Methoden für Usability-Evaluationen auf Basis von Aufgabenmodellen

Gregor Buchholz
Institut für Informatik
Universität Rostock
Albert-Einstein-Straße 21
18051 Rostock
grbuc@informatik.uni-rostock.de

Stefan Propp
Institut für Informatik
Universität Rostock
Albert-Einstein-Straße 21
18051 Rostock
stefan.propp@informatik.uni-rostock.de

Peter Forbrig
Institut für Informatik
Universität Rostock
Albert-Einstein-Straße 21
18051 Rostock
pforbrig@informatik.uni-rostock.de

Abstract

Die erfolgreiche Entwicklung interaktiver Systeme setzt die Einbeziehung des Nutzers in alle Phasen der Entwicklung voraus. Ausgehend von einem eigenen Ansatz zur modellbasierten Softwareentwicklung und dem Werkzeug ReModEl (Remote Model based Usability Evaluation) stellt dieser Beitrag Methoden für nutzerzentriertes Design und Usability-Tests vor, die sich auf die verwendeten Modelle beziehen. Ziel dieser Methoden ist es, die Modelle zu evaluieren, Vorschläge zur Umsetzung der Modelle in abstrakte Oberflächen abzuleiten und konkrete Oberflächen auf ihre Effizienz zu testen.

Keywords

Usability-Tests, Modellbasierte Softwareentwicklung, Usability-Methoden

1.0 Einleitung

Mit den ständig zunehmenden Fähigkeiten mobiler Geräte, der Vielfältigkeit ihrer spezifischen Charakteristika in Hinblick auf Interaktions- und Leistungsfähigkeit sowie der stetig wachsenden Komplexität und Verbreitung interaktiver Systeme wachsen auch die Ansprüche an eine effiziente, den Nutzer, seine Bedürfnisse und Anforderungen einbeziehende Softwareentwicklung. Modellbasierte Ansätze unterstützen den Entwickler dabei, den Kontext des Nutzers, sein Arbeitsumfeld, die Arbeitsgegenstände, die Werkzeuge, seine Tätigkeiten und seine Kooperationen mit anderen Nutzern zu erfassen, miteinander in Beziehung zu setzen und diese Modelle als Ausgangspunkt für die Softwareentwicklung und insbesondere für die Oberflächenentwicklung zu nutzen.

2.0 Modellbasierte Softwareentwicklung

Der an der Universität Rostock entwickelte Ansatz zur Unterstützung des model based designs (MBD) stellt den Anspruch in den Mittelpunkt, dass alle Entwicklungsprozesse auf den gleichen Modellen basieren und als eine Sequenz von Transformationen von zunächst abstrakten Modellen hin zu detaillierten Spezifikationen von interaktiven Systemen und insbesondere deren Nutzungsschnittstelle zu sehen sind. Grundlage und Mittel für diesen Prozess bilden das Aufgabenmodell zur hierarchischen Dekomposition der mit dem System ausführbaren Aktivitäten und deren wechselseitigen Beziehungen, das Objektmodell für die Arbeitsgegenstände und Werkzeuge, das Nutzermodell zur Beschreibung der Anwender und -gruppen (Rollen) sowie das Umgebungsmodell. Dem Aufgabenmodell als zentralem Bestandteil kommt dabei eine besondere Bedeutung zu: Die darin beschriebenen vom Anwender auszuführenden Tätigkeiten werden im Entwicklungsprozess durch abstrakte UI-Elemente im Dialogmodell realisiert, die mit Unterstützung von HCI-Patterns bis hin zu konkreten UIs weiterentwickelt werden (Forbrig et al. 2006). Diese Umformungen können nicht vollautomatisch geschehen; das Entwerfen eines User Interfaces ist ein kreativer Prozess, der der maßgeblichen Beteiligung von Interface-Designern bedarf. Es wurden jedoch eine Reihe von Tools erstellt, größtenteils als Eclipse-Plugins, die den Entwickler bestmöglich unterstützen: Angefangen von der Erstellung der Modelle und deren Simulation über die patterngestützte Verfeinerung der im Dialoggraphen repräsentierten abstrakten Oberfläche bis hin zur Generierung konkreter UIs unter Einbeziehung der Spezifikation des Zielgerätes bilden sie eine durchgehende Kette zur modellbasierten Entwicklung.

Dieser Beitrag befasst sich mit Methoden, mittels derer die Modelle und deren Umsetzung evaluiert werden können. Die Software ReModEl stellt dafür den Rahmen zur Verfügung und erlaubt die Evaluation der Modelle, insbesondere des Aufgabenmodells, und deren Umsetzung in Benutzungsoberflächen.

3.0 ReModEl

Das System ReModEl ist als Client-Server-Architektur implementiert und unterstützt sowohl die Simulation von Aufgabenmodellen durch mehrere Be-

nutzer als auch das Testen von Prototypen und fertigen Systemen, die mit den oben vorgestellten Tools erstellt wurde. Die Client-Software enthält ein Administrationsmodul zur Verwaltung der Projekte auf dem Server, die jeweils ein Aufgabenmodell (womöglich in mehreren Versionen), aufgezeichnete Protokolle von Testdurchläufen und weitere Artefakte zur Testdurchführung (Aufgabenstellungen für den Nutzer, Notizen, Verweise auf andere Tests) zusammenfassen. Der Server nutzt zur Simulation von Aufgabenmodellen einen Interpreter, der Kommandos zur Ausführung von Aufgaben entgegennimmt und die aus den im Aufgabenmodell spezifizierten temporalen Abhängigkeiten zwischen den Aufgaben sowie deren Vor- und Nachbedingungen mit Bezug auf das Objektmodell sich neu ergebenden Zustände der Aufgabeninstanzen (enabled, running, done, finished, skipped) ermittelt und alle Änderungen als Test-Protokoll im XML-Format speichert. Im Gegensatz dazu werden diese Statusänderungen, die den Fortschritt des Anwenders bei der Nutzung der Applikation bzw. der Simulation des Modells wiedergeben, bei Tests von Prototypen und fertigen Anwendungen aus der zu testenden Software gewonnen und an den Server übermittelt. Das Client-Modul „Wrapper“, das diese Funktionalität bereitstellt, kann auch in einem unsichtbaren Modus arbeiten, um die potentielle Beeinflussung des Nutzers möglichst gering zu halten.

4.0 Usability-Analysen mit ReModEl

Mit den hier vorgestellten Methoden soll das Spektrum der untersuchbaren Problemfelder auf strukturelle Belange des Systems erweitert bzw. in diesem Bereich verbessert werden. Eine Kombination mit anderen Techniken ist in jedem Fall eine gute Wahl: Die Verknüpfung von modellbezogener Analyse mit UI-Tests, bei denen es um ganz konkrete Oberflächenelemente (Fenster, Menüs, Dialoge, Buttons etc) geht, ist Gegenstand weiterer Entwicklungen. Ein guter Grundstein dafür ist mit der patternbasierten sukzessiven Verfeinerung der Oberfläche vom Dialoggraphen über weiter ausgestaltete Prototypen bis hin zum fertigen Interface gelegt. Die von ReModEl angebotenen Verfahren lassen sich in Analysen nach einem Testlauf (different-time) und währenddessen (same-time) unterteilen.

4.1 Different-time

Im Folgenden erläutern wir die Nutzung der Protokolle eines oder mehrerer Testläufe für Usability-Tests und nutzerzentrierte UI-Design-Entwicklung. Dazu werden zunächst zwei Aufbereitungsvarianten der Protokolldaten vorgestellt und im Anschluss deren Verwendung beschrieben.

Aus dem Protokoll eines Testlaufes oder aus den Protokollen mehrerer Testläufe ermittelt ReModEl die Ausführungshäufigkeiten und die Zeitspannen der Bearbeitung jeder Aufgabe. Diese Daten sind Grundlage für eine zu Heatmaps analoge Einfärbung der Blätter des Aufgabenbaumes, wobei die Visualisierung der Arbeitsdauer und die der Ausführungshäufigkeit zur Auswahl stehen. Die Werte für innere Knoten ergeben sich aus der Summe der Daten ihrer Kindknoten. So können Aufgaben identifiziert werden, in denen besonders häufig oder besonders lange gearbeitet wurde. Per Auswahl einer konkreten Aufgabe werden die dazugehörigen Daten im Detail angezeigt, beispielsweise die Verteilung der Ausführungshäufigkeit der Aufgabe über mehrere Testläufe. Die Gegenüberstellung von Protokoll-Darstellungen verschiedener Anwender kann helfen, anhand der tatsächlich ausgeführten Tätigkeiten Rollen und Verantwortlichkeiten zu identifizieren und bei der weiteren System- und UI-Gestaltung zu berücksichtigen. Durch Anwendung von Filterregeln kann die Darstellung auf Aufgaben reduziert werden, die den Filterkriterien entsprechen.

In einem zweiten Schritt werden die Übergangshäufigkeiten zwischen den einzelnen Blattaufgaben ermittelt und als Verbindungen zwischen den Aufgaben im Aufgabenbaum dargestellt. Die resultierende Visualisierung ist in Abbildung 1 dargestellt. Eine Verbindungslinie zwischen zwei Aufgaben kodiert dabei die Wechsel in beide Richtungen: Die Stärke und Farbe der zu einer Aufgabe gerichteten Linienhälfte verdeutlichen die Häufigkeit der Wechsel zu dieser Aufgabe.

Die Darstellung aller Übergänge kann schnell unübersichtlich werden und dadurch an Aussagekraft verlieren. Zwei Ansätze sollen dies vermeiden helfen: Zum einen ist die Menge der dargestellten Knoten veränderbar, d.h. Teilbäume können ausgeblendet werden, und jeder innere Knoten ist zuklappbar, so dass ein- und ausgehende Linien seiner

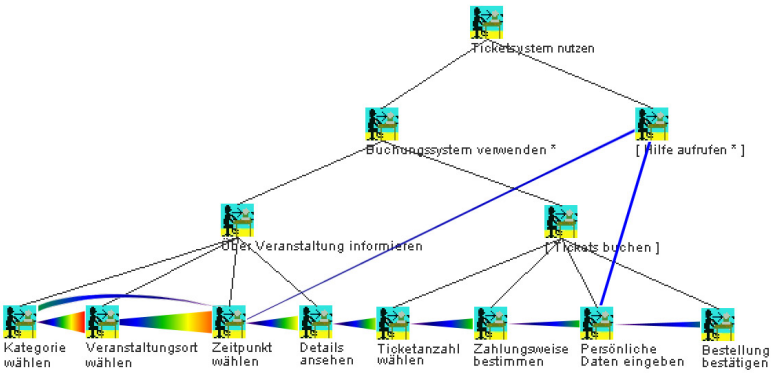

Abb 1: Visualisierung der Übergangshäufigkeit

Kindknoten nun im übergeordneten Knoten zusammengefasst sind. Dadurch können Wechsel zwischen Aufgabenbereichen besser erkannt und in die UI-Gestaltung einbezogen werden. Zum anderen ermöglichen Filtereinstellungen eine Selektion der darzustellenden Übergangslinien; per Angabe von unterer und oberer Grenze für die Anzahl der Wechsel ist die Anzeige auf besonders häufige oder besonders seltene Wechsel reduzierbar. Weiterhin können Wechsel zwischen Blättern mit gemeinsamem Elternknoten aus der Darstellung entfernt werden.

Die allein grafische Anzeige der Aufgabenhäufigkeit, -dauer und Übergänge ist für eine detaillierte Untersuchung nicht ausreichend, da eine Vermittlung der konkreten Werte durch Farben und Linienstärken nur Anhaltspunkte und keine exakte Information liefern kann. Jede Visualisierung wird daher durch ein Datenpanel unterstützt, in dem die jeweils zugrunde liegenden Daten gezeigt werden. Für die Häufigkeit und Dauer der Aufgabenausführungen ist dies eine Tabelle, in der die Anzahl bzw. Zeit pro Aufgabe für jeden Testlauf und die Summe über alle betrachteten Durchläufe angegeben sind.

Für die Wechsel zwischen den Aufgaben ergibt sich eine Tabelle, in der die Spalten mit den Namen von Aufgaben bezeichnet sind, deren ausgehende Wechsel in Richtung derjenigen Aufgaben, die den Zeilen der Tabelle vorangestellt sind, in der entsprechenden Zelle notiert ist. Die Auswahl der betrachteten Aufgaben ist dabei frei konfigurierbar.

Die horizontal und vertikal angeordneten Aufgaben müssen nicht notwendigerweise identisch sein. Die Auswahl kann jeweils separat erfolgen. So lassen sich die Übergänge zwischen Aufgaben und Aufgabenbereichen (Teilbäumen des Modells) gezielt untersuchen.

4.2 Analyse

Die Analyse der dargestellten Heatmaps und der Übergangsmatrix eröffnet vielfältige Möglichkeiten, die UI-Gestaltung anzupassen und damit die Gebrauchstauglichkeit zu steigern.

Eine Möglichkeit ist die Analyse nach Aufgaben, zwischen denen häufig hin- und hergewechselt wurde. Grafisch in der Heatmap sind diese durch beidseitige starke Verbindungslinien zu erkennen. Diese Aufgaben sollten bei ausreichend großer inhaltlicher Nähe in gleichen Containern untergebracht sein, also je nach Oberflächentechnik bspw. in den gleichen Dialogen oder Fenstern. Ist diese Nähe nicht gegeben, sollte zumindest ein schneller Wechsel zwischen den Aufgaben begünstigt werden, bspw. durch Vermeidung separater modaler Fenster, die Anbringung von Navigationsschaltflächen oder eine räumliche Nähe von Menüpunkten im Hauptmenü. Durch die vorgeschlagenen Maßnahmen bleiben zwar die Anzahl der Wechsel im Aufgabenmodell gleich, aber die Transitionen in der Oberfläche können vom Nutzer schneller und bequemer vollzogen werden.

Eine Analyse von Übergängen im Aufgabenmodell kann zudem längere geradlinige Arbeitsabläufe identifizieren, wie bspw. das schrittweise Ausführen eines Bestellvorganges oder das Anlegen neuer Mandanten. Diese hochgradig linearen Vorgänge kennzeichnen sich typischerweise durch eine geringe Varianz und folgen einer klaren Struktur von Teilaufgaben. Erkennen lassen sich solche Ketten durch zusammenhängende, gerichtete Übergänge mit hoher Intensität. Es bietet sich häufig eine „Wizard"-Unterstützung an: Der Nutzer kann so zeitlich effizient auf dem Pfad geführt werden, dass er in seinem Arbeitsablauf optimal unterstützt wird. Aufgaben, die gelegentlich zusätzlich ausgeführt wurden, können als optional in den Wizard integriert werden.

Komplexere Zusammenhänge in den Interaktionspfaden kann eine Mustererkennung aufdecken. Dazu wird ein Algorithmus zur Mustererkennung auf die Sequenz der ausgeführten Aufgaben angewandt. Ein Algorithmus speziell für das Auswertung von Interaktionspfaden ist IPM („Interaction-Pattern Miner"), vorgestellt in (El-Ramly et al. 2002). In der Interaktion gefundene Muster können auf thematische Nähe von Oberflächenelementen hinweisen und entsprechend als Grundlage dienen, um entsprechende Oberflächenelemente von stark verknüpften Aufgaben näher zueinander anzuordnen und damit die Arbeit des Nutzers zu erleichtern.

Die oben beschriebenen Analysen können auch außerhalb von Usability-Tests genutzt werden. Bspw. bei Schulungen zu PC-Anwendungen können Aufgabenpfade aufgezeichnet werden, um Fehler zu erkennen, um daraus gezielt Hinweise und weitere Übungen abzuleiten. Gleichzeitig kann für die Teilnehmer der Erfolg der Übungen kontrolliert werden.

4.3 Same-time

Das aufgabenbasierte Usability-Werkzeug ReModEl erlaubt es, bereits während des laufenden Tests die Statusinformationen der Aufgabenausführung zu betrachten. Diese Daten kann der Usability-Experte auch von einem entfernten Ort, wie einem separaten Beobachtungsraum, aus verfolgen. Die Betrachtung der Interaktion des Nutzers als Pfad durch ein Aufgabenmodell bietet eine intuitiv verständliche Ansicht. Die Darstellung dieses Pfades kann in einer Baumansicht des Aufgabenmodells erfolgen oder in einem Aktivitätsdiagramm. Im Rahmen der Analyse ist dieser tatsächlich gewählte Pfad mit

dem erwarteten zu vergleichen. Auch hierbei ist Werkzeugunterstützung hilfreich.

Der erwartete Pfad des Nutzers durch das System wird vorab spezifiziert, damit während des Tests Abweichungen automatisch gemeldet werden können. Freiheitsgrade berücksichtigen dabei Alternativen für den Nutzer. Um einen Pfad festzulegen, wird zunächst ein Durchlauf eines Experten als Referenzpfad aufgezeichnet. Dieser wird anschließend angereichert mit Informationen zu tolerierten Abweichungen im Bezug auf erlaubte Zeitvorgaben und die Aufgabenreihenfolge. Zeitvorgaben können für den kompletten Pfad oder bestimmte Teilpfade festgelegt werden. Pausen sind durch den Nutzer anzuzeigen oder durch den Experten zu protokollieren, damit die Auswertung nicht verfälscht wird. Die Aufgabenreihenfolge ist zunächst eingeschränkt durch das Aufgabenmodell selbst, das in jedem Zustand des Modells die Ausführung einer Menge von Aufgaben erlaubt: das „enabled task set“ (Paterno 1999). Diese Aufgaben lassen sich im konkreten Usability-Test in drei Gruppen von Aufgaben unterscheiden:

- erwünschte Aufgaben führen in Richtung des Zieles
- unerwünschte Aufgaben führen auf einem großen Umweg zum Ziel oder gar wieder zurück und sind daher unbedingt zu vermeiden
- neutrale Aufgaben wirken sich nicht direkt auf das Erreichen des Zieles aus und sind bis zu einem festgelegten Umfang unproblematisch, wie z.B. das Aufrufen der Hilfe oder das Anzeigen einer Seitenansicht in einer Textverarbeitung.

Während des Tests werden die Interaktionen des Nutzers zeitgleich beim Experten angezeigt. Auf der Basis des erwarteten Nutzerverhaltens wird dieser in zwei Stufen über Abweichungen des tatsächlichen Verhaltens informiert:

- Hinweise signalisieren neutrale Handlungen, wenige davon sind unproblematisch
- Warnungen signalisieren unerwünschte Handlungen, bei denen der Experte das Handeln des betreffenden Nutzers genauer untersuchen sollte oder sofort eingreifen kann, da die Informationen zeitgleich zur Testzeit vorliegen.

5.0 Zusammenfassung

Mit den hier vorgestellten Methoden werden die in der modellbasierten Softwareentwicklung verwendeten Konzepte für Usability-Tests erschlossen. Softwareentwickler und Usability-Experten arbeiten auf einer gemeinsamen Grundlage, so dass die Rückführung der Ergebnisse von Usability-Tests in die weitere Entwicklung der Software optimal erfolgen kann. Die beschriebenen Ansätze und Methoden fördern insbesondere bei komplexen Systemen mit vielen beteiligten Nutzern eine frühzeitige Erkennung von strukturellen Fehlern und liefern schon in den ersten Phasen der Softwareentwicklung Vorschläge für eine anwenderfreundliche Oberflächengestaltung.

Besondere Vorteile bieten diese Methoden in Systemen, bei deren Entwicklung Geräteklassen mit unterschiedlichen Schnittstellenspezifikationen zu berücksichtigen sind, wie z.B. PDA, Laptop, PC oder Handy. Durch die Simulation von Aufgabenmodellen und Tests von abstrakten User Interfaces lassen sich Erkenntnisse gewinnen, die in das Design verschiedener konkreter Oberflächen einfließen können, und so den Gesamtaufwand für eine nutzerzentrierte Entwicklung komplexer interaktiver Systeme reduzieren helfen.

6.0 Literaturverzeichnis

Baber, C.; N. Stanton (1994): Task analysis for error identification: a methodology for designing error-tolerant consumer products. Ergonomics 37(11), S. 1923–1941.

El-Ramly, M.; Stroulia, E.; Sorenson, P. (2002): Recovering software requirements from system-user interaction traces. SEKE 2002, S. 447-454.

Fields, R. (2001): Analysis of erroneous actions in the design of critical systems, Submitted for the degree of Doctor of Philosophy, University of York.

Forbrig, P.; Wolff, A.; Dittmar, A.; Reichart, D (2006): Tool Support for an Evolutionary Design Process using XML and User-Interface Patterns, Proceedings CUSEC 2006, Montreal.

Malý, I; Slavík, P (2006): Towards Visual Analysis of Usability Test Logs Using Task Models; Workshop proceedings, TAMODIA 2006, Hasselt: Universiteit Hasselt, S. 25-32.

Paterno, F. (1999): Model-Based Design and Evaluation of interactive applications. Berlin: Springer.

Rouse,W. (1990): Designing for human error: concepts for error tolerant systems. In H. Bohher (Hrsg.): MANPRINT: An approach to systems integration. New York: Van Nostrand Reinhold, S. 237–256.