Kooperative Methoden- und Werkzeugentwicklung zur Cloudmigration von proprietären Anwendungskomponenten

Benjamin Nagel[1], Klaus Schröder[2], Steffen Becker[3], Stefan Sauer[1], and Gregor Engels[1]

[1]s-lab - Software Quality Lab, Warburger Straße 100, 33098 Paderborn , {bnagel,sauer,engels}@s-lab.upb.de
[2]S&N AG, Klingender Straße 5, 33100 Paderborn , kschroeder@s-und-n.de
[3]Software Engineering Chair, Chemnitz University of Technology , Straße der Nationen 62, 09111 Chemnitz , steffen.becker@informatik.tu-chemnitz.de

Abstract: Die cloudbasierte Bereitstellung von Funktionen als Software-as-a-Service (SaaS) ermöglicht die Erschließung neuer Kundenpotenziale bei kleinen und mittelgroßen Unternehmen. Um bestehende Anwendungskomponenten als SaaS anzubieten, muss eine entsprechende Migration durchgeführt werden. Das Projekt AACC beschäftigt sich mit der modellgetriebenen Überführung von Anwendungskomponenten in Cloud-Computing-Umgebungen. Der hierbei durchgeführte Wissenstransfer beinhaltet softwaretechnische Kompetenzen zur Methodendefinition, Konzepte der modellgetriebenen Softwareentwicklung und technische Kenntnisse zur Implementierung einer integrierten Werkzeugunterstützung. In diesem Beitrag beschreiben wir die Inhalte des Wissenstransfers, die erarbeiteten Lösungen und unsere Erfahrungen aus dem Projekt.

1 Einleitung

Betriebliche Informationssysteme (IS) werden branchenübergreifend in Unternehmen verschiedener Größe eingesetzt, um durch die effektive Steuerung der Geschäftsprozesse dem steigenden Wettbewerbsdruck zu begegnen. Die aufwändige Einführung und Wartung von betrieblichen IS stellt insbesondere für kleinere und mittlere Unternehmen (KMU) ein hohes wirtschaftliches Risiko dar. Bei der Bereitstellung dieser Funktionen durch eine Cloud-Lösung, als Software-as-a-Service (SaaS), werden Wartung und Betrieb von einem Provider übernommen. Unternehmen können so die bereitgestellten Funktionen nutzen, ohne selbst eine entsprechende Infrastruktur pflegen zu müssen. Für die Anbieter von Komponenten betrieblicher IS erschließen sich durch das Angebot ihrer Software als SaaS neue Kundenfelder insbesondere im Bereich von KMU.

Die Überführung von Komponenten eines betrieblichen IS zu cloud-fähigen Services gestaltet sich aufgrund der hohen Komplexität und der zu berücksichtigen Abhängigkeiten äußerst anspruchsvoll. Die manuelle Migration bedeutet einen erheblichen Aufwand, der gerade für kleine und mittlere Softwarehäuser ein erhebliches technisches und wirtschaftliches Entwicklungsrisiko darstellt.

Im Rahmen des Förderprojektes AACC entwickelt das s-lab - Software Quality Lab der Universität Paderborn in Kooperation mit der S&N AG ein methodenbasiertes Werkzeug für die automatisierte Überführung von Anwendungskomponenten in Cloud-Computing-Umgebungen. Um einen hohen Grad an Automatisierung und Anpassbarkeit zu erreichen, verfolgt AACC einen modellgetriebenen Migrationsansatz. Wie in Abbildung 1 dargestellt, werden hierzu ein methodisches Vorgehen definiert und eine Werkzeugunterstützung implementiert.

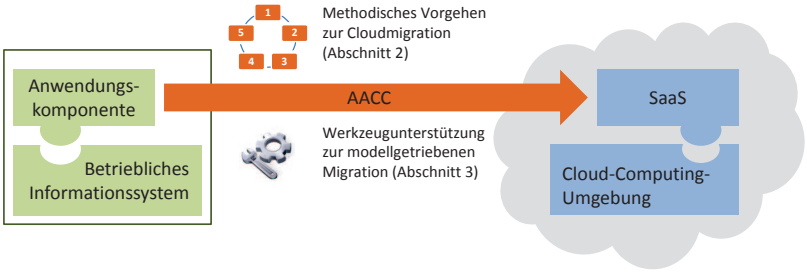

Abbildung 1: Überblick und Arbeitspakete des AACC Projektes

Das Projekt AACC wird vom Bundesministerium für Wirtschaft und Technologie im Rahmen des Programms ZIM - Zentrales Innovationsprogramm Mittelstand gefördert. Ein wesentliches Ziel dieses Programms ist der Wissenstransfer von Forschungseinrichtungen und KMU. In diesem Projekt fokussiert sich der bilaterale Wissenstransfer auf die folgenden drei Kompetenzen:

- Softwaretechnisches Know-How zur Methodendefinition (1)
- Konzeptionelles Know-How zur modellbasierten Softwareentwicklung (2)
- Technisches Know-How zur Implementierung der Werkzeugunterstützung (3)

Die Kompetenzen (1) & (2) dienen zur Definition der Methode. Für die Implementierung der Werkzeugunterstützung werden die Kompetenzen (2) & (3) benötigt. Unsere Erfahrungen aus der Methodenentwicklung werden in Abschnitt 2 beschrieben. In Abschnitt 3 erläutern wir die Realisierung des Werkzeugs.

2 Methodisches Vorgehen zur Cloudmigration

Eine modellgetriebene Migration bietet den Vorteil eines hohen Grads an Automatisierung und Wiederverwendbarkeit. Die methodische Herausforderung besteht in einer systematischen Ableitung der benötigten Modelle und Transformationsregeln, um sicherzustellen, dass diese konsistent zu den Zielen und Anforderungen der involvierten Stakeholder sind. Hierfür haben wir in dem AACC Projekt ein methodisches Vorgehen definiert, in

dem ausführbare Transformationsregeln aus abstrakteren Anforderungen abgeleitet werden. Um zwei wesentliche Sichtweisen auf Anforderungen zu berücksichtigen, unterscheiden wir zwischen Geschäftsanforderungen und technischen Anforderungen.

In den Geschäftsanforderungen werden Aspekte erfasst, die sich aus dem Geschäftsmodell ergeben. Technische Anforderungen werden als plattformunabhängig oder plattformspezifisch klassifiziert. Die plattformunabhängigen Anforderungen ergeben sich aus der Migration zu einem SaaS (z.B. Skalierbarkeit, Sicherheit, Pay-per-Use Funktionalität). Sie sind unabhängig von der konkreten Zielplattform und adressieren Aspekte, die sich aus dem allgemeinen Konzept der Cloud ergeben. Im Gegensatz dazu ergeben sich plattformspezifische Anforderungen aus der Entscheidung für eine bestimmte Cloud-Computing-Umgebung.

Um Nachvollziehbarkeit und Verständlichkeit für die verschiedenen Stakeholder zu gewährleisten, werden verschiedene Techniken zur Anforderungserhebung und -spezifikation kombiniert. Als Ausgangspunkt dient die Definition von Migrationszielen, die erreicht werden sollen [DvLF93]. Hierbei verwendeten wir eigene Forschungsarbeiten im Bereich zielorientierter Anforderungserhebung zur Definition von zeitlichen Abhängigkeiten zwischen Zielen [NGPE13]. Diese haben wir mit bestehenden Techniken des Projektmanagements kombiniert, um Zeitpläne für die Durchführung der Migration zu erstellen.

Für die Identifikation der Migrationsziele haben wir verschiedene Erhebungstechniken verwendet. Zur Erhebung und Spezifikation der Geschäftsanforderungen, haben sich Szenarien [MKK05, Poh10] und User-Stories als sehr hilfreich und effizient erwiesen, da sie gut von der konkreten technischen Zielsetzung abstrahieren und sich zur Erfassung der benötigten Abläufe eignen. Die Beschreibung dieser Szenarien und User-Stories wurde durch Interviews und Workshops mit den Fachexperten durchgeführt. Hierdurch wurde auch die Einbeziehung der Stakeholder in das innovative Forschungsprojekt und die damit verbundenen Ziele erleichtert.

Migrationsziele, die sich aus plattformspezifischen und -unabhängigen Rahmenbedingungen ergeben, konnten sehr gut mit einem einfachen Top-Down-Ansatz definiert werden. Hier war die intensive Analyse sowie das Sichten relevanter Informationen ein wesentlicher Teil des Aufwands. Da die Plattform zu Beginn des Projektes noch nicht verfügbar war, mussten die plattformabhängigen Anforderungen über Korrespondenzen mit dem Anbieter der proprietären Software und den zur Verfügung gestellten Previews abgeleitet werden.

Ausgehend von den Zielen haben wir einen Use-Case-orientierten Ansatz auf Basis von UML Use-Case-Diagrammen [OMG11] gewählt. Hierbei definieren wir die einzelnen Schritte der Migration als Migrations-Use-Cases (MUC). Ein MUC kann als eine abstrakte und leicht verständliche Sicht auf eine Modelltransformation verstanden werden. In einem letzten Schritt wurden aus den MUCs formalisierte und ausführbare Transformationsregeln abgeleitet. Die technische Ausführung dieser Regeln wird in Abschnitt 3 erläutert.

Die systematische, zielorientierte Verfeinerung bis zu ausführbaren Transformationsregeln bietet eine praktikable Systematik. Des weiteren kann durch einfache Techniken, wie das Vergeben von eindeutigen Bezeichnern und der Definition und Pflege von Anforderungsbeziehungen, die Rückverfolgbarkeit von Transformationsregeln zu Migrationszielen er-

reicht werden.

Einen wesentlichen Beitrag zur Identifikation des tatsächlichen *Migrationsbedarfs* im Sinne von notwendigen Änderungen an der Ausgangssoftware, hat die intensive Einbindung der Entwickler geleistet. Wie bereits in anderen Migrationsprojekten beobachtet werden konnte [GGS12], ist die Verfügbarkeit von Domänenwissen ein kritischer Faktor für den Erfolg eines solchen Vorhabens.

Eine wichtige Erfahrung aus der Methodenentwicklung war die Wichtigkeit der richtigen Detail- und Formalisierungsebene in der Modellierung. Während die automatisierte Transformation eine formalisierte Beschreibung der Regeln erfordert, müssen die einzelnen Schritte der Migration auch verständlich und nachvollziehbar sein. Hierfür hat sich die Kombination aus Zielen, Szenarios und Use-Cases als sehr hilfreich erwiesen.

3 Werkzeugunterstützung zur modellgetriebenen Migration

Das zweite Ziel des AACC Projektes ist die Entwicklung eines Werkzeugs zur modellgetriebenen Migration. Des weiteren soll die Migration für einen konkreten Anwendungsfall realisiert und durchgeführt werden. Hierbei soll eine bestehende SAP R/3 Komponente zu einem Service migriert werden, der in der SAP HANA Cloud ausgeführt werden kann.

Wie in Abbildung 2 dargestellt, ist der implementierte Migrationsprozess an das bekannte Migrationshufeisen angelehnt [KWJC98]. Hierbei wird zunächst der Programmcode aus dem Quellsystem exportiert und anschließend in ein Modell geparst. Dieses kann durch einen abstrakten Syntaxbaum (AST) abgebildet werden. Mithilfe der definierten Transformationsregeln wird das Modell des Quellsystems in ein Modell des Zielsystems transformiert. Aus dem transformierten Modell wird Programmcode generiert, welcher in das Zielsystem importiert werden kann.

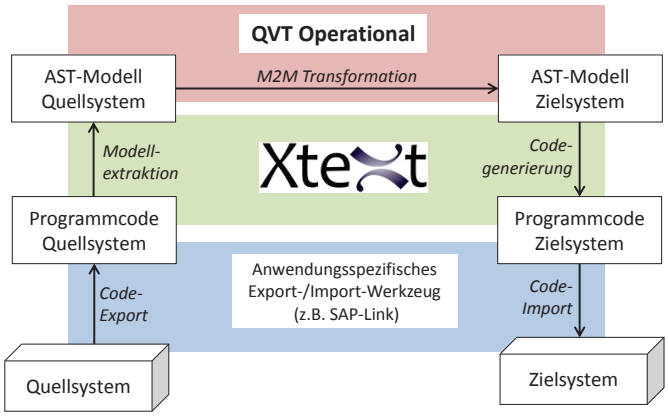

Abbildung 2: Übersicht des implementierten Migrationsansatzes

Für dieses Vorgehen konnten bestehende Verfahren und Sprachen verwendet werden. Als

Grundlage für die Entwicklung diente das Eclipse Framework. Zur Realisierung der Transformationsregeln haben wir QVT Operational [1] verwendet. Des weiteren nutzen wir XText [2] und die unterliegenden Parsing-Funktionalitäten zur Modellerstellung und Codegenerierung. Für den Export bzw. Import des Programmcodes müssen anwendungsspezifische Werkzeuge verwendet werden. Die Einordnung der verwendeten Technologien in die verschiedenen Phasen der modellgetriebenen Migration ist in Abbildung 2 dargestellt.

Im Rahmen des Projektes wurden die einzelnen Schritte beispielhaft für die Überführung einer Anwendungskomponente erprobt. Um den relevanten Programmcode zu extrahieren, haben wir in unserem Anwendungsszenario das Werkzeug SAPLink [3] verwendet. Die zu migrierende Komponente ist in der proprietären Programmiersprache ABAP implementiert. Für diese Sprache konnte kein frei verfügbarer Parser gefunden werden. Eine entsprechend formalisierte Grammatik der Sprache ABAP lag ebenfalls nicht vor.

Die Aufwände für die manuelle Entwicklung einer entsprechenden Grammatik hätten den geplanten Zeitraum und die verfügbaren Ressourcen für dieses Projekt deutlich überschritten. Aus diesem Grund haben wir einen innovativen Ansatz gewählt, der die Entwicklungskompetenzen des Industriepartners S&N und das konzeptionelle Know-How des s-labs vereint. Auf Basis der online verfügbaren ABAP Sprachreferenz [4] haben wir einen Generator mit verschiedenen Übersetzungsregeln entwickelt, der die Erzeugung einer verwendbaren Grammatik ermöglicht. Unter Nutzung der erstellten Grammatik, konnte eine entsprechende Werkzeugunterstützung realisiert werden.

Bei der Konzeption und Entwicklung haben sich die Kompetenzen beider Partner gut ergänzt. Die Entwicklungskompetenzen des Industriepartners waren v.a. in der Integration zu einer durchgängigen Werkzeugkette hilfreich. Bei der Auswahl der verwendeten Technologien konnten Erfahrungen aus Forschungs- und Entwicklungsprojekten genutzt werden.

4 Zusammenfassung und Ausblick

Die enge Kooperation und Abstimmung im Rahmen eines agilen Vorgehens haben wesentlich zu einem effizienten Wissenstransfer und damit auch zum Erfolg des Projektes beigetragen. Die explizite Definition von Migrationszielen und ihre schrittweise Verfeinerung haben zu einer guten Akzeptanz durch die verschiedenen Stakeholder geführt und auch eine solide Diskussionsgrundlage geboten.

Durch die Verwendung etablierter Technologien und frühzeitiger technischer Durchstiche konnten regelmäßige Projektfortschritte erzielt werden, die ebenfalls die Einbindung der verschiedenen Stakeholder erleichterten und dabei halfen die Potenziale eines modellgetriebenen Ansatzes aufzuzeigen.

Der in Abbildung 3 dargestellte Wissenstransfer war hierbei ein kritischer Erfolgsfaktor.

[1] http://wiki.eclipse.org/QVTo
[2] http://www.eclipse.org/Xtext/
[3] http://wiki.scn.sap.com/wiki/display/ABAP/SAPlink
[4] http://help.sap.com/abapdocu_731/en/abenabap.htm

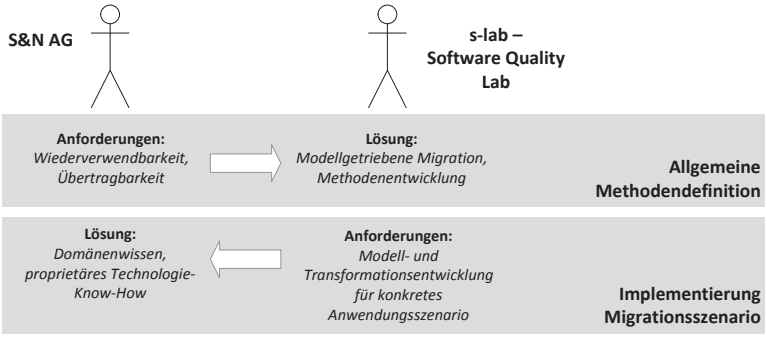

Abbildung 3: Wissenstransfer für die modellgetriebene Migration

Zur Definition einer allgemeinen Methode hat der Forschungspartner Konzepte und Techniken der modellgetriebenen Migration beigetragen. Bei der Anwendung auf das konkrete Anwendungsszenario war das Domänenwissen und die Kenntnisse proprietärer Technologien des Industriepartners notwendig um entsprechende Modelle und Transformationen zu entwickeln. Neben dem Transfer des Fachwissens, war auch ein grundlegendes Verständnis für die verschiedenen Aspekte auf beiden Seiten (Transfernehmer und -geber) erforderlich.

Literatur

[DvLF93] Anne Dardenne, Axel van Lamsweerde und Stephen Fickas. Goal-directed requirements acquisition. *Science of Computer Programming*, 20(1-2):3–50, 1993.

[GGS12] Marvin Grieger, Baris Güldali und Stefan Sauer. Sichern der Zukunftsfähigkeit bei der Migration von Legacy-Systemen durch modellgetriebene Softwareentwicklung. In *Proc. of the 14th Workshop Software-Reengineering (WSR)*, Jgg. 32, Seiten 37–38. Softwaretechnik-Trends, 2012.

[KWJC98] Rick Kazman, Steven G Woods und S Jeromy Carriere. Requirements for integrating software architecture and reengineering models: CORUM II. In *Reverse Engineering, 1998. Proceedings. Fifth Working Conference on*, Seiten 154–163. IEEE, 1998.

[MKK05] Subhas Misra, Vinod Kumar und Uma Kumar. Goal-oriented or scenario-based requirements engineering technique - What should a practitioner select? In *Canadian Conference on Electrical and Computer Engineering*, Seiten 2288–2292. IEEE, 2005.

[NGPE13] Benjamin Nagel, Christian Gerth, Jennifer Post und Gregor Engels. Kaos4SOA - Extending KAOS Models with Temporal and Logical Dependencies. In *Proc. of the Forum at the 25th Int. Conf. on Advanced Information Systems Engineering*, Seiten 9–16, 2013.

[OMG11] OMG. OMG Unified Modeling Language (OMG UML), Superstructure, Version 2.4.1, 2011.

[Poh10] Klaus Pohl. *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer, 2010.