendung verwendet wird oder nicht. In den letzten drei Jahrzehnten haben sich jedoch Gegenstandsbereiche, Methoden und Instrumente ständig erweitert und analog zum Wandel der Computersysteme weiter entwickelt. Im Allgemeinen gibt es vier Aspekte, die das Usability-Engineering beeinflusst haben: (1) die Vision des Computers als Medium; (2) die Entwicklung von Desktop- und Internet-Anwendungen mit komplexen grafischen Benutzungsoberflächen (GUI); (3) der Einsatz von Mobilen Systemen, nicht nur für die Sprachkommunikation, sondern als universelles und multifunktionales Gerät; und (4) die Verlagerung von performance- und aufgabenorientierten Systemen zu Erfahrungen mit und durch digitale Produkte. Um Mobile Systeme mit einem angenehmen Benutzungserlebnis und einer guten Gebrauchstauglichkeit zu entwickeln, müssen sich Entwickler und Designer daher mit den Herausforderungen und Beschränkungen, die durch die Mobile Welt verursacht werden, beschäftigen. Mit der Einführung von Smartphones, dem iPhone und dem iPad hat sich der Bereich der Mobilen Systeme erneut verändert, nicht nur in Bezug auf Formfaktor, sondern insbesondere in Bezug auf die Nutzungsszenarien (z. B. beiläufige Benutzung und weniger komplexe Aufgaben), Anwendungstypen (z. B. Casual Games, Augmented Reality und immersive Anwendungen) und Interaktionsmodalitäten (z. B. Multitouch, Gesten- und Sprachsteuerung).

So überrascht es nicht, dass Forscher und Praktiker zunächst versucht haben, die traditionellen Methoden des Usability-Engineering eins zu eins für diese neuen Systeme anzuwenden, um die Interaktion zu bewerten und die Zufriedenheit der Gebraucher zu messen. Leider mit wenig Erfolg: Verschiedene Autoren bemängeln die schlechte Usability Mobiler Systeme (z. B. Weiss 2002; Bernhaupt 2008) und sehen dies unter anderem in der fehlenden Berücksichtigung des Benutzungskontextes begründet (z. B. Bernhaupt 2008).

Auch wenn das vorliegende Thema etwas antiquiert erscheint, so sind die bisher erreichten Ergebnisse wenig zufriedenstellend. Dies lässt sich anhand von zwei Punkten verdeutlichen:

(1) Die hauptsächliche Begrenzung der klassischen Methoden besteht in ihrer Laborfixierung und damit in der Ausblendung des Benutzungskontextes. Der Laboransatz geht von dem „Irrglauben" aus, die Gebrauchstauglichkeit eines Systems sei nur von den Eigenschaften des Systems abhängig. Diese Annahme ist gänzlich obsolet, wenn sich Systeme nicht mehr an einem Ort (Arbeitsplatz) befinden, sondern von ihren Gebrauchern durch die Welt bewegt werden, denn die Interaktion differiert, da sie (a) oft unterbrochen wird, (b) sehr stark vom Benutzungskontext

abhängt und (c) häufig an Orten stattfindet, die für eine Interaktion gänzlich ungeeignet erscheinen.

(2) Seit dem Beginn der Software-Ergonomie existiert eine klassische Trennung zwischen Analyse und Design. So gibt es die Zuständigkeit der (Arbeits-)Psychologie für die Analyse und Kritik der Usability-Probleme und die Zuständigkeit der Informatik für die Entwicklung und Einführung derartiger Systeme. Die Übersetzung der Analyseergebnisse in konstruktive Anforderungen stellt eine kaum bewältigte Herausforderung dar, eine möglicherweise grundsätzliche methodische Lücke, die aus dem unterschiedlichen Methodenverständnis von Sozial- und Technikwissenschaften resultiert.

Um diese Herausforderungen endgültig zu meistern, sind daher neue Methoden, Werkzeuge und Vorgehensmodelle erforderlich, die gemeinsame Aspekte der mobilen Geräte reflektieren und ihre Unterschiede respektieren. Gemeinsame Aspekte sind nicht nur durch physikalisch-stoffliche Eigenschaften (z. B. Bildschirmgrößen, Eingabemodalitäten, Formfaktoren und Haptik) und Software-Einschränkungen (wie z. B. Speicher-und Computing-Ressourcen) begrenzt, sie umfassen auch den (meist heterogenen) Benutzungskontext. Das allgemeine Ziel sollte daher sein, abstrakte Modelle und praktische Instrumente zu einer Synthese zu bringen und diese in den gesamten Entwicklungsprozess so zu integrieren, dass Usability-Testing nicht mehr als Basis für eine ex-post-Kritik an Usability-Problemen des Endprodukts zu verstehen ist – die meist folgenlos bleibt, weil Änderungen praktisch nicht mehr möglich sind – sondern als Brücke zwischen Analyse und Design. Dabei geht es weniger darum, existierende Methoden in Frage zu stellen, sondern ein neues Bewusstsein zu schaffen, Analyse und Design als einen geschlossenen Prozess der Systementwicklung zu verstehen, und möglichst früh im Entwicklungsprozess mit der Entwicklung, Exploration und Evaluation von Prototypen unter realen Bedingungen zu beginnen.

Darüber hinaus sollten wir uns für zukünftige technologische Entwicklungen rüsten: Durch die Etablierung generativer Fertigungstechniken (z. B. in Form von Fab Labs) sind wir aufgrund kostengünstiger Technologien (z. B. RepRap oder MakerBot Replicator) in der Lage stoffliche Artefakte selbst zu kreieren, die wir mit Elektronik und einer grafischen Benutzungsoberfläche zum Leben erwecken können. Gerade diese technologischen Entwicklungen ermöglichen uns eine holistische Betrachtung eines Systems bestehend aus Hard- und Software, anstatt diese als zwei getrennte Bereiche zu verstehen. Somit könnten wir das Konzept des Rapid-Prototypings auf beide Bereiche gleichermaßen anwenden und deren Zusammenhänge und Abhängigkeiten erforschen.

2. Methodische Ansätze des Mobile Usability-Testings

Die Unterschiede von mobilen Geräten und deren Anwendungen stellen eine große Herausforderung an das Usability-Engineering und die Software-Entwicklung im Allgemeinen: Eingabemethoden, Formfaktoren, Betriebssystemen, zusätzliche Features wie Netzwerkfähigkeit und ihre Anwendungsgebiete unterscheiden sich signifikant. Für die Durchführung von Usability-Tests (Benutzertests) ist es daher erforderlich (a) aussagekräftige Daten für die spätere Analyse zu sammeln, (b) die Tests im Nutzungskontext und auf realen Endgeräten durchzuführen und (c) die Test so früh wie möglich anzusetzen, um kostenintensive Ausbesserungen oder Neuentwicklungen des Produkts zu verhindern.

Es existieren diverse Ansätze zur Aufzeichnung der Interaktionen, in denen beispielsweise Kameras verwendet werden (z. B. Holtz-Betiol u. de Abreu-Cybis 2005; Kiljander 2004 oder Roto et al. 2004). Alle Methoden haben jedoch eines gemeinsam: Sie beeinflussen die Haptik des Geräts und verändern dessen Formfaktor erheblich, indem sie das Endgerät in eine verkabelte und unhandliche Maschine verwandeln und die erfassten Daten durch Lichteinflüsse und starkes Ruckeln beeinträchtigen.

Zudem sind derartige Ansätze mit einem hohen Vorbereitungsaufwand (z. B. durch Verkabelung) verbunden und stellen – bedingt durch die Behinderung in der Benutzung und dem erhöhten Aufmerksamkeitsfaktor in der Öffentlichkeit – einen psychologischen Hemmfaktor für die Testperson dar. Letztendlich geht der ursprüngliche Charakter eines kleinen und mobilen Endgerätes verloren.

Verschiedene Studien (u. a. Kjeldskov & Stage 2004; Rantanen et al. 2002), haben bewiesen, dass der originäre Benutzungskontext in Labortests nicht hinreichend simuliert werden kann. Bernhaupt et al. (2008) merken an, dass nicht einmal so genannte Portable Usability-Labore, bei denen das Test-Equipment zum Kunden beziehungsweise in den Benutzungskontext gebracht wird, diese Beeinträchtigungen bewältigen können.
Der starke Nutzen von Emulatoren und Simulatoren auf einem Desktop-Computer wurde zwar in verschiedenen Untersuchungen gezeigt, dennoch werden bestimmte Aspekte, wie die Haptik, der Benutzungskontext und die technische Beschränkungen und Eigenschaften stark vernachlässigt (vgl. u. a. Dahl et al. 2009; Ballard 2008).

Das frühzeitige Durchführen von Usability-Tests ist weitgehend verbreitet (Weiss 2002; Bernhaupt et al. 2008). Ansätze wie das Participatory Design oder das User-Centered Design sollen Akzeptanz und eine Steigerung der Qualität forcieren. Dabei sind herkömmliche Prototyping-Verfahren als proof-of-concept hilfreich und äußerst zeit- und kostensparend. Ein derartiges Vorgehen findet jedoch selten im vollen Umfang – durchgehend von der Konzeption bis zum fertigen Produkt – statt.

Es lässt sich daher konstatieren, dass, obwohl verschiedene Werkzeuge zum Erstellen von Prototypen und ein breites Spektrum methodischer Ansätze zum Mobile Usability-Testing existieren, effektive Tools mit denen man schnell und iterativ grafische Benutzungsoberflächen erstellen und diese direkt auf dem Zielgerät im originären Benutzungskontext testen kann, fehlen.

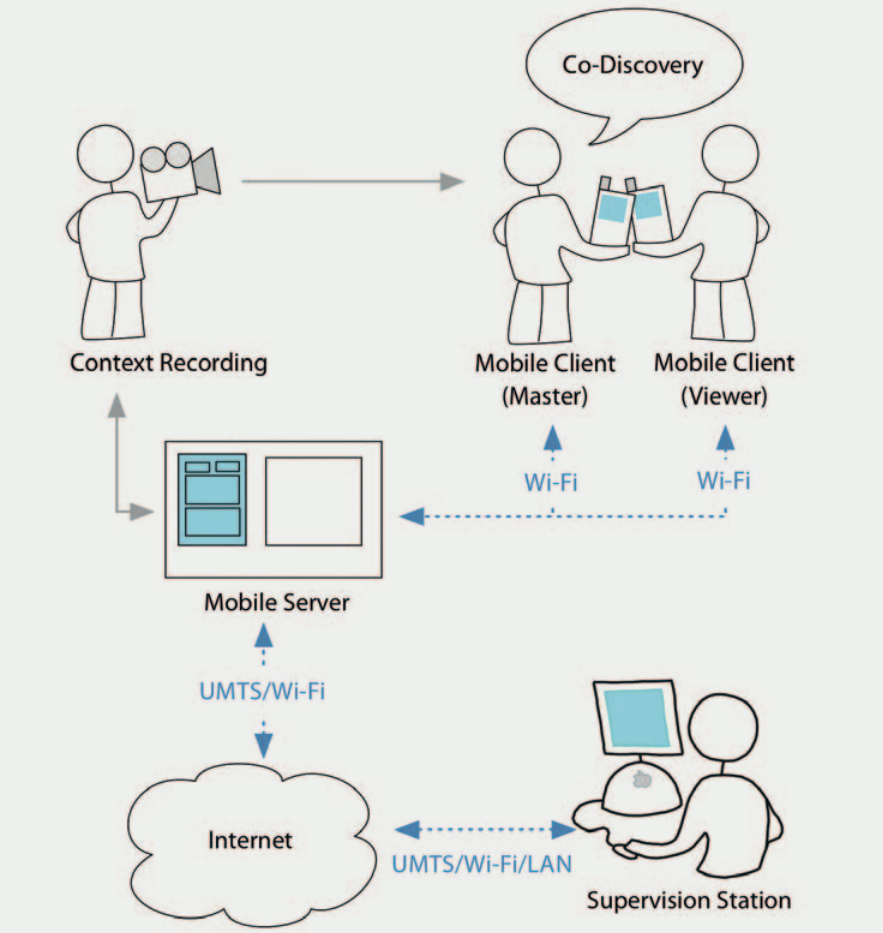

Abb. 1.

3. Ein Toolbasiertes Vorgehensmodell

Basierend auf dem Mangel an geeigneten Werkzeugen und Methoden für die Entwicklung Mobiler Systeme, haben wir ein toolbasiertes Vorgehensmodell, bestehend aus drei Komponenten, entwickelt: (1) ein Instrument (ripcord2.0) zum Rapid-Prototyping und Usability-Testing; (2) ein Phasenmodell zum Einbinden von Rapid-Prototyping und Usability-Tests in den Software-Entwicklungsprozess; und (3) ein Metamodell zum Postulieren von Forschungsfragen, zur Bestimmung von Arbeitsaufgaben und Prüfkriterien, und zur Auswahl des geeigneten Testansatzes (nicht in diesem Beitrag behandelt, eine detaillierte Beschreibung ist in (Krannich 2010) zu finden).

3.1. ripcord2.0 = Rapid-Prototyping + Usability-Testing Tool

Im Folgenden beschreiben wir die allgemeine Funktionalität des ripcord2.0 Systems, wie in der **[Abb. 1]** dargestellt. Eine detaillierte Beschreibung kann in (Krannich2010) gefunden werden.

Genereller Ansatz: Das generelle Konzept sieht zwei wesentliche Bestandteile vor: (1) die einfache, schnelle und iterative Erstellung und Modifikation von GUI-Prototypen in Form von HTML5 + CSS3 + JS und (2) die Trennung von Aufzeichnungsfunktionalität und Bereitstellung der Prototypen. Der Mobile Client (MC) sendet die getätigten Eingaben/Interaktionen an den Mobile Server (MS). Dieser kann – je nach den Gegebenheiten des mobilen Endgeräts – entweder auf einem separatem Gerät oder auf dem Testgerät selbst installiert sein. Der MS interpretiert die Eingaben und sendet ein Ergebnis an den MC zurück. Dies ist entweder ein Screenshot (J2ME-basierte Endgeräte) oder eine HTML-Seite (iOS-basierte Endgeräte). Der MS zeichnet parallel Audiosignale auf und fügt diese mit den Bildschirmabgriffen, dem Gesicht des Benutzers (iOS-basierte Endgeräte mit einer Frontkamera) und Kontextinformationen (z. B. Touch-Position, Gesten, Tastatureingabe, GPS-Koordinaten und Zeitstempel) zu einer Videodatei zusammen. Durch die Verbindung mehrerer MC (1 Master, n Viewer) soll das Testen in Gruppen (basierend auf der Co-Discovery Methode) ermöglicht werden. Als Verbindung zwischen MS und MC können je nach Unterstützung des Endgeräts Bluetooth oder Wi-Fi verwendet werden. Eine Supervision Station zur entfernten Beobachtung und Steuerung kann via UMTS oder Wi-Fi angebunden werden.

Entfernte Beobachtung und Steuerung: Mit der Supervison Station (SuS) kann der Testleiter auf den MS über das Internet zugreifen, um sich Bildschirminhalte und Interaktionen anzuschauen. Er kann auch Änderungen an der Anwendung vornehmen, wie zum Beispiel das Aufrufen einer alternativen Version. Darüber hinaus kann der Testleiter Teile oder sogar den kompletten Prototypen ersetzen, auch während des Tests. Auf diese Weise können Fehler, die zum Abbruch geführt hätten, beseitigt werden oder alternative Benutzungsoberflächen oder Interaktionskonzepte „on-the-fly" umgesetzt und ausgewertet werden (die Testbedingungen sollten hierdurch jedoch nicht geändert werden). Wir nennen diesen Ansatz „Remote Rapid-Prototyping".

Aufnahme des Benutzungskontextes: Context Recording (CR) wird verwendet, um mögliche Einflüsse der Testperson sowie das

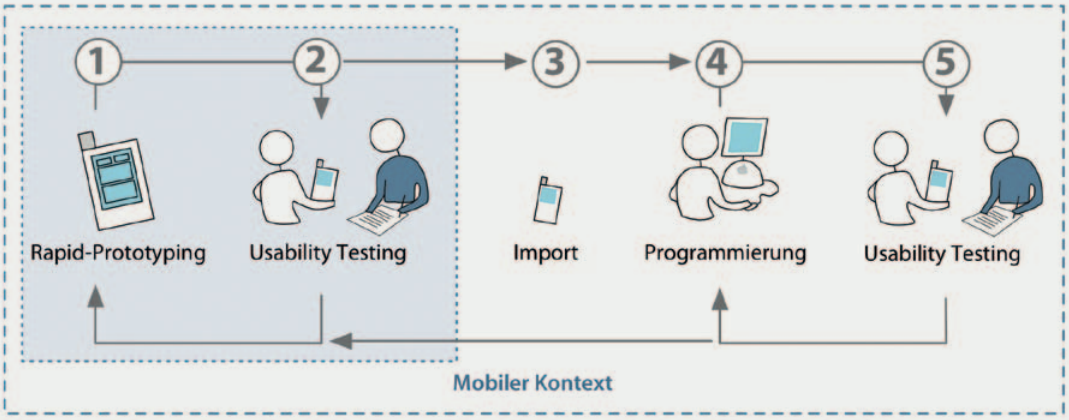

Abb. 2.

Umfeld, in dem sich die Testperson befindet, aufzuzeichnen. Dies kann durch eine weitere Person, Kameras oder Sensoren erfolgen (GPS-Koordinaten, Zeit, Temperatur, Lichtverhältnisse, Lautstärke, etc.).

Schnelles und iteratives Erstellung von Prototypen (Rapid-Prototyping): Der Prototyp wird mit HTML5, CSS3 und JavaScript erstellt. Die Auswahl dieses Ansatzes ermöglicht nicht nur Programmierern, sondern auch Designern – die mit einer komplexen Programmiersprache (wie Objective-C oder Java) nicht vertraut sind – Prototypen zu erstellen. Spezielle HTML-kompatible Attribute können für Ereignisse (wie z. B. Vibration oder Ton) verwendet werden. Neben der reinen Verwendung von HTML können Entwickler CSS für die Gestaltung und JavaScript für komplexe Interaktionen nutzen, sowie Bibliotheken wie jQuery einbinden. So können die erstellen Prototypen leicht in andere Web-basierte Frameworks wie Apache Cordova (PhoneGap) oder Appcelerator Titanium überführt werden.

3.2. Phasenmodell

Das Phasenmodell besteht aus fünf Phasen (siehe **[Abb. 2]**). Der Ablauf der einzelnen Phasen ist nicht als streng lineares Schema zu verstehen. Bei Bedarf werden Schleifen und Rückkoppelungen zwischen den einzelnen Phasen durchgeführt, das heißt, es können zum Beispiel Bestandteile auf dem Programmierungsprozess (Schritt 4) wieder in den Prototyping-Prozess (Schritt 1 und 2) übergeben werden. Schritt 3 stellt die Schnittstelle zwischen Prototy- ping und der eigentlichen Produktentwick- lung dar.

Das toolbasierte Vorgehensmodell ermöglicht die Erprobung neuer Funktionen, Gestaltungselemente und Interaktionskonzepte, wobei diese (a) ganz bewusst aus dem Konzept heraus prototypisch umgesetzt werden, (b) im Laufe des Entwicklungsprozesses hinzukommen, (c) als Proof-of-Concept oder Machbarkeitsstudie umgesetzt werden, oder (d) iterativ verändert werden. Wesentliches Augenmerk liegt in der Skalierbarkeit in den einzelnen Phasen und der Flexibilität zwischen den einzelnen Phasen. So können beispielsweise unterschiedliche Prototyping-Ansätze (z. B. low- und high-fidelity oder horizontal und vertikal) verwendet werden oder beliebig viele Bestandteile der Prototypen in die Implementierung übernommen werden.

3.3. Technische Implementierung

Das Rapid-Prototyping- und Usability-Testing-Instrument wurde exemplarisch für den Mobile Server (MS) in Objective-C für Mac OS X und für den Mobile Client (MC) in J2ME (MIDP2.0/CLDC1.1 midlet) für ein Mobiltelefon umgesetzt. Darüber hinaus existieren eine Portierung des Ansatzes für iOS-basierte Endgeräte, sowie eine Weiterentwicklung (in Bearbeitung), die den Mobile Client und den Mobile Server in einer Anwendung kombiniert und ausschließlich für iOS-basierte Endgeräte ausgerichtet ist.

Zur Verbindung zwischen MS und MC können entweder Bluetooth (J2ME-Geräte) oder WiFi (via Bonjour Service) verwendet werden. Die Kommunikation zwischen MS und MC erfolgt über den Austausch spezieller „ripcord Nachrichten" (RN), die über einen bidirektionalen Kanal gesendet werden, wobei die Reihenfolge dieser Nachrichten sehr genau festgelegt ist. Jede der Nachrichten besteht aus einem Kopfbereich (Header) und einem für die Nachricht spezifischen Inhaltsbereich (Body). Alle Integer werden als Big Endian abgespeichert. Es werden die folgenden sieben Message Type Identifier (MTI) unterschieden: RN1 Hello Server, RN2 Image, RN3 Event, RN4 JFIF head, RN5 JFIF tail, RN6 Vibrate, RN7 Goodbye, RN8 Mouse Event und RN9 URL Change Event. Bildschirminhalte können entweder an den MC gesendet werden (J2ME-Geräte) oder an den MS gesendet werden (iOS-Geräte).

Der idealtypische Anwendungsfluss für ein iOS-Gerät sieht wie folgt aus: Nach Tippen auf einen Link wird die Fingerposition an den MS gesendet. Anschließend wird auf

dem MC ein Bildschirmfoto erstellt, an den MS gesendet und dort zusammen mit der Fingerposition als ein neues Keyframe gerendert und im Videostream gespeichert. Als nächstes wird ein HTTP Request an den MS gesendet und die korrespondierende Website empfangen (HTTP Response) und auf dem MC angezeigt. Der neue Bildschirminhalt wird abgegriffen und als Bild an den MS gesendet, wo es als neues Keyframe im Videostream gespeichert wird. Parallel dazu werden Audio und die Frontkamera des mobilen Geräts sowie die GPS-Position aufgezeichnet.
Bei J2ME-basiertem Endgeräten werden die Tastatureingaben an den MS gesendet, dort interpretiert (z. B. Ausführen eines Links) und die geänderten Bildschirmsegmente als komprimiertes JPEG-Bild an den MC gesendet. Der Bildschirminhalt wird ebenso als neues Keyframe im Videostream gespeichert.

4. Ergebnisse der Evaluation und deren Analyse

Zum Testen und Bewerten des toolbasierten Vorgehensmodells wurde eine Methodentriangulation bestehend aus Beobachtung, Befragung und Experteninterviews verwendet. Es wurden fünf Workshops (mit einer Grundgesamtheit von 72 Teilnehmern) und vier Experteninterviews durchgeführt. Die Erkenntnisse aus den Workshops führten zu einer vorläufige Theoriebildung, die durch Experteninterviews überprüft wurde. Der Fokus der Evaluation lag nicht auf der Analyse einer bestimmten mobilen Anwendung, sondern auf der Bewertung des toolbasierten Vorgehensmodells im Allgemeinen. Eine detaillierte Beschreibung des Evaluationsdesigns ist in (Krannich 2010) zu finden.

Zusammenfassend kann ein erfolgreicher Verlauf und eine positive Einstellung der Befragten festgehalten werden. Betrachtet man die Ergebnisse der Workshops und Experteninterviews, so lässt sich feststellen, dass die Experten die gewonnenen Erkenntnisse aus den Workshops mit ihren Erfahrungen größtenteils bestätigen konnten. Nachteile wurden insbesondere in der Anwendbarkeit des Ansatzes in späteren Entwicklungsphasen gesehen.

Funktionalität und Performance: Sowohl die Workshop-Teilnehmer, als auch Experten bevorzugten das aktuelle System in Bezug auf Funktionalität und Aufwand für die Inbetriebnahme. Im Vergleich zu anderen Ansätzen ist ripcord2.0 einfach zu bedienen und hat keinen Einfluss auf die Testperson durch zusätzliche Geräte oder Kabel. Fast alle Befragten aus den Workshops kritisierten jedoch den fehlenden Anwendungsfluss (insbesondere das automatische Weiterleiten oder Aufblinken von Statusmeldungen). Die Hälfte der Testpersonen bemängelte die Übertragungsverzögerung, die durch die langsame Bluetooth-Verbindung verursacht wird. Im Gegensatz dazu haben die Experten die Übertragungsverzögerung und fehlende automatisierte Sequenzen als nicht gravierend bewertet. Aus ihrer Sicht hat dies keinen Einfluss auf das generelle Anwendungs- und Interaktionskonzept und ist ausreichend genug zum Explorieren und Evaluieren der Prototypen. Darüber hinaus wurden mehrere Nachteile entdeckt: (a) fehlende Animationen und flüssiger Spielablauf, (b) keine Berücksichtigung von Multiuser-Apps, und (c) keine Videowiedergabe möglich.

Einflüsse des mobilen Kontextes: Die Experimente zeigten, dass die Probanden bei Tests mit einem Simulator am Desktop-PC zwar konzentrierter waren, als im originären Benutzungskontext, dennoch konnten einige Usability-Probleme – insbesondere in Bezug auf das Interaktionskonzept – nur durch den mobilen Kontext aufgedeckt werden. Auch die Dauer der Tests unterschied sich stark. So ist zu schlussfolgern, dass einige Usability-Probleme nur durch Dual-Task-Situationen aufgedeckt und Aussagen über Interaktionsdauer und -fluss nur im originären Benutzungskontext getätigt werden können.

Potentiale der Supervision Station: Alle Befragten (Experten und Workshop-Teilnehmer) konnten die Testpersonen im Kontext beobachten und die Benutzungsoberfläche (Prototypen) aus der Ferne wechseln und verändern. Die Fähigkeit, Teile des Prototyps zu verändern, zu ersetzen oder hinzufügen, ermöglicht nicht nur sofort neue Variationen zu testen, sondern gibt Entwicklern auch die Möglichkeit, Fehler schnell zu beheben, die den Test abbrechen würden, und verhindert so ein aufwändiges Koordinieren neuer Tests.

Potentiale des Rapid-Prototyping: In Bezug auf Lesbarkeit und visuelle Repräsentation von Informationen, empfanden die Probanden die Simulatoren aufgrund ihres nicht-nativen Skalierungsfaktors angenehmer. Andererseits führte dieses Phänomen auch zu schweren Usability-Problemen: Vor allem in den Prototyping-Workshops konnten wir feststellen, dass die Probanden die Proportionen und Dimensionen des Bildschirms falsch eingeschätzt haben, so dass einige Elemente auf dem realen Endgerät (aufgrund der Bildschirmgröße und Auflösung) nicht mehr lesbar waren.

Anwendbarkeit von bestehenden Usability-Methoden: Wir haben festgestellt, dass die Co-Discovery Methode besonders gut für das toolbasierte Vorgehensmodell geeignet ist, da es eine natürliche Kommunikationssituation erzeugt. Dennoch müssen Alternativen untersucht werden, um psychologische Einflüsse zu verhindern, wenn Methoden wie zum Beispiel Thinking-Aloud Protocol verwendet werden sollen. Zudem ergab die Evaluation, dass weder ein Methodenmix erforderlich ist, noch die Anwendbarkeit vorhandener Methoden beeinträchtigt wird. Ein Kompensieren etwaiger Schwachpunkte, wie es in anderen Ansätzen der Fall ist, ist daher nicht erforderlich.

Korrelation zwischen Usability-Testing und Rapid-Prototyping: Ein wichtiger Diskussionspunkt der Experteninterviews war der Zusammenhang zwischen Prototyping und Usability-Testing. Auch wenn in kleinen und mittelständischen Unternehmen das Prototyping und Usability-Testing aufgrund des geringen Budgets und des enormen Zeitdrucks kaum angewendet wird, wurde dieser Aspekt von allen Experten stark betont. Die Experteninterviews haben gezeigt, dass

es von Vorteil sein kann, frühzeitig mit dem Prototyping anzufangen und gewisse Funktionen, Interaktionskonzepte und Gestaltungen zu testen, bevor eine kostenintensive Implementierung stattfindet. Durch Ansätze wie PhoneGap könnte das Prototyping noch effektiver gestaltet werden, da einzelne oder komplette Teile der Prototypen in die Anwendung fließen könnten.

5. Fazit und Ausblick

Durch den Ansatz des toolbasierten Vorgehensmodells wurde die Realisierung eines Instruments bewiesen, dass auf der einen Seite Rapid-Prototyping und Usability-Testing kombiniert und zum anderen, dass es technisch möglich ist, ein Instrument zu entwickeln, dass nicht nur die Haptik des Mobilen System unangetastet lässt, sondern auch das Testen im originären Benutzungskontext ermöglicht. Durch die Kombination mit dem Phasenmodell wurde der Weg für das frühzeitige Testen bereitet, so dass Ideen und Kreativität nicht der „Schere im Kopf" zum Opfer fallen müssen. Gerade diese Tatsache wird in Zukunft eine entscheidende Rolle spielen, da es durch die hohen Anforderungen der Gebraucher und dem enormen Zeit- und Kostendruck auf innovative Hilfsmittel ankommt.

Die Verbindung von Prototyping- und Usability-Testing und deren Integration in einen übergangslosen Entwicklungsprozess kann nur den Anfang darstellen. Dieser Ansatz stellt einen entscheidenden Beitrag für das Mobile Usability-Testing und die zukünftige Entwicklung Mobiler Systeme im Allgemeinen dar und zeigt vielversprechende Möglichkeiten für einen Paradigmenwechsel innerhalb dieses Forschungsfeldes.

In unserer zukünftigen Forschung werden wir uns auf die Entwicklung des ripcord Systems für andere Betriebssysteme (wie z. B. iOS und Android) konzentrieren. Wir werden das toolbasierte Vorgehensmodell in realen Projekten einsetzen, weiter evaluieren und optimieren.

Literatur

1. Bernhaupt, R., Mihalic, K., und Obrist, M. Usability Evaluation Methods for Mobile Applications. In: Lumsden, J. (Hrsg.) Handbook of Research on User Interface Design and Evaluation for Mobile Technology, IGI Global, S. 745-758. 2008.
2. Ballard, B. Designing the Mobile User Experience. Wiley, 2007.
3. Dahl, Y., Alsos, O. A. und Svanas D. Evaluating Mobile Usability: The Role of Fidelity in Full-Scale Laboratory Simulations with Mobile ICT for Hospitals. In: Jacko, J. A. (Hrsg.) Human-Computer Interaction, Part 1, HCII 2009, LNCS 5610, S. 232-241, Springer Verlag Berlin Heidelberg. 2009.
4. Holtz-Betiol, A. und de-Abreu-Cybis, W. Usability-Testing of Mobile Devices: A Comparison of Three Approaches. In: Proceedings of INTERACT 2005, LNCS 3585. Springer Berlin/Heidelberg, S. 470-481. 2005.
5. Kiljander, H. Evolution and Usability of Mobile Phone Interaction Styles. Helsinki University of Technology. Publications in Telecommunications Software and Multimedia. Otamedia Oy, 2004.
6. Kjeldskov, J. und Stage, J. New Techniques for Usability Evaluation of Mobile Systems. In: International Journal of Human-Computer Studies (IJHCS), Vol. 60, S. 599-620. 2004.
7. Krannich, D. "Mobile Usability-Testing: Ein toolbaisertes Vorgehensmodell zum Rapid-Prototyping und Usability-Testing von Mobilen Systemen im orgininären Benutzungskontext". Dissertation, E-LIB Universität Bremen, 2010.
8. Rantanen J., Impio J., Karinsalo T., Reho A., Tasanen M. und Vanhala J. Smart Clothing Prototype for the Artic Environment. In: Personal and Ubiquitous Computing, 6, S. 3-16. 2002.
9. Roto, V., Oulasvirta, A., Haikarainene, T., Kuorelahti, J., Lehmuskallio, H. und Nyyssönen, T. Examining Mobile Phone Use in the Wild with Quasi-Experimentation, Helsinki Institute for Information Technology, August 2004. 2004.
10. Weiss, S. Handheld Usability. Wiley & Sons, 2002.