MMSM 2018 – Ergebnisse des 4. Workshops „Modellgetriebene und modellbasierte Softwaremodernisierung“

Stefan Sauer[1], Matthias Riebisch[2], Marco Konersmann[3], Robert Heinrich[4], Steffen Becker[5]

[1]Universität Paderborn, SICP | Software Innovation Lab, 33102 Paderborn | sauer@sicp.upb.de
[2]Universität Hamburg, Fachbereich Informatik, 22527 Hamburg | riebisch@informatik.uni-hamburg.de
[3]Universität Duisburg-Essen, PALUNO, 45127 Essen | marco.konersmann@paluno.uni-due.de
[4]KIT – Karlsruher Institut für Technologie, IPD, 76131 Karlsruhe | robert.heinrich@kit.edu
[5]Universität Stuttgart, Institut für Softwaretechnologie, 70569 Stuttgart | steffen.becker@iste.uni-stuttgart.de

1 Einleitung

Dieser Beitrag liefert eine Retrospektive auf den 4. Workshop „Modellgetriebene und modellbasierte Softwaremodernisierung“ (MMSM 2018)[1], der im Rahmen der Konferenz „Modellierung“ am 21. Februar 2018 in Braunschweig stattfand. Er fasst die Ergebnisse der Vorträge, Diskussionen und interaktiven Arbeitsformate zusammen. Die Proceedings sind bei CEUR[2] erschienen.

2 Workshopreihe MMSM

Bei MMSM 2018 handelt es sich um einen gemeinsamen Workshop der Arbeitskreise „Langlebige Software-Systeme“ (L2S2), „Modellgetriebene Software-Entwicklung“ (MDA) und „Traceability/Evolution“ der Gesellschaft für Informatik (GI). Der Workshop findet traditionell alle zwei Jahre im Rahmen der Konferenz „Modellierung“ statt. Am 21. Februar 2018 trafen sich circa 20 Teilnehmer(innen) aus Wissenschaft und Praxis, um neue Forschungstrends auf Basis des aktuellen industriellen Bedarfs und geeignete Lösungsansätze zu identifizieren sowie Erfahrungen mit modellbasierten und modellgetriebenen Techniken und Methoden für die Softwaremodernisierung auszutauschen.

3 Vorträge und Diskussionen

Der Workshop begann mit einer **Keynote** von Markus Völter von der itemis AG zum Thema „Mit DSLs zu nachhaltigen Systemen“. Er präsentierte, wie mit domänenspezifischen Sprachen (DSLs) die Fachlichkeit getrennt von der Implementierungstechnologie entworfen und gewartet werden kann. Dazu gab er Einsichten in die Vorteile und Fallstricke der Entwicklung von DSLs. Anschaulich illustrierte er dies an Praxisprojekten aus der Versicherungsbranche, der Medizin und dem Güterkraftverkehr. Wichtiges Werkzeug bei Erstellung, Kombination und Wartung von DSLs sind „Language Workbenches“, die den Aufwand für diese Tätigkeiten reduzieren. Als Beispiel nannte er u.a. das Open-Source-Tool JetBrains MPS und gab Unternehmen die Empfehlung, in den Aufbau von DSL-Know-how zu investieren, um zukünftige Legacy-Probleme zu vermeiden.

Zwei weitere eingeladene Vorträge berichteten von **Erfahrungen aus der Praxis**:

Dr. Jens Zimmermann von der Altran Deutschland S.A.S. & Co. KG präsentierte eine Fallstudie für die Modernisierung von Automotive-Software. Mithilfe einer domänenspezifischen Sprache für Softwarearchitektur wurde hier die Softwarequalität hinsichtlich Wartbarkeit und Evolvierbarkeit erhöht und es wurden wiederverwendbare Methoden und Werkzeuge entwickelt.

Kenny Wehling von der Volkswagen AG präsentierte ein Verfahren zur „automatisierten Identifizierung von Restrukturierungspotentialen in Systemvarianten zur Reduzierung unnötiger IT-Komplexität“. Auf Basis einer Variabilitätsanalyse werden Architekturempfehlungen gegeben. Hiermit wird die Konsolidierung und Modernisierung einer IT-Landschaft vorbereitet und geplant.

Darüber hinaus wurden noch zwei **begutachtete Beiträge** im Rahmen des Workshops präsentiert:

Harry Sneed von der SoRing Kft präsentierte Erfahrungen aus der „Rückwärtsmodellierung bestehender Finanzdienstleistungssysteme“, wo komplexe COBOL-basierte Programme auf Basis modellbasierter Entwicklung modernisiert werden.

Ivan Jovanovikj von der Universität Paderborn beschrieb einen Ansatz zur modellbasierten Testfall-Migration. Der Ansatz erweitert das „Hufeisenmodell“ der modellbasierten Softwaremigration um eine Schicht zur Testfall-Co-Migration.

[1] http://akl2s2.ipd.kit.edu/veranstaltungen/mmsm2018/
[2] http://ceur-ws.org/Vol-2060/

4 Working Session

Ziel der interaktiven „Working Session“ war die Diskussion und Bewertung der Methoden, Ansätze, Technologien und Vorgehensweisen, die von gemeinsamem Interesse sind. Dazu wurden zunächst die Interessen der Teilnehmer(innen) zusammengetragen und danach priorisiert, um drei ausgewählte Fragestellungen zu erörtern.

Als das am höchsten priorisierte Thema wurde zuerst eine Bewertung des Standes der Technik zur **Modellierung von Fachlichkeit** vorgenommen. Ziel war es, Defizite der existierenden und Potenzial neuer Lösungsansätze zu ermitteln. Ergebnis: Die etablierten Modellierungssprachen der verschiedenen Disziplinen wie BPMN, UML, ER-Diagramme und Entscheidungstabellen haben den Vorteil, durch Fachexperten der Domäne (meist sind dies auch die Kunden der Softwareprojekte) akzeptiert zu werden, so dass Modelle der Fachlichkeit gemeinsam von Entwicklern und Fachexperten erstellt werden können. Weniger formale Modellierungsansätze und Kreativmethoden wie Design Thinking unterstützen, Kunden und Benutzer zu beteiligen und Ideen zu generieren. Prototypen, Mock-ups und Simulationsmodelle stellen eine weitere Möglichkeit dar, Fachlichkeit zu modellieren. Damit ist sowohl das Ziel der Kommunikation zwischen Fachexperten und Entwicklern als auch das der Verifikation und Generierung erreichbar. Als dritte Option wurde die Entwicklung einer kontext- oder projektspezifisch definierten Modellierungssprache (DSL) erörtert. Der klare Vorteil besteht in der Anpassbarkeit der Syntax an die Bedürfnisse der fachlichen Fragestellung und das Verständnis der Fachexperten. Als offenes Problem wurde die Beschreibung der Semantik festgestellt, was beispielsweise durch Bezug auf die Semantik einer etablierten Modellierungssprache gelöst werden kann.

Das nächste Thema gemäß der Priorisierung war die Frage der **Übertragung von Transformationswissen** bei der Weiterentwicklung der (Transformator-)Technologie, verbunden mit der Frage, wie Transformationen effizient gestaltet werden können. Betrachtet wurden dabei drei Arten von Transformationen aus allen Bereichen des Hufeisenmodells der Softwaremigration:

1. Realisierung (Code) → Modell (Abstraktion),
2. Modell (Abstraktion der Realisierung in Technologie A) → Modell (Abstraktion der Realisierung in Technologie B),
3. Modell (Abstraktion) → Realisierung (Code, Konfigurationsdateien, Testfälle oder Dokumente).

Dafür lagen Erfahrungen mit verschiedenen Werkzeugen vor: Für (2) wird beispielsweise Quellcode in einen Aufrufgraph überführt und dann mit dem Zeichenwerkzeug NEATO visualisiert, um Cluster zu identifizieren. Für (1) unterstützt z.B. das Tool SoMoX vom KIT bei einer Überführung von Quellcode in Komponenten; Codeling (Universität Duisburg-Essen) sowie Beagle wurden hier ebenfalls genannt.

Impressionen aus der MMSM 2018 Working Session

Als letztes Thema wurde die Frage nach einer **allgemeinen Methode oder Vorgehensweise** bei der Analyse und Sanierung von Legacy-Systemen behandelt, die zwei Teilnehmer in industriellen Projekten zu lösen haben. Erfahrene Teilnehmer verwiesen auf einführende Literatur zu Reengineering und Migration sowie auf gut anwendbare Zusammenstellungen von Vorgehensweisen, aus denen man einen für das eigene Projekt und dessen spezifische Ziele, Bedingungen und Risiken passenden Methoden-Mix zusammenstellen kann. Einen weiteren Lösungsansatz bieten Architektur-Refactorings und Reengineering-Patterns.

5 Fazit

Wir danken allen Workshop-Teilnehmern für ihre wertvollen Beiträge, die Ausgangspunkt für weitere Forschung, praktische Anwendung, Diskussionen und zukünftige Workshops sein werden.