

REGELBASIERTE ENTWICKLUNG VON BARRIEREFREIEN UND PLATTFORMUNABHÄNGIGEN MOBILEN BENUTZEROBERFLÄCHEN

Helm E¹, Schuler A¹, Mayr H²

Kurzfassung

Die Erstellung mobiler Applikationen im Gesundheitsbereich stellt Entwickler vor Herausforderungen in den Bereichen Usability und Barrierefreiheit. Durch das Schaffen einer Abstraktionsschicht über verschiedenen Entwicklungsplattformen können Usability-Regeln im Vorfeld definiert und bereits zur Entwicklungszeit auf die Benutzeroberfläche angewandt werden. Zusätzlich erlaubt dieser Ansatz die Generierung von plattformspezifischen Code und somit die Entwicklung von barrierefreien, mobilen Benutzeroberflächen für mehrere Plattformen.

Abstract

The development of mobile applications in the healthcare sector bears special challenges for programmers regarding usability and accessibility. By providing an abstraction layer over different development platforms usability rules defined in advance can be applied to the the user interface during development time. In addition, this approach allows the generation of platform-specific code, and thus the development of accessible, mobile user interfaces for multiple platforms.

Keywords – *Barrierefreiheit, Usability, mHealth*

1. Einleitung

Mobile Endgeräte, wie Smartphones oder Tablets, erfreuen sich zunehmender Beliebtheit in der Gesellschaft. Durch die hohe Portabilität bei dauerhafter Konnektivität dieser Geräte ergibt sich eine Vielzahl möglicher Anwendungsgebiete, insbesondere auch im Gesundheitsbereich. Unser Forschungsprojekt PIN (Patientenzentriertes Integriertes Netzwerk) befasst sich mit (mobilen) e-Health-Anwendungen, unter anderem mit den Einsatzmöglichkeiten von Telemonitoring in der kardiologischen Rehabilitation. Konkret wird eine mobile Applikation auf Android-Smartphones eingesetzt, um Patienten, die am Rehabilitationsprogramm teilnehmen, aktiv zu begleiten. Der wesentliche Aufgabenbereich der Applikation besteht in der Erhebung und Weiterverarbeitung rehabilitationsrelevanter Vitaldaten. Aufgrund der Heterogenität der Anwender im Projekt PIN, vom technikaffinen Mittdreißiger, der selbst ein Smartphone besitzt, bis zur Generation 60+, die wenig mit moderner Technologie in Berührung kommt, ergeben sich besondere Anforderungen an

1 FH OÖ Forschungs & Entwicklungs GmbH, Österreich

2 FH OÖ Fakultät für Informatik, Kommunikation und Medien, Campus Hagenberg, Österreich

die Applikation und insbesondere an die Benutzerführung [1]. Neben der Gestaltung der grafischen Benutzerschnittstelle (große, leicht wählbare Elemente bei gleichzeitig relativ kleinen Bildschirmdiagonalen) spielt besonders die Führung des Arbeitsablaufs eine entscheidende Rolle für die Akzeptanz unseres Systems, da die Bediener ihre gewohnten Abläufe beibehalten wollen. Damit rückt der Bereich Usability und die damit verbundene Ausrichtung der Benutzerinteraktion an geltende Richtlinien stark in den Vordergrund.

Für die Schaffung von Anwendungen für Szenarien mit ähnlich heterogenem Benutzerkreis ist es wünschenswert, die Einhaltung bestimmter Usability-Richtlinien schon während der Entwicklung automatisch zu überprüfen. Dies erfordert die Möglichkeit, Regeln auf der Grundlage von Usability-Richtlinien festzulegen, die Restriktionen in Bezug auf eine konkrete Benutzerschnittstelle darstellen. Deutlich wird die Notwendigkeit entsprechender Regelfestlegung und -prüfung bei Benutzern mit bestimmten Einschränkungen. Betriebssysteme wie Apple iOS und Android bieten auf bestimmten Endgeräten bereits systemseitig eine Unterstützung barrierefreier Zugänge für Menschen mit Einschränkungen. Dies reicht von der Möglichkeit der Sprachsteuerung, bis hin zum Vorlesen von Benutzerschnittstellen [2,3]. Ein wesentliches Erfordernis in diesem Zusammenhang sind von den Herstellern definierte Regeln, deren Einhaltung die angeführten barrierefreien Zugänge zu mobilen Anwendungen erst ermöglicht. Nachfolgend präsentieren wir die von uns angewandten Methoden, sowie deren Umsetzung im *Rule-based Generation of Mobile User Interfaces (RUMO)* Framework, das derzeit im Rahmen des Forschungsprojektes PIN entwickelt wird.

2. Methoden

Es existieren bereits Ansätze zur plattformunabhängigen Erstellung von Apps [4, 5], von denen aber keiner die Einbindung von Regeln zur Entwicklungszeit unterstützt. Im Bereich der automatischen Evaluierung von Benutzerschnittstellen anhand definierbarer Regeln existieren eine Reihe unterschiedlicher Ansätze.

Dazu zählen beispielsweise die automatische Prüfung von Usability-Guidelines mittels der Client-Server Anwendung Sherlock [6]. Sherlock extrahiert einzelne Eigenschaften einer Benutzerschnittstelle, beispielsweise Farben, Texte, Positionen einzelner Komponenten, etc. und prüft anhand entsprechender Daten die Einhaltung definierbarer Regeln. Da Regeln nur durch Quellcodeerweiterungen ergänzt werden können, wird Sherlock auch zur Gruppe der statischen Analysewerkzeuge gezählt [7]. Im Vergleich funktioniert das Werkzeug REVISER mit einem regelbasierten System und das Hinzufügen neuer Guidelines gestaltet sich einfacher.

Ein Vertreter eines Werkzeuges, das speziell für die Prüfung von mobilen Anwendungen, im speziellen von Benutzerschnittstellen, konzipiert wurde, ist das in der Android Entwicklungsumgebung enthaltene Android Lint. Es verfügt über eine Reihe an definierten Regeln, die es ermöglichen, etwaige Fehler in Bezug auf die Kategorien Korrektheit, Sicherheit, Performanz, Usability, Barrierefreiheit und Lokalisierung festzustellen und in weiterer Folge zu vermeiden. Nachteilig ist allerdings die fehlende Möglichkeit neue Regeln zu ergänzen. Darüber hinaus beschränkt sich die Verwendung von Android Lint auf die Android Plattform [8].

Im Folgenden wird der eigene in RUMO verfolgte Ansatz und die notwendigen Schritte beschrieben, um spezielle Anforderungen an die Benutzeroberfläche in den Entwicklungsprozess einzubinden und darüber hinaus unterschiedliche mobile Plattformen zu unterstützen.

2. 1. Das plattformunabhängige Modell

Zuerst muss ein Modell geschaffen werden, das die Gemeinsamkeiten der verschiedenen Zielplattformen abbildet, ein sogenanntes PIM (Plattform Independent Model). Am Beispiel dreier großer Anbieter von mobilen Plattformen lassen sich schnell einige Ähnlichkeiten feststellen. Sowohl IOS von Apple, als auch Android von Google und Windows Phone von Microsoft abstrahieren die grafische Benutzeroberfläche von der dahinterliegenden Interaktions- und Businesslogik. Alle drei Anbieter setzen dabei auf XML-basierte Dateien und verfolgen für die Anordnung der Komponenten einen hierarchischen Ansatz. Die Konsequenz daraus ist, dass sich mobile Benutzeroberflächen generisch definieren lassen. Zwar sind in einem generalisierten Modell nicht alle Spezifika der einzelnen Plattformen abbildbar, für einfache Benutzeroberflächen, die aus klassischen Menüstrukturen bestehen reicht es dennoch.

2. 2. Definition und Prüfung der Regeln

Um die Anforderungen an die Benutzeroberfläche hinsichtlich Barrierefreiheit und Usability automatisiert überprüfen zu können, müssen Regeln definiert werden. Diese Regeln bestehen im einfachsten Fall aus einem wenn-dann-Konstrukt und müssen in einer konkreten Regelsprache, z.B. RuleML, formuliert werden [9]. Mit Hilfe einer Regelverarbeitungskomponente, die sowohl die Regelsprache als auch das PIM kennt, können die Regeln auf eine Instanz einer generischen mobilen Benutzeroberfläche angewandt werden.

2. 3. Modelltransformation

Der modellbasierte Ansatz erlaubt unter Verwendung moderner M2T (Model to Text) Technologien eine Transformation des generischen Benutzerschnittstellenmodells in konkrete Ausprägungen, wie z.B. Android oder IOS (siehe *Abbildung 1*).

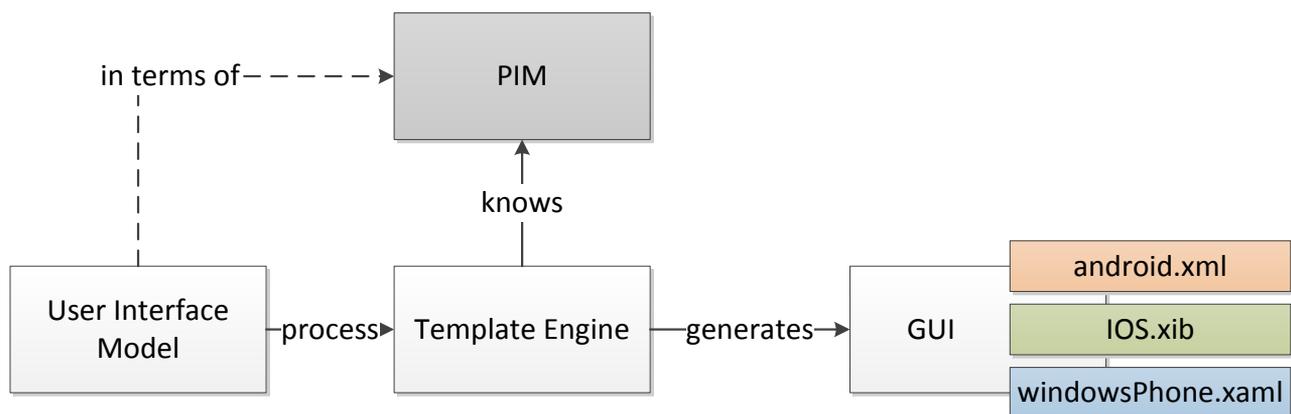


Abbildung 1: Ablauf der Modelltransformation

3. Ergebnisse

Das vorgestellte Konzept findet seine konkrete Umsetzung im Rahmen des RUMO-Frameworks. Das Framework setzt sich im Wesentlichen aus drei Komponenten zusammen. D.s.: das plattformunabhängige Benutzerschnittstellenmodell (PIM), die Regelverarbeitungskomponente (Rule Engine) und die Überführung definierter Modelle in eine beliebige Zielplattform.

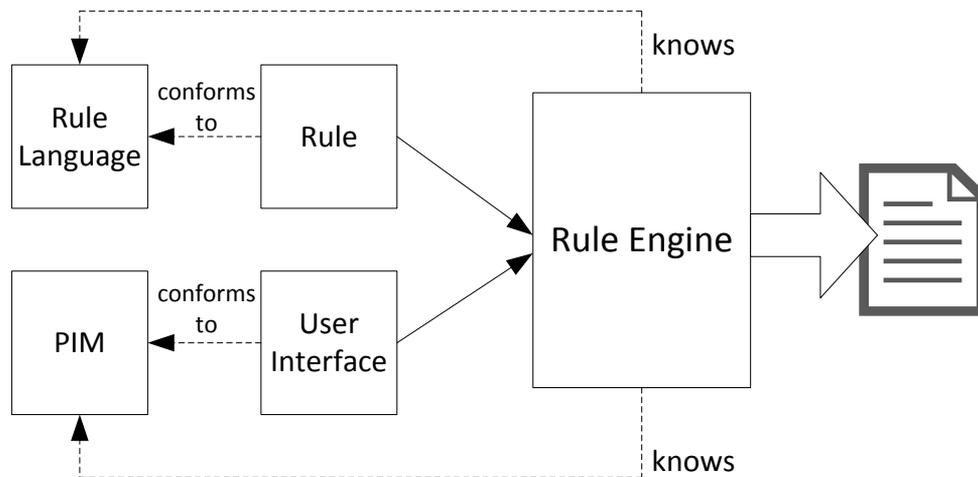


Abbildung 2: Architektur der Regelverarbeitungskomponente

Das Zusammenspiel bzw. die Verwendung des Frameworks kann anhand des folgenden Workflows beschrieben werden. Ausgangsbasis für die regelbasierte Erstellung bildet eine konkrete Modellinstanz als Ausprägung des Benutzerschnittstellenmodells. Entsprechende Modellinstanz in Verbindung mit der Festlegung einer Menge zu prüfender Regeln bildet den Input für die JSR-94-konforme Regelverarbeitungskomponente [10] (Abbildung 2). Ein Inferenzmechanismus schließt aufgrund der gegebenen Fakten in Form der Modellinstanz, welche der Regeln als erfüllt, bzw. nicht erfüllt gelten. Die Resultate der Regelprüfungen werden in einem Bericht zusammengefasst (siehe Abbildung 3). Den Abschluss bildet die Transformation der Modellinstanz in eine Benutzerschnittstelle einer beliebigen Zielplattform, wie beispielsweise Android oder IOS.

RUMO rule execution report

5/5 rules where executed, see details below

action	rule	content	result
✘	buttonColor	button background color should be set to #FFFFFF	✓
✘	buttonSize	buttons should be at least have a size of 32x32	✓
ℹ	labelSize	label fontSizes should be at least 12pt	✓
✘	componentDistance	the horizontal distance between (button1, button2) is to small	✓
⚠	buttonForeground	the foreground color for buttons should be #143A56	✓

Abbildung 3: Bericht über Resultate der Regelprüfungen

Das RUMO-Framework wird derzeit im Forschungsprojekt PIN mit einigen ausgewählten Testpersonen getestet. Eine Android-basierte Applikation zur Erhebung von Vitaldaten bildet den Ausgangspunkt für die entsprechende Evaluierung. Die wesentliche Fragestellung die im Rahmen der Evaluierung geklärt werden soll, ist, ob das vorgestellte Framework im praktischen Einsatz bestehen kann. Dies betrifft vor allem die Änderungen und Anpassungen in Bezug auf den verwendeten Softwareentwicklungsprozess, der gemäß ISO 9241-210 an den Usability Engineering

Lebenszyklus anzupassen ist [11]. In diesem Zusammenhang wird darüber hinaus der Fragestellung nachgegangen, welche benutzerseitigen Anforderungen sich durch Regeln im Rahmen des RUMO-Frameworks abbilden lassen.

4. Diskussion

Unser Ansatz und der darauf basierende, von uns entwickelte Prototyp zeigen, dass für mobile Anwendungen, die für breite Benutzerkreise gedacht sind (insbesondere bei verantwortungsvollen Einsatzgebieten wie z.B. der Vitaldatenerfassung in der Rehabilitation), ein flexibles Regelbasiertes Interaktionsmodell geeignet ist. Unser Ansatz bietet die Möglichkeit, die Regelbasis um ELGA-Usability-Guidelines zu erweitern, sobald diese entsprechend formuliert worden sind (geplant für 2013).

Derzeit können mit den von uns vorgestellten Regeln nur einfache Komponenten wie Buttongröße, Farbkombinationen, Schriftgröße, Computerlesbarkeit (großer Anfangsbuchstabe für richtige Betonung) etc. abgebildet werden. Unser Regelmodell bietet noch keine Möglichkeit, Interaktionslogik abzubilden. Einfache User-Interfaces (wie sie oft im Gesundheitsbereich, vor allem für die Endanwender gefordert sind), lassen sich exzellent konfigurierbar festlegen. Das generalisierte Modell enthält außerdem keine Plattformspezifika, was eine saubere Schichtentrennung und leichte Erweiterbarkeit auf (später geplante) weitere Plattformen (z.B. Windows Phone 8) ermöglicht. Nach unserem Wissensstand ist der von uns vorgestellte Ansatz der erste modellbasierte Ansatz für Plattformunabhängigkeit im Bereich mobiler e-Health-Anwendungen. Bei der Definition der Regeln muss allerdings die Struktur des Plattform-independent Model (PIM) bekannt sein und es ist nicht möglich, die Regeldefinition auszulagern.

Unsere nächsten Schritte umfassen intensive Tests mit einem wesentlich breiteren Feld an Testanwendern, die das gesamte Spektrum der Zielgruppe abdecken, sowie Untersuchungen, wie (zumindest einfache) Informationsabläufe im Modell abgebildet werden können, um diese ebenfalls plattformunabhängig modellieren zu können.

5. Referenzen

- [1] Juha Häikiö, Arto Wallin, Minna Isomursu, Heikki Ailisto, Tapio Matinmikko, and Tua Huomo. Touch-based user interface for elderly users. Proceedings of the 9th international conference on Human computer interaction with mobile devices and services (MobileHCI '07) 2007. ACM, New York, NY, USA, 289-296.
- [2] Apple Inc. Accessibility Programming Guide for iOS, http://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/iPhoneAccessibility/Accessibility_on_iPhone/Accessibility_on_iPhone.html, 2012
- [3] Google Inc. Accessibility Guide for Android. <http://developer.android.com/guide/topics/ui/accessibility/index.html>. 2012.
- [4] Puder Arno. An XML-Based Cross-Language Framework. Springer Berlin Heidelberg. Proceedings of the On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops.
- [5] Xamarin, Build cross-platform iOS, Android and Mac apps with C# and .NET, <http://xamarin.com/about>, 2012
- [7] D. Grammenos, D. Akoumianakis, und C. Stephanidis. Integrated Support for working with Guidelines: The Sherlock Guideline Management System. Interacting with Computers, 12(3):281–311, 2000.

[6] N. Hamacher und K.-F. Kraiss. Expertensystem zur kriterienorientierten Bewertung der Gebrauchsfähigkeit von Dialogsystemen. In VDE-Kongress „Innovationen für den Menschen“, Volume 1, Seiten 193–198, Berlin, November 2004. VDE, VDE Verlag GmbH.

[8] I. Android. Android lint: Improving your code with lint, <http://developer.android.com/tools/debugging/improving-w-lint.html>, 2013

[9] Harold Boley, Tara Athan, Adrian Paschke, Said Tabet, Benjamin Gro-soff, Nick Bassiliades, et. al. Schema Specification of Deliberation RuleML. Version 1.0. April 2012.

[10] Java Community Process, “Java Rule Engine API JSR-94”, 2003.

[11] ISO. ISO 9241-210:2010 - Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems. 2010.

Corresponding Author

Emmanuel Helm
FH OÖ Forschungs & Entwicklungs GmbH
Softwarepark 13, 4232 Hagenberg, Austria
Email: emmanuel.helm@fh-hagenberg.at