Konzeption modellbasierter Benutzungsschnittstellen für verteilte Selbstbedienungssysteme

Conference Paper · January 2013

CITATIONS
0

READS
37

3 authors:

Enes Yigitbas
Universität Paderborn
16 PUBLICATIONS **27** CITATIONS
SEE PROFILE

Christian Gerth
Hochschule Osnabrück
43 PUBLICATIONS **660** CITATIONS
SEE PROFILE

Stefan Sauer
Universität Paderborn
100 PUBLICATIONS **736** CITATIONS
SEE PROFILE

Konzeption modellbasierter Benutzungsschnittstellen für verteilte Selbstbedienungssysteme

Enes Yigitbas, Christian Gerth, Stefan Sauer

Universität Paderborn
s-lab – Software Quality Lab
Zukunftsmeile 1
33102 Paderborn
{eyigitbas | gerth | sauer}@s-lab.upb.de

Abstract: Selbstbedienungssysteme sind komplexe technische Systeme und stellen Produkte und Dienstleistungen für den Endbenutzer zur Verfügung. In Anbetracht der heterogenen Nutzerlandschaft solcher Systeme ist die einfache Bedienbarkeit der Benutzungsschnittstellen von hoher Bedeutung. Die Benutzungsschnittstellen müssen sich an die vielfältigen Rahmenbedingungen (z.B. Menschen mit Behinderungen, demografische Verteilung, usw.) anpassen lassen bzw. selbst adaptieren. Aufgrund der monolithisch geprägten Systemarchitektur von existierenden Selbstbedienungssystemen ist eine einfache und flexible Bedienbarkeit der Benutzungsschnittstellen oft nur eingeschränkt möglich. Die Entwicklung adaptiver Benutzungsschnittstellen beinhaltet Herausforderungen für die Entwickler, die von Frameworks wie dem CAMELEON Reference Framework (CRF) adressiert werden. Allerdings fehlen konkrete Ansätze zur Unterstützung der Entwicklung von flexiblen und adaptiven Benutzungsschnittstellen für verteilte Selbstbedienungssysteme. In diesem Beitrag beschreiben wir einen modellbasierten Ansatz für die Entwicklung von Benutzungsschnittstellen, der Aspekte der Adaption in den modellbasierten Entwicklungsprozess des CRF integriert. Zur Veranschaulichung erläutern wir unsere Motivation und Lösungsideen mittels eines Fallbeispiels.

1 Einleitung und Motivation

Die Einsatzbereiche von Selbstbedienungssystemen erstrecken sich heutzutage über ein breites Dienstleistungsspektrum von Unterhaltung, Information, Verwaltung, Geld- und Zahlungswesen sowie Personenbeförderung oder Versand. Zunehmende Mobilität und Schnelllebigkeit steigern den Bedarf nach flexibel erreichbaren Produkten und Dienstleistungen. Gleichzeitig muss eine einfache Bedienbarkeit für ein breites Spektrum an Menschen mit unterschiedlichen Vorkenntnissen und Bedürfnissen ermöglicht werden, damit eine hohe Kundenakzeptanz sichergestellt wird. Diese heterogene Nutzerlandschaft erfordert eine besondere Berücksichtigung der Entwicklung und Nutzung von Benutzungsschnittstellen und ihrer Bedienbarkeit.

Aufgrund der stark monolithisch geprägten Systemarchitektur von existierenden Selbstbedienungssystemen werden die Nutzererwartungen hinsichtlich Flexibilität und Einfachheit der Bedienung nur eingeschränkt erfüllt. Dies liegt im Wesentlichen daran, dass

die jeweiligen Benutzungsschnittstellen speziell auf die Systemarchitektur und die genutzten Funktionen angepasst sind. Folglich steht die Funktionalität im Mittelpunkt und nicht der Benutzer des Selbstbedienungssystems. Zu beobachten ist auch, dass solche monolithischen Selbstbedienungssysteme von modernen Technologien im Bereich der Hardware- und Softwaretechnologie (wie z.B. Multi-Touch, optische Verfahren, Sprachsteuerung) nicht ausreichend profitieren.

Angesichts dieser Schwächen von monolithischen Selbstbedienungssytemen entsteht der Bedarf an einer neuen Generation von Selbstbedienungssystemen, die eine verbesserte Bedienbarkeit durch flexible und adaptive Benutzungsschnittstellen bieten, um eine hohe Kundenakzeptanz für ein breites Spektrum an Endbenutzern zu ermöglichen. Die Bedienbarkeit existierender Selbstbedienungssysteme soll außerdem durch die Einbettung von neuen Technologien und die Nutzung ihrer Vorteile erleichtert werden, indem der Vertrieb von Produkten und Dienstleistungen an Automaten um neue Kanäle erweitert wird. Als Kanäle bezeichnet man dabei die verschiedenen Arten des Zugangs zu einem Dienst und damit verbunden unterschiedliche Formen der Interaktionsschnittstellen, wie z.B. Steuerelemente eines Automaten, Webbrowser oder Smartphones. Für die effiziente Entwicklung solcher adaptiver, interaktiver Selbstbedienungssysteme ist die Berücksichtigung der benötigten Benutzungsschnittstellen von essentieller Bedeutung.

Im Rahmen eines kooperativen Forschungsprojekts mit einem Industriepartner, der Selbstbedienungssysteme für unterschiedliche Branchen entwickelt, erarbeiten wir eine Methodik für die Entwicklung von Benutzungsschnittstellen für verteilte vernetzte Multikanalsysteme (VVMKS), wie sie in Abschnitt 2 genauer beschrieben sind. In existierenden Selbstbedienungssystemen ist eine flexible Erweiterung um neue Kanäle nur erschwert möglich, sodass die Nutzung derselben Funktion in verschiedenen Kanälen auf unterschiedlichen Plattformen oft nicht hinreichend unterstützt wird. Für den Wandel vom monolithischen Selbstbedienungssystem hin zum VVMKS ist daher die Entwicklung einfach und flexibel zu bedienender Benutzungsschnittstellen notwendig. Damit verbunden sind die Beherrschung der Komplexität, die sich durch die Verteilung der Benutzungsschnittstellen ergibt und die Steigerung der Effizienz bei der Multiplattformentwicklung.

Zur Erfüllung dieser Anforderungen verfolgen wir in unserer Methodik ein modellbasiertes Entwicklungsvorgehen basierend auf dem CAMELEON Reference Framework (CRF) [Ca03]. Das CRF ist eine etablierte Referenzarchitektur für die modellbasierte Entwicklung von Benutzungsschnittstellen. Diesen Ansatz erweitern wir um den Aspekt der Adaption zur Laufzeit, da existierende Ansätze dies bisher nicht vollständig abdecken. Im Rahmen unseres kooperativen Forschungsprojekts wollen wir die Entwicklung adaptiver interaktiver Selbstbedienungssysteme für eine Multiplattformumgebung ermöglichen, bei denen sowohl die Benutzungsschnittstelle als auch die Systemfunktionalität adaptiv sind. Hierzu haben wir wichtige Einflussfaktoren zur Umsetzung der beschriebenen Anforderungen auf die Modelle des CRFs abgebildet.

Im Folgenden gehen wir zunächst auf ein Fallbeispiel aus der Praxis ein, woraus wir die Herausforderungen und Anforderungen hinsichtlich der modellbasierten Entwicklung von interaktiven Selbstbedienungssystemen ableiten. Daraufhin folgt die Beschreibung

der Vorgehensweise zur Zielerreichung, in der wir den von uns verfolgten modellbasierten Ansatz erläutern. Anschließend werden verwandte Arbeiten vorgestellt und eine Zusammenfassung gegeben.

2 Fallbeispiel und Anforderungen

Im Rahmen eines Forschungsprojekts kooperieren wir mit einem Industriepartner, der sich auf die Entwicklung und den Vertrieb von Selbstbedienungssystemen spezialisiert hat. Dabei untersuchen wir die Entwicklung von flexiblen, grafischen Benutzungsschnittstellen, die sich über eine Vielzahl von Plattformen in einem vernetzten System erstrecken können. Der Wandel vom monolithischen Selbstbedienungssystem zum VVMKS soll durch einen modellbasierten Ansatz zur Entwicklung von Benutzungsschnittstellen für eine Multiplattformumgebung unterstützt werden.

Hierfür sind jedoch unterschiedliche Herausforderungen und Probleme zu bewältigen, die sich aus dem folgenden Beispielszenario aus dem Bereich des Fahrkartenverkaufs ableiten lassen (siehe Abbildung 1).

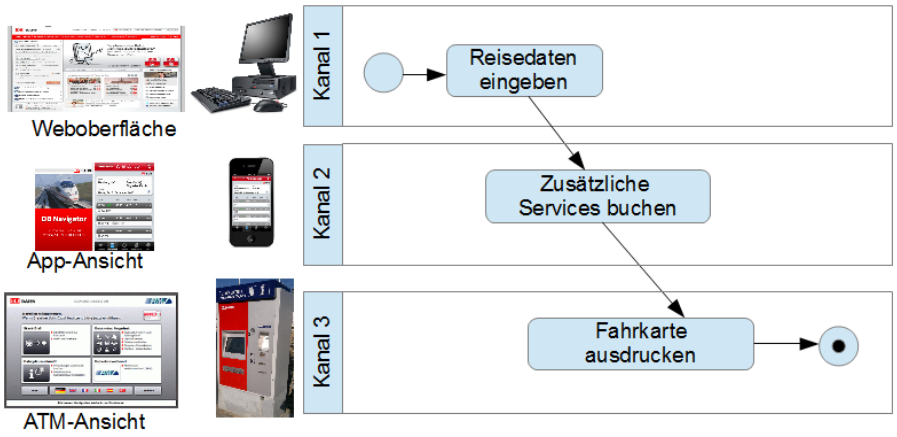

Abbildung 1: Beispielszenario Fahrkartenkauf

Das in der Abbildung 1 dargestellte Beispielszenario zeigt ein verteiltes und vernetztes Multikanalsystem, bei dem der Fahrkartenkauf durch den Einsatz von verschiedenen Kanälen durchgeführt wird, die über unterschiedliche Interaktionsschnittstellen verfügen. So wird der Fahrkartenkauf zunächst auf dem heimischen PC vorbereitet und begonnen, indem die Reisedaten (Datum bzw. Uhrzeit für Hin- und Rückfahrt) eingegeben werden. Anschließend können unterwegs auf dem Smartphone zusätzliche Services (Sitzplatz reservieren, Gepäckservice, Hotel am Reiseziel, …) dazu gebucht werden. Schließlich erfolgt der Ausdruck der Fahrkarte am Fahrkartenautomaten.

Betrachtet man das Beispiel des Fahrkartenverkaufs, so lassen sich unterschiedliche Problemstellungen und Herausforderungen für die Entwicklung der jeweiligen Benutzungsschnittstellen von verteilten Selbstbedienungssystemen ableiten:

1. Hohe Komplexität beherrschen

- Die Verteilung der Benutzungsschnittstelle und der clientseitigen Logik auf verschiedene, heterogene Endgeräte zur Unterstützung verteilter Geschäfts- und Interaktionsprozesse verändert die Architektur und die Komplexität der technischen Realisierung der Interaktion.

- Der Komplexitätsgrad erhöht sich zusätzlich durch unterschiedliche Interaktionsmodalitäten (Grafik, Sprache, Gesten).

2. Effizienz bei der Multiplattformentwicklung steigern

- Die Entwicklung einer Selbstbedienungsanwendung für heterogene Plattformen (iOS, Android, Windows Phone, usw.) erfordert ein dediziertes Vorgehen, um eine effiziente Entwicklung für multiple Plattformen zu gewährleisten.

- Es werden Techniken benötigt, um eine verteilte Selbstbedienungsanwendung effizient für unterschiedliche Plattformen zu entwickeln.

3. Systemfunktionalität und Benutzungsschnittstellen adaptieren

Um Funktionalität und Benutzungsschnittstelle anpassungsfähig zu gestalten, müssen verschiedene Arten von Adaptivität unterstützt werden:

- Anpassbarkeit an unterschiedliche Umgebungsbedingungen

- Anpassung von Benutzungsschnittstellen in Verbindung mit der Anpassung der Systemfunktionalität

- Anpassung nach Fähigkeiten des Benutzers oder nach seinen Bedürfnissen

- Verschiebung von Systemfunktionalitäten zwischen Komponenten des verteilten Systems

- geeignete Modelle zur Beschreibung der Adaption sowie der Anbindung der Funktionalität an die Benutzungsschnittstelle.

Auf Grundlage dieser Beobachtungen kann man zusammengefasst feststellen, dass es im Bereich der Selbstbedienungssysteme einen Bedarf nach einem integrierten modellbasierten Entwicklungsvorgehen gibt. Dieser sollte alle Aspekte eines Softwaresystems wie etwa Funktionalität, Benutzungsschnittstelle sowie Selbst-Adaptivität umfassen und einen für die Industrie geeigneten verzahnten Entwicklungsprozess ermöglichen.

3 Vorgehensweise zur Zielerreichung

Techniken der modellbasierten Entwicklung von Benutzungsschnittstellen wurden bereits in vielen Arbeiten erforscht (siehe Abschnitt 4). Im Rahmen unseres kooperativen Forschungsprojekts aus dem Industriekontext wollen wir eine effektive Methodik für die effiziente Entwicklung adaptiver, interaktiver Selbstbedienungssysteme für eine Multiplattformumgebung erarbeiten, bei denen sowohl die Benutzungsschnittstelle als auch die Systemfunktionalität anpassungsfähig ist.

Bevor wir auf die von uns angestrebte Zielarchitektur eingehen, wollen wir einen kurzen Überblick über das CAMELEON Reference Framework (CRF) geben, das im Rahmen des europäischen CAMELEON-Projekts (Context Aware Modelling for Enabling and Leveraging Effective interactiON) [Ca03] entstanden ist. Im Wesentlichen besteht dieses Framework aus vier verschiedenen Schichten (vgl. Abbildung 2).

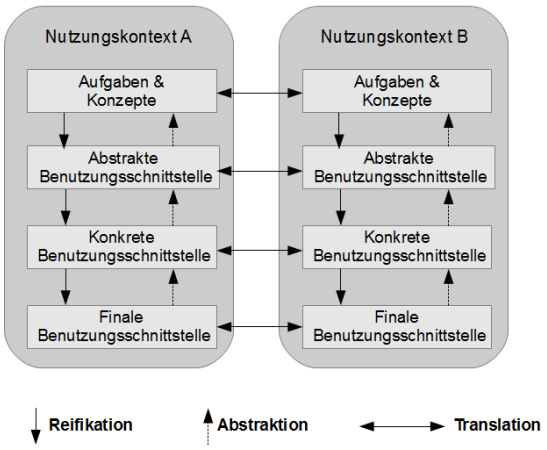

Abbildung 2: Vereinfachtes CAMELEON Reference Framework ([Me11], [Ca03])

Die oberste Schicht *Aufgaben & Konzepte* beinhaltet ein Aufgabenmodell, das zur hierarchischen Beschreibung der Tätigkeiten und Handlungen einzelner Benutzer der Benutzungsschnittstelle dient. Die *Abstrakte Benutzungsschnittstelle* wird in Form eines Dialogmodells beschrieben, das die Interaktionen des Benutzers mit der Benutzungsschnittstelle spezifiziert, ohne eine konkrete Technik zu berücksichtigen. Die eigentliche Darstellung der Benutzungsschnittstelle erfolgt durch die *Konkrete Benutzungsschnittstelle,* die durch ein Präsentationsmodell dargestellt wird. Die unterste Schicht des Frameworks ist die *Finale Benutzungsschnittstelle*, die schließlich für die Zielplattform eines bestimmten Endgerätes erzeugt wird. Die vertikale Dimension beschreibt den Weg von abstrakten zu konkreten Modellen. Hierbei wird ein Top-down-Ansatz verfolgt, bei dem die abstrakten Beschreibungen relevanter Informationen über die Benutzungsschnittstelle durch Modell-zu-Modell-Transformationen (M2M) zu verfeinerten Modellen angereichert werden. Anschließend werden die verfeinerten Modelle interpretiert

oder transformiert (Modell-zu-Code-Transformation, M2C), um die Finale Benutzungsschnittstelle zu erzeugen. Auf der horizontalen Ebene wird die Anpassungsmöglichkeit eines Modells abhängig vom gegenwärtigen Kontext dargestellt. Hierdurch wird abhängig vom Nutzungskontext eine Anpassung der Modelle zur Laufzeit ermöglicht (Kontextadaptivität).

Zur Untersuchung der Anpassungsmöglichkeiten haben wir eine Zielarchitektur entworfen, die Aspekte der Adaption besonders berücksichtigt und diese in den modellbasierten Entwicklungsprozess für Benutzungsschnittstellen integriert. Abbildung 3 zeigt die angestrebte Zielarchitektur unserer Lösung, die auf dem CAMELEON-Ansatz basiert. Auf der Ebene der konzeptionellen Modellierung (Computation-Independent Model, CIM) werden ein Aufgabenmodell und ein Nutzermodell (repräsentiert verschiedene Benutzergruppen oder Personen) erstellt, die als Eingabe für die Erstellung eines abstrakten Modells der Benutzungsschnittstelle auf der Ebene der plattformunabhängigen Modellierung (Platform-Independent Model, PIM) dienen.

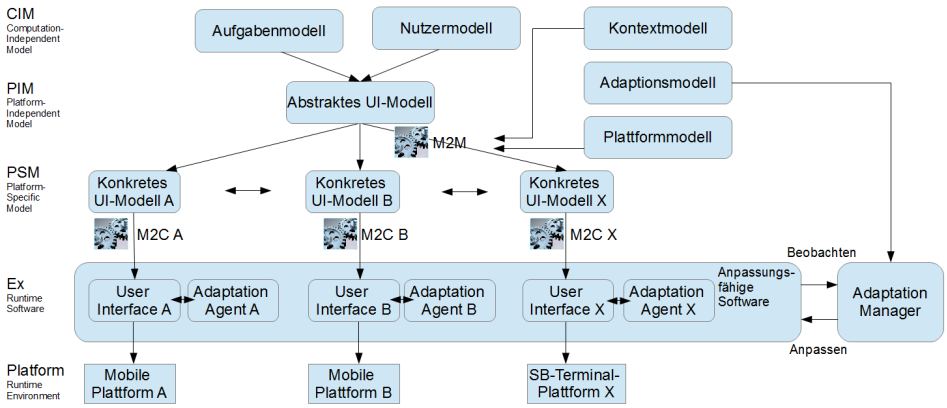

Abbildung 3: Schematische Darstellung zur Bearbeitung der Problemstellungen

Ein weiteres konzeptionelles Modell beschreibt den Nutzungskontext in Form von Kontextfaktoren, wie zum Beispiel die Lokalisation des Benutzers (Kontextmodell). Dieses Modell wird zusammen mit einem Modell der eingesetzten Plattformen verwendet, um die Modell-zu-Modell-Transformation vom abstrakten Benutzungsschnittstellenmodell (Abstraktes UI-Modell) in das konkrete Benutzungsschnittstellenmodell (Konkretes UI-Modell) auf der Ebene der plattformspezifischen Modellierung (Platform-Specific Model, PSM) zu realisieren. Dieser Übersetzungsschritt soll durch die Verwendung entsprechender Werkzeuge unterstützt werden. Das konkrete Benutzungsschnittstellenmodell besteht aus einer Menge gekoppelter Teilmodelle für die jeweiligen Plattformen. Mit Hilfe von spezifischen Generatoren und Interpretern wird dann aus dem jeweiligen Teilmodell die für eine konkrete Plattform erforderliche Benutzungsschnittstelle erzeugt, die dann auf dieser Plattform ausgeführt werden kann.

Dieses modellgetriebene Entwicklungsvorgehen soll auch die dynamische Anpassung der Benutzungsschnittstelle zur Laufzeit unterstützen. Hierzu wird neben einem Monito-

ring-Konzept ein Adaptionsmodell erstellt, das die Anpassung der Benutzungsschnittstelle (sowie der hiermit gekoppelten Funktionalität) beschreibt. Wir favorisieren hierfür die Nutzung von ECA-Regeln (Event-Condition-Action), die über ein Kontextmodell getypt sind, und die Verwendung von Adapt Cases [Lu11] zur Spezifikation des Anpassungsverhaltens der Benutzungsschnittstellen. Adapt Cases sind angelehnt an Use-Case-Diagramme aus der UML und ermöglichen eine intuitive Spezifikation von Selbst-Adaptivität. Mit dem Adapt-Case-Ansatz können zu überwachende Systeminformationen (Monitoring), relevante Kontextinformationen für die Adaption der Benutzungsschnittstelle und die jeweiligen Adaptionen einfach und präzise beschrieben werden. Aus den daraus resultierenden Adaptionsmodellen wird der Adaptation Manager abgeleitet. Dies ist eine Softwarekomponente, die die anpassungsfähige Software beobachtet und die Anpassung gemäß dem Adaptionsmodell steuert. Es ist angedacht, den Adaptation Manager durch dedizierte Subkomponenten bzw. Adaptation Agents zu ergänzen. Letztere interoperieren auf den jeweiligen Plattforminstanzen und sind für die Anpassung der Benutzungsschnittstelle (Interaktion) und der mit ihr gekoppelten Funktionalität auf der jeweiligen Plattforminstanz verantwortlich. Betrachtet man das Beispielszenario aus der Einleitung (siehe Abbildung 1), so gibt es jeweils unterschiedliche Adaptionskomponenten für die unterschiedlichen Endgeräte. Beispielsweise muss beim Übergang von Kanal 1 auf Kanal 2 die Adaptionskomponente des Smartphones in Kooperation mit dem Adaptation Manager dafür sorgen, dass die Funktionalität und Anzeige auf die spezifischen Eigenschaften und Anforderungen des Smartphones angepasst werden, sodass irrelevante oder unpassende Informationen der Weboberfläche (zusätzliche Angebote, Werbungen, Logos etc., die viel Platz in Anspruch nehmen) ausgeblendet werden.

Zur Erreichung der Zielsetzung haben wir zunächst heutige Selbstbedienungssysteme hinsichtlich ihrer Benutzungsschnittstellen und Architekturen untersucht. Aufbauend auf den Ergebnissen der Ist- und Anforderungsanalyse wird unter Beachtung des aktuellen Stands der Wissenschaft und Technik ein modelbasiertes Entwicklungsvorgehen entworfen, das die Entwicklung der verteilten Benutzungsschnittstellen moderner Selbstbedienungssysteme effizienter und verlässlicher machen soll. Zukünftige Arbeiten umfassen die Entwicklung einer Modellierungssprache zur Unterstützung der Modellierung von verteilten und adaptiven Benutzungsschnittstellen im Bereich der Selbstbedienungssysteme. Zur Unterstützung des modellbasierten Entwicklungsprozesses werden Softwarewerkzeuge benötigt, um die Entwicklungsaufgaben effektiv umzusetzen. Neben einem Editor zur Erstellung und Bearbeitung von Modellen in der spezifizierten Modellierungssprache werden insbesondere Werkzeuge für die Erstellung einer ausführbaren Benutzungsschnittstelle aus den konkreten Benutzungsschnittstellenmodellen benötigt. Hierbei ist insbesondere auch vor dem Hintergrund der angestrebten Laufzeitadaptivität zu entscheiden, ob ein Generator- oder Interpreteransatz verfolgt werden soll. Die entwickelte Lösung – modellbasierter Entwicklungsprozess, Architektur- und Bedienkonzept, Modellierungssprache und Werkzeugkette – wird schließlich anhand eines Beispielszenarios wie in Abbildung 1 in Form eines Demonstrators realisiert und evaluiert.

4 Verwandte Arbeiten

Modellbasierte Entwicklungsmethoden wurden in der Vergangenheit für verschiedene, einzelne Aspekte eines Softwaresystems und für verschiedene Einsatzdomänen diskutiert und eingeführt (vgl. [FR07]). Dies gilt etwa für die Entwicklung der Datenhaltungsschicht, der fachlichen Funktionalität bzw. für die Entwicklung einer Benutzungsschnittstelle [HMZ11]. Das CAMELEON Reference Framework (CRF) [Ca03] stellt einen vereinheitlichten Rahmen für die modellbasierte und modellgetriebene Entwicklung von Benutzungsschnittstellen bereit. Es gibt bereits unterschiedliche Arbeiten wie [Li08], [Kl11], [Bo06], die angelehnt an das CRF eine Umsetzung für die modellbasierte Entwicklung von Benutzungsschnittstellen vorgeschlagen. In diesen Ansätzen wird der Fokus auf die modellbasierte Entwicklungsvorgehensweise und ihre technologische Umsetzung gelegt. Hierbei werden jedoch Aspekte der Anpassungsfähigkeit bzw. Selbst-Adaptivität zur Steigerung der Flexibilität von Benutzungsschnittstellen für heterogene Benutzergruppen nicht ausreichend in den modellbasierten Entwicklungsprozess integriert.

Mit Selbst-Adaptivität wird der Prozess der automatischen Anpassung von Software als Reaktion auf sich ändernde Einflussfaktoren bezeichnet. Insbesondere soll dadurch der manuelle Wartungsaufwand der Software verringert werden.

Für die Spezifikation von Anforderungen zur Selbst-Adaptivität gibt es unterschiedliche Ansätze, die in adaptiven Systemen genutzt werden können. Whittle et al. unterstützen mit RELAX [Wh09] die Spezifikation von Adaptionsanforderungen durch Beschreibungsmöglichkeiten, mit Hilfe derer inhärente Eigenschaften der Adaptivität (z.B. Ungewissheit) gezielt berücksichtigt werden. Baresi und Pasquale [BP10] schlagen einen Ziel-basierten (goal-based) Modellierungsansatz zur Spezifikation von Anforderungen für adaptive Systeme vor. Eine andere Möglichkeit zur Modellierung von Selbst-Adaptivität wird durch die Modellierungssprache Adapt Cases [Lu11] zur Verfügung gestellt, die angelehnt an Use-Case-Diagramme aus der UML eine intuitive Spezifikation von Selbst-Adaptivität ermöglicht.

Existierende Architekturkonzepte für selbst-adaptive Systeme wie das MAPE-K von Kephart und Chess [KC03] und die 3-Schichten-Referenzarchitektur von Kramer und Magee [KM07] beschreiben die logischen Konzepte für die Implementierung adaptiver Software. Basierend auf dem MAPE-K-Ansatz existieren Frameworks wie Rainbow von Garlan et al. [GCS03] oder StarMX von Asadollahi et al. [AST09], die eine verfeinerte Architektur zur Implementierung selbst-adaptiver Systeme bereitstellen. Geihs et al. entwickeln im EU-Projekt MUSIC [Ge09] ein Framework für die Entwicklung selbstadaptiver und rekonfigurierbarer Software auf Basis einer verteilten Softwarearchitektur. Der Fokus liegt insbesondere auf der Erstellung von adaptiven, mobilen Anwendungen, die dem Nutzer ortsabhängige Dienste anbieten.

Weyns et al. führen FORMS [WMA10] ein, das ein formales Referenz-Modell für Selbst-Adaptivität darstellt. Dieser Ansatz beschreibt und verknüpft Konzepte der Selbst-Adaptivität in einem generischen Meta-Modell. FORMS bietet eine präzise Möglichkeit zur Modellierung von Selbstadaption auf einer formalen Basis. Auf der Model-

lierungsseite stellen Blair et al. [BBF09] mit Models@run-time eine Repräsentationsform vor, die Struktur, Verhalten und Ziele von Software aus unterschiedlichen Sichten des Problembereichs zur Laufzeit explizit (als Modelle) verfügbar macht.

In unserem Ansatz für die modellbasierte Entwicklung von Benutzungsschnittstellen verfolgen wir eine Entwicklungsmethodik, die alle Aspekte eines Softwaresystems wie etwa Funktionalität, Benutzungsschnittstelle sowie Selbst-Adaptivität integriert. Modellbasierte Ansätze aus dem Bereich der selbst-adaptiven Systeme sollen hierdurch mit Ideen aus dem Bereich der modellbasierten Entwicklung von Benutzungsschnittstellen zusammengeführt werden.

5 Zusammenfassung und Ausblick

In diesem Beitrag wurde ein Konzept für die effiziente Entwicklung adaptiver interaktiver Selbstbedienungssysteme in einer verteilten Multiplattformumgebung vorgestellt. Durch die Anpassungsfähigkeit der Benutzungsschnittstelle und der Systemfunktionalität sollen die jeweiligen Benutzungsschnittstellen der Selbstbedienungssysteme eine hohe Flexibilität und gute Bedienbarkeit für heterogene Benutzergruppen unterstützen. In diesem Beitrag sind wir auf ein Fallbeispiel aus der Praxis eingegangen, woraus Anforderungen für die Entwicklung adaptiver interaktiver Selbstbedienungssysteme abgeleitet wurden. Aufbauend auf den beschriebenen Anforderungen wurde ein modellbasierter Ansatz für die Umsetzung von adaptiven flexiblen Benutzungsschnittstellen für interaktive Selbstbedienungssysteme vorgestellt, der an das etablierte CAMELEON Reference Framework angelehnt ist. Für die Planung der technischen Umsetzung sind wir aktuell dabei − auf Basis der durchgeführten Analyse von relevanten Anwendungsfällen und gebräuchlichen Architekturen und Schnittstellen im Bereich der Selbstbedienungssysteme − geeignete Adaptionsmodelle zu entwickeln, die in den modellbasierten Entwicklungsprozess von Benutzungsschnittstellen integriert werden können.

Literaturverzeichnis

[AST09] Asadollahi, R.; Salehie, M.; Tahvildari, L.: StarMX: A Framework for Developing Self-managing Java-based Systems. In: Proceedings of the 2009 Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS ‘09); S. 58–67.

[BBF09] Blair, G. S.; Bencomo, N.; France, R.: Models@run.time. IEEE Computer 42 (2009), Nr. 10, S. 22–27.

[BP10] Baresi, L.; Pasquale, L.: Live Goals for Adaptive Service Compositions. In: Proceedings of the 2010 Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS’10). ACM, New York, NY, USA, 2010; S. 114–123.

[Bo06] Botterweck, G.: A Model-driven Approach to the Engineering of Multiple User Interfaces. In (Kühne, T. Hrsg.): Proceedings of the 2006 International Conference on Models in Software Engineering (MoDELS'06). Springer-Verlag, Berlin/Heidelberg, 2006; S. 106–115.

[Ca03] Calvary, G.; Coutaz, J.; Thevenin, D.; Limbourg, Q.; Bouillon, L.; Vanderdonckt, J.: A Unifying Reference Framework for Multi-target User Interfaces. Interacting with Computers (2003), S. 289–308.

[FR07] France, R.; Rumpe, B.: Model-driven Development of Complex Software: A Research Roadmap. In: Proceedings of 2007 Future of Software Engineering (FOSE '07). IEEE Computer Society, Washington, DC, USA, 2007; S. 37–54.

[GCS03] Garlan, D.; Cheng, S.-W.; Schmerl, B.: Increasing System Dependability Through Architecture-based Self-repair. In: Architecting Dependable Systems. Springer-Verlag, Berlin/Heidelberg, 2003, S. 61–89.

[Ge09] Geihs, K.; Reichle, R.; Wagner, M.; Khan, M. U.: Modeling of Context-aware Self-adaptive Applications in Ubiquitous and Service-oriented Environments. In: Proceedings of the 2009 Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS '09); S. 146–163.

[HMZ11] Hussmann, H.; Meixner, G.; Zuehlke, D. (Hrsg.): Model-Driven Development of Advanced User Interfaces, Springer-Verlag, Berlin/Heidelberg, 2011.

[KC03] Kephart, J. O.; Chess, D. M.: The Vision of Autonomic Computing. Computer 36 (2003), Nr. 1, S. 41–50.

[Kl11] Kluge, V.; Honold, F.; Schüssel, F.; Weber, M.: Ein UML-basierter Ansatz für die modellgetriebene Generierung grafischer Benutzerschnittstellen. In: Informatik 2011: Informatik schafft Communities, Workshop: Modellbasierte Entwicklung von Benutzungsschnittstellen 2011, Berlin. Gesellschaft für Informatik e.V. (GI), 2011.

[KM07] Kramer, J.; Magee, J.: Self-managed Systems: An Architectural Challenge. In: Proceedings of 2007 Future of Software Engineering (FOSE'07). IEEE Computer Society, Washington, DC, USA, 2007; S. 259–268.

[Lu11] Luckey, M.; Nagel, B.; Gerth, C.; Engels, G.: Adapt Cases: Extending Use Cases for Adaptive Systems. In: Proceedings of the 6th Intl. Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS'11). ACM, New York, NY, USA, 2011; S. 30–39.

[Li08] Link, S.; Schuster, T.; Hoyer, P.; Abeck, S.: Modellgetriebene Entwicklung grafischer Benutzerschnittstellen (Model-Driven Development of Graphical User Interfaces). i-com 6 (2008), Nr. 3. Oldenbourg, München, S. 37–43.

[Me11] Meixner, G.: Modellbasierte Entwicklung von Benutzungsschnittstellen. Informatik Spektrum 34 (2011), Nr. 4, S. 400–404.

[WMA10] Weyns, D.; Malek, S.; Anderson, J.: FORMS: A FOrmal Reference Model for Self-adaptation. In: Proceedings of the 7th International Conference on Autonomic Computing (ICAC'10). ACM, New York, NY, 2010; S. 205–214.

[Wh09] Whittle, J.; Sawyer, P.; Bencomo, N.; Cheng, B. H. C.; Bruel, J.-M.: RELAX: Incorporating Uncertainty into the Specification of Self-adaptive Systems. In: Proceedings of the 17th IEEE International Requirements Engineering Conference (RE '09). IEEE Computer Society, Washington, DC, USA, 2009; S. 79–88.