Toolgestützte Software-Migration im Wandel der Zeit

Uwe Erdmenger
pro et con Innovative Informatikanwendungen GmbH, Dittesstraße 15, 09126 Chemnitz
uwe.erdmenger@proetcon.de

Abstract

Die Nachfrage nach Software-Migrationen von Legacy-Systemen in moderne Architekturen ist ungebrochen. Ursache dafür sind Kostenreduktion, höhere Anforderungen an die Funktionalität, der Einsatz moderner Entwicklungstechnologien sowie der zunehmende Mangel an Host-Spezialisten. In den letzten Jahren wird auch verstärkt die Forderung nach einer automatischen Konvertierung von antiquierten Programmiersprachen (z.B. das auf Mainframe immer noch weit verbreitete COBOL) in moderne Sprachen laut. Die Firma pro et con entwickelt Technologien und Werkzeuge, welche Software-Migrationen automatisieren. Diese werden in Migrationsprojekten eingesetzt. Dieser Beitrag gibt einen Überblick über die Entwicklung von Technologien und Werkzeugen der Software-Migration bei pro et con im Zeitraum zwischen dem 10. WSR 2008 und dem 20. WSRE 2018.

1 Es war einmal ...

Bis zum Jahr 2008 wurden bei pro et con Stand-alone-Werkzeuge für die Software-Migration auf Kundenanforderung und projektbezogen entwickelt. Auf dem 10. WSR wurde dazu in einem Beitrag berichtet [1]. Ab dem Jahr 2009 wurde der Notwendigkeit Rechnung getragen, die vorhandenen Einzellösungen zu systematisieren, für weitere Einsatzgebiete zu öffnen und in einer sogenannten pecBOX (pro et con Toolbox für die Software-Migration) zusammenzufassen. Basis war ein vom BMBF gefördertes Verbundprojekt *SOAMIG* mit Partnern aus dem universitären Umfeld und aus der Praxis. Projektziel war die Erarbeitung eines Prozessmodells für eine SOA-Migration und prototypische Tools für deren Automatisierung. Dies beinhaltete auch die prototypische Entwicklung eines Translators *CoJaC* (COBOL to Java Converter). In diese Entwicklung flossen die Erfahrungen früherer Migrationsprojekte ein (z.B. ARNO – eine Migration von BS2000 nach UNIX bei Amadeus Germany). Es konnte auf ein „pro et con"-eigenes COBOL-Front-End aufgebaut werden. Zusätzlich kamen auch die eigenentwickelten Meta-Tools *BTRACC* (ein Parsergenerator auf Basis des Backtracking-Verfahrens) und *ReTrans* (ein Werkzeug für die Model-to-Model-Transformation) zum Einsatz. Diese Metawerkzeuge wurden im Rahmen von SOAMIG professionalisiert und um weitere Metawerkzeuge ergänzt. Bei neuen Migrationsprojekten sind diese Meta-Tools die Grundlage für schnelle und kostengünstige Anpassungen der Migrationswerkzeuge an neue Kundenanforderungen (z.B. Sprachdialekte, eingebettete Systeme, ...). Mit CoJaC als Prototyp konnte bereits ein SOAMIG-Beispielprojekt von ca. 80.000 Zeilen COBOL-Quelltext nach Java konvertiert und ablauffähig gestaltet werden.

2 Erste Konfrontation mit Kundensourcen

Im Jahr 2012 hatte CoJaC eine erste, große „Bewährungsprobe" zu bestehen. Eine Legacy-Anwendung von ca. 120.000 Zeilen COBOL-Code sollte auf ein Windows-Server-System migriert werden. Eingeschlossen war eine Konvertierung von COBOL nach Java mit CoJaC. Ein Pilot von ca. 40.300 LOC konnte dabei in nur 12 Tagen lauffähig bereitgestellt werden, inklusive der notwendigen, manuellen Nachbearbeitungen. Allerdings zeigte das Projekt auch Potenzial für die Weiterentwicklung des Werkzeugs auf. Die Bearbeitung von eingebettetem SQL musste um eine Dialektmigration von PostgreSQL nach MS SQL Server erweitert werden. Außerdem zeigte sich, dass die automatische Aufteilung des generierten Codes auf verschiedene Java-Klassen, die sich an den Copybooks des Originals orientiert, noch nicht den Kundenanforderungen genügte. Das war der Anlass für die Entwicklung eines weiteren Werkzeugs *JPackage*. JPackage stellt eine grafische Oberfläche zur Verfügung, mit der die Strukturierung des generierten Codes feingranular mit Drag&Drop realisiert werden kann. Mit JPackage wird die Zielarchitektur des migrierten Systems in Zusammenarbeit mit dem Kunden und seinen Anforderungen gestaltet. Die Menge der notwendigen, manuellen Nachbearbeitungen konnte mit JPackage nochmals deutlich reduziert werden.

3 Willkommen im wirklichen Leben

Einen folgenden, „großen Auftritt" hatten CoJaC und JPackage im Jahr 2014. Es war im Rahmen eines Kundenauftrages ein Legacy-System von BS2000 nach Linux zu migrieren. Bestandteil dieses Systems waren ca. 830.000 Zeilen COBOL-Quellcode, der nach Java konvertiert werden sollte. Wie bei jedem größeren Kundenprojekt waren dazu die Werkzeuge wiederum zu erweitern. Das betraf diesmal die Speicherung von binären COBOL-Zahlenfeldern (`USAGE COMP` oder `COMP-3`) im Java-Zielcode. Diese wurden bis dahin, wie andere Zahlen auch, in einem lesbaren Format (`DISPLAY`) gespeichert. Im Legacy-System wurden die COBOL-Felder jedoch vielfältig überlagert, so dass im konvertierten Java-Code dadurch Verschiebungen auftraten. Eine manuelle Anpassung dieser Stellen wurde aufgrund der hohen Anzahl verworfen. Daraufhin wurde CoJaC so erweitert, dass alle Datenfelder in Java intern genauso gespeichert werden, wie dies auch ursprünglich in COBOL der Fall war.
In so einem komplexen Migrationsprojekt ist die Konvertierung von COBOL nach Java ein wesentlicher Bestandteil, aber natürlich nicht die einzige Migration, die zu realisieren ist. Es sind auch Jobs, Dateien und Datenbanken, die Middleware und die Benutzeroberflächen (Masken) zu migrieren. Für alle diese Migrationspfade existierten bereits zu Projektbeginn Lösungen:

• *S2P* migriert SDF (BS2000-Jobsteuersprache) nach Perl,
• *FiRe* ist ein Werkzeugkasten für die Migration von Datenfiles/Datenbanken,
• *MidaS* konvertiert Online-Programme in Webservices,
• *MaTriX* migriert proprietäre Eingabemasken in Web-Anwendungen.

Diese Werkzeuge kamen bereits in anderen Migrationsprojekten zum Einsatz und wurden an die Anforderungen des Projektes angepasst. Beispielsweise wurde MaTriX um das Maskenbeschreibungsformat IFG erweitert [3]. Für FiRe war eine Erweiterung dahingehend erforderlich, dass mehrere, unterschiedlich strukturierte Satzarten in einzelnen COBOL-Files auftraten. Das Projekt stellte also auch eine „Bewährungsprobe“ für das Zusammenspiel der Werkzeuge der pecBOX dar. Der Erfolg zeigt, dass diese Probe bestanden wurde: Im Jahr 2016 wurde das migrierte System produktiv geschaltet [2]. CoJaC erreichte dabei einen Automatisierungsgrad von ca. 95%, d.h., 95 von 100 COBOL-Statements konnten automatisch ohne manuelle Nachbearbeitung konvertiert werden.

4 Wer anderen eine Grube gräbt ...

In einem Migrationsprojekt spielen auch „soziale“ Aspekte eine Rolle. Dazu gehört z.B. auch die „Migration der bisherigen Mitarbeiter“. Dies betrifft z.B. Schulungen mit dem Ziel, dass diese die modernisierte Software selber warten und weiterentwickeln können. Die fachliche Logik ist den bisherigen Entwicklern bekannt, so dass noch eine Einarbeitung in die neuen Sprachen und Entwicklungsumgebungen notwendig ist. Da einige Entwickler des Kunden altersbedingt ausschieden, entschloss sich der Kunde, externe Unterstützung bei Wartung und Pflege einzubinden. pro et con erhielt bei der Ausschreibung den Zuschlag und kam nun in den „Genuss“, gemeinsam mit den verbliebenen Mitarbeitern des Kunden den migrierten Java-Code zu pflegen und weiterzuentwickeln. Das stellte eine Herausforderung dar, da eine Entwicklungstätigkeit eine tiefere Einsicht in die fachlichen Zusammenhänge erfordert, als sie für die Migration notwendig war. Nach einer intensiven Einarbeitungsphase, in der das Fachwissen der ursprünglichen COBOL-Entwickler „abgeschöpft“ wurde, zeigte sich, dass der generierte Java-Code gut gewartet werden kann. Alle Anforderungen des Kunden wurden in Zusammenarbeit zwischen Entwicklern des Kunden und pro et con fristgerecht realisiert. Die Wartungstätigkeit deckte jedoch auch Verbesserungspotenzial auf. Unter anderem besteht dieses bei der Migration von COBOL-Datenfeldern. Um das Verhalten von COBOL im Zielsystem zu emulieren, wird dafür Code in den Java-Quelltext generiert. Bei Änderung des Datentyps oder beim Einfügen neuer Felder muss dann an mehreren Stellen geändert werden, um die Funktionalität weiterhin zu gewährleisten. Zusätzlich entstehen durch die Migration lange Bezeichner. Ursache ist, dass in COBOL bei der Namensbildung übergeordnete Zwischenstrukturen weggelassen werden können, solange der Name eindeutig ist. Java bietet diese Möglichkeit nicht, und die dadurch entstehenden Bezeichner erschweren zuweilen das Lesen des Quellcodes. Als Lösung ist ein Refakturierungswerkzeug vorgesehen, welches optional lokale Variablen einführt, die dann einen *short name* zu einem längeren Bezeichner bereitstellen und diesen in den Ausdrücken ersetzen.

Zusammenfassend kann festgestellt werden, dass sowohl die „migrierten“ Entwickler des Kunden als auch die Mitarbeiter von pro et con den konvertierten Java-Sourcecode warten und weiterentwickeln können. Dieses Projekt widerlegt also erneut häufig geäußerte Bedenken, automatisch konvertierter Code wäre nicht wartbar.

5 Vorwärts zu neuen Tools

Die Entwicklung schreitet fort und auch bei pro et con wurden in der Zwischenzeit weitere Migrationsprojekte bearbeitet. Die Erfahrungen vorangegangener Projekte flossen dabei mit ein, so dass sich der Aufwand für eine manuelle Vor- oder Nachbearbeitung ständig reduziert. Trotzdem gleicht kein Migrationsprojekt dem anderen und es sind immer Anpassungen und „Feinjustierungen“ der Werkzeuge und der Technologie erforderlich. Dieser Aufwand wird häufig von Kunden unterschätzt. Die Werkzeugentwicklung ist mit der Existenz der pecBOX nicht abgeschlossen. Neben der projektbezogenen Vervollkommnung ist auch die Entwicklung neuer Werkzeuge geplant. Fokussiert wird u.a. auf die Phase vor der eigentlichen Migration. Der zu migrierende Programmbestand ist entsprechend von Vorgaben umzustellen, um die Wartbarkeit des generierten Zielcodes zu verbessern bzw. eine Konvertierung überhaupt erst zu ermöglichen (*Sanieren vor Migrieren*). Im „Migrationswortschatz“ von pro et con hat sich hierfür der Begriff „Preengineering“ (Reengineering vor der eigentlichen Migration) durchgesetzt. Auf Zielcodeebene soll die Wartbarkeit des generierten Codes toolgestützt verbessert werden. Für die migrierten Datenstrukturen ist z.B. die Entwicklung eines Eclipse-Plugin geplant, welches einen grafischen Editor für diese Java-Klassen bereitstellt. Ihr Quelltext wird mit Hilfe des *Java Development Toolkits* (JDT) analysiert, die Datenfelder und -strukturen werden baumartig visualisiert. Diese können dadurch komfortabel und redundanzfrei bearbeitet werden.

pro et con wird also auch in den nächsten Jahren bis zum 30. WSRE über Werkzeuge, Technologien und Erfahrungen in der Software-Migration berichten können.

Literaturverzeichnis

[1] Erdmenger, U.; Kaiser,U.; Loos, A.; Uhlig, D.: Methoden u. Werkzeuge für die Software Migration. In: Gimnich, R.; Kaiser, U.; Quante, J.; Winter, A. (Eds.): Lecture Notes in Informatics, (LNI)-Proceedings, Volume P-126, S. 83-97

[2] Uhlig, D.; Fricke, H.: Beschwipste Zebras und Biergartenbiber - ein Projektbericht. Softwaretechnik-Trends, Band 36, Heft 2, Mai 2016

[3] Erdmenger, U.: Oberflächenmodernisierung mit MaTriX. Softwaretechnik-Trends, Band 32, Heft 2, Mai 2012