Fabian Christ: Automatische Kompatibilitätsprüfung Framework-basierter Anwendungen

1. Gutachter: Prof. Dr. Gregor Engels (Universität Paderborn)

2. Gutachter: Prof. Dr. Wilhelm Schäfer (Universität Paderborn)

Datum der Prüfung: 20.12.2012

Zusammenfassung Software-Architekturen betrieblicher Informationssysteme bestehen aus Architekturbausteinen wie Frameworks, Komponenten und Bibliotheken. Die Entwicklung dieser Architekturen unter Einbeziehung wiederverwendbarer Architekturbausteine ist eine Voraussetzung einer industriell organisierten Software-Entwicklung. Für eine effiziente Trennung der Zuständigkeiten werden dabei häufig Architekturbausteine von Drittanbietern wiederverwendet. Die Gewährleistung der Kompatibilität der eingesetzten Architekturbausteine untereinander stellt in einem komplexen Geflecht von Abhängigkeiten eine besondere Herausforderung dar.

Frameworks als erweiterbare Architekturbausteine bieten besondere Vorteile. Sie erlauben sowohl die Wiederverwendung der Funktionalität als auch der durch das Framework vorgegebenen Software-Architektur. Beispiele sind Frameworks für Benutzungsoberflächen oder für die Anbindung von Datenbanken. Durch Implementierung anwendungsspezifischer Erweiterungen wird ein Framework für den konkreten Anwendungsfall angepasst. Eine Anwendung, deren Software-Architektur ein Framework einsetzt, benutzt das Framework über dessen Erweiterungspunkte.

Im Laufe der Evolution einer solchen Anwendung entsteht häufig die Situation, dass das Framework durch eine neuere Version aktualisiert werden soll. Die Aktualisierung enthält das Risiko, dass Inkompatibilitäten zwischen bestehender Anwendung und neuer Framework-Version auftreten, die wiederum zu aufwendigen Anpassungen führen. Daher müssen mögliche Inkompatibilitäten vor der Aktualisierung erkannt und bewertet werden. Diese Inkompatibilitäten entstehen häufig an den so genannten Erweiterungspunkten eines Frameworks, also an den Stellen, an denen eine Software die gegebene Funktionalität des Frameworks erweitert. Nach aktuellem Stand der Technik ist eine automatische Kompatiblitätsprüfung nicht möglich, so dass es in der industriellen Praxis zu unvorhergesehenen Problemen verbunden mit hohen Kosten kommt.

Wir stellen ein Verfahren zur automatischen Kompatibilitätsanalyse Framework-basierter Anwendungen vor, mit dem das beschriebene Problem gelöst wird. Durch eine Kombination aus Codeanalyse und neuartiger Framework-Beschreibung lassen sich mögliche Inkompatibilitäten vor Durchführung der Aktualisierung automatisch berechnen. Hierzu definieren wir mit dem Framework Description Metamodel (FDMM) eine ganzheitliche Beschreibungssprache für Frameworks, die es insbesondere ermöglicht die charakterisistischen Erweiterungspunkte von Frameworks formal zu beschreiben. Dies ist eine wesentliche Voraussetzung für die Durchführung einer automatischen Kompatibilitätsprüfung. Eine prototypische Implementierung des Verfahrens im Werkzeug "Companian" demonstriert dessen praktische Einsetzbarkeit.

Für die Definition der benötigten Framework-Beschreibungssprache mittels Meta-Modellierung entwickeln wir in dieser Arbeit die Modellierungstechnik der parametrisierten Meta-Modelle. Diese Technik ist ein erweiterter Ansatz zur Modularisierung von Meta-Modellen, die eine anforderungsbasierte Wiederverwendung von Sprachen unterstützt. Mit diesem Ansatz können bestehende Sprachen für Teilaspekte neu definierter Sprachen wiederverwendet werden. Die Wiederverwendung wird auf Basis von Meta-Modell-Parametern realisiert. Hierzu definieren wir in einem modularisierten Meta-Modell formale Meta-Modell-Parameter, an die Meta-Modelle wiederverwendeter Sprachen gebunden werden. Ein Meta-Modell-Parameter ist definiert durch ein Meta-Modell, das bei Bindung an das Meta-Modell der wiederverwendeten Sprache durch dieses ersetzt wird.

Durch die Einführung von Meta-Modell-Parametern führen wir eine formale Technik ein, um Sprachen auf Ebene der Meta-Modelle wiederverwenden zu können. Unser pragmatischer Ansatz stellt dabei sicher, dass hierbei nur die Sprachen wiederverwendet werden können, deren Meta-Modelle syntaktisch und semantisch dem geforderten Meta-Modell entsprechen.

Veröffentlicht als: Fabian Christ: Automatische Kompatibilitätsprüfung Framework-basierter Anwendungen. PhD thesis, Universität Paderborn (2012)
Online unter
http://nbn-resolving.de/urn:nbn:de:hbz:466:2-10527

Christian Gerth: Change Management for Business Process Models

1. Gutachter: Prof. Dr. Gregor Engels (Universität Paderborn)

2. Gutachter: Prof. Dr. Wilhelm Schäfer (Universität Paderborn)

3. Gutachterin: Prof. Dr. Gerti Kappel (Technische Universität Wien)

Datum der Prüfung: 05. Juli 2012

Zusammenfassung:
Prozessmodelle sind seit vielen Jahren ein wichtiges Mittel, um den dynamischen, flexiblen Anteil von Softwaresystemen und Unternehmensarchitekturen zu beschreiben. Dies gilt insbesondere für Softwaresysteme, die nach dem Paradigma einer serviceorientierten Architektur (SOA) strukturiert sind. Hier werden Prozessmodelle genutzt, um den häufigen Veränderungen unterworfenen Anteil eines Systems zu beschreiben.

Aufgrund der Größe heutiger Unternehmensarchitekturen bzw. heutiger Softwaresysteme ist es unumgänglich, dass bei der Entwicklung und Pflege zugehöriger Prozessmodelle mehrere Personen in verteilten Teams beteiligt sind. Dies führt damit auch unmittelbar zu dem Problem, dass zu einem Prozessmodell mehrere unterschiedliche Versionen entstehen, die im Rahmen einer Konsolidierung wieder zusammengebracht werden müssen. Hierzu müssen entstandene Abhängigkeiten und Konflikte in unterschiedlichen Versionen eines Prozessmodells erkannt und aufgelöst werden.

Dieses bekannte Problem aus dem Bereich der Versions- und Konfigurationsmanagementsysteme ist für textuelle Softwareentwicklungsartefakte weitgehend gelöst und wird im alltäglichen Betrieb durch entsprechende Werkzeuge unterstützt. Im Falle von (graphischen) Modellen und insbesondere für Prozessmodelle steht eine derartige Werkzeugunterstützung bisher noch nicht in ausreichendem Maße zur Verfügung.

Diesem Problem des Versionsmanagements für Prozessmodelle widmet sich die vorliegende Dissertation. Um das Zusammenführen von Prozessmodellen, die in verschiedenen Sprachen modelliert sind, zu unterstützen, wird dazu zunächst eine sprachunabhängige Zwischendarstellung für Prozessmodelle entwickelt. Diese *Intermediate Representation* (IR) für Prozessmodelle beinhaltet gemeinsame syntaktische und semantische Kernkonzepte von verschiedenen Prozessmodellierungssprachen, wie z.B. *Business Process Model and Notation* (BPMN), *Business Process Execution Language* (BPEL) oder *Aktivitätendiagramme der Unified Modeling Language* (UML-AD).

Aufbauend auf dieser einheitlichen Zwischendarstellung wird ein Verfahren zur Erkennung von Unterschieden zwischen Prozessmodellversionen in Form von Änderungsoperationen vorgestellt, das insbesondere zur Erkennung von zusammengesetzten Änderungen führt. Die erkannten Änderungsoperationen repräsentieren Unterschiede zwischen zwei Prozessmodellversionen und können zur Konsolidierung der unterschiedlichen Versionen angewendet werden, um Unterschiede zu beseitigen. Im nächsten Schritt wird eine Methode zur Feststellung von Reihenfolgeabhängigkeiten zwischen einzelnen Änderungsoperationen beschrieben, die auf Technikenn aus der Theorie der parallelen Graphtransformationen basiert. Durch die Berücksichtigung der Positionen im Prozessmodell, an denen Änderungsoperationen auszuführen sind, und die *dynamische* Festlegung von Positionsparametern nach Ausführung einer jeweiligen Änderungsoperation, ermöglicht die Methode einen komfortablen Umgang mit sequentiellen Abhängigkeiten.

Anschließend werden verschiedene Typen von Konflikten klassifiziert, die bei einer parallelen Weiterentwicklung von Prozessmodellen entstehen können. Zusätzlich wird ein Verfahren vorgestellt, wie die verschiedenen Konflikttypen beim Zusammenführen von Prozessmodellversionen entdeckt und aufgelöst werden können. In dem Verfahren werden Prozessmodelle semantisch verglichen, um syntaktisch unterschiedliche Bereiche in Prozessmodellen zu identifizieren, die dieselbe Bedeutung haben.

Als initiale Validierung wurden große Teile des Framework für das Versionsmanagement für Prozessmodelle prototypisch implementiert. Die Identifikation von Unterschieden zwischen Prozessmodellen und deren Abhängigkeiten wurden auch in dem von IBM kommerziell vertriebenen Werkzeug *IBM WebSphere Business Modeler V 7.0*[1] (WBM) übernommen.

Veröffentlicht als:
Christian Gerth. Business Process Models - Change Management, volume 7849 of Lecture Notes in Computer Science. Springer-Verlag, Berlin/Heidelberg, 2013.
Online unter: http://www.springeronline.com/978-3-642-38603-9

[1] `http://www.ibm.com/software/integration/wbimodeler/entry/`