Migration des SüdLeasing COBOL-Kernbanken-Systems nach JAVA mit einem iterativ-inkrementellen Ansatz

Michael Maleika, Sebastian Seek
SüdLeasing GmbH, Pariser Platz 7, 70173 Stuttgart

Abstract

Anfang 2019 hat die *SüdLeasing GmbH,* eine der führenden herstellerunabhängigen Leasing-Gesellschaften in Deutschland mit 21 Standorten, das Projekt „Technisches Reengineering Bestandssystem *LEASCO"* begonnen. Ziel war es, die Programmiersprache COBOL zu ersetzen und ein zukunftsfähiges System zu erhalten – Unter Einbehaltung sämtlicher Qualitätsvorgaben in einem agilen Projekt. Der folgende Beitrag geht insbesondere auf die Organisation des Gesamtvorgehens ein und spart die technischen Details der eigentlichen Migration von COBOL nach Java aus. Diese Details werden im separaten WSRE2022-Beitrag der *pro et con Innovative Informatikanwendungen GmbH* detailliert erörtert.

Ausgangssituation

Das System *LEASCO* war ursprünglich eine Standardsoftware, die seit über zwei Jahrzehnte in Eigenregie der *SüdLeasing GmbH* weiterentwickelt wird. Zu Beginn des Projekts bestand LEASCO aus über 5.000.000 Zeilen COBOL, die sich auf ca. 1.400 COBOL-Programme verteilen, und knapp 500.000 Zeilen PL/SQL-Code. Zudem bietet *LEASCO* ca. 1.900 Masken und enthält über 1.500 Datenbank-Tabellen. Wegen der großen Menge an Code und einer knappen Projekt-Zeitleiste wurde zu Beginn des Projekts beschlossen, dass ein manuelles Refactoring nicht in absehbarer Zeit zum Erfolg führen wird. Fachlich sind in *LEASCO* sowohl das Haupt- und Nebenbuch der *SüdLeasing GmbH* verortet. Ebenso, findet die gesamte Vertragsverwaltung im System statt. Aus diesem Grund war es für die Planung des Projekts unabdingbar, Weiterentwicklungen betreiben zu können. Und dies parallel zur Modernisierung unter Einbehaltung sämtlicher Qualitätsvorgaben der Aufsicht, Revision und Qualitätsprozesse.

Vorgehen des Modernisierungsprojekts

Zu Beginn des Projekts sind das gesamte System und die Systemlandschaft analysiert worden. Ergebnis für das Vorgehen im Projekt waren folgende Schritte:

1. Aufräumen von nicht verwendeten Code
2. Erneuerung des Frontends – Abschaltung der alten Oberfläche
3. (Automatische) Migration von COBOL nach JAVA

In Schritt 1 wurden sämtliche COBOL-Programmaufrufe ausgewertet. In Abhängigkeit zum letzten Aufruf (noch nie, länger als 1 Jahr, länger als 2 Jahre) wurde eine priorisierte Liste erstellt. Die Programme, die noch nie verwendet wurden, konnten relativ zeitnah wegarchiviert werden. Die restlichen Programme sind in Abstimmung mit den Fachbereichen analysiert und ebenfalls entfernt worden. Auf diese Weise konnte knapp die Hälfte des COBOL-Codes eingespart werden. Dies war möglich, da *LEASCO* als Standardsystem verschiedene Module anbietet, die die *SüdLeasing GmbH* jedoch noch nie verwendet und benötigt hat. Der übrige Source-Code stellt die Ausgangslage für die Gesamtmigration dar. Die neue Web-Oberfläche ist in Schritt 2 auf Basis von AngularJS, Kotlin und Spring Boot als Client-Server-Applikation entwickelt worden. Sie ersetzte die bestehende Windows-Applikation, die in den 90er Jahren als Ersatz für die ursprüngliche ASCII-Terminal-Bedienung entwickelt worden war. In Schritt 3 wurde nach einer Herstellerauswahl von Anbietern, die eine automatische Migration von Cobol nach Java unterstützen, die *pro et con Innovative Informatikanwendungen GmbH* als Partner für das Migrationsprojekt ausgewählt.

Proof of Concept

Nach dem Auswahlprozess eines geeigneten Partners zur Unterstützung unseres Vorhabens, galt es zunächst den Ansatz des Dienstleisters zur Übersetzung des Cobol Codes nach Java zu validieren. Bevor ein komplexes und kostenintensives Projekt gestartet werden kann, muss sichergestellt sein, dass das anschließende Ergebnis auch den eigenen Erwartungen entspricht. Dies bedeutete, dass ein erster Test zunächst zeigen musste, dass nach der Migration die entsprechenden Programme in Java 1:1 genauso funktionieren und ebenso performant sind wie in Cobol. Hierzu haben wir uns eines der wichtigsten und zugleich ein relativ großes und komplexes Massenverarbeitungs-Programm herausgesucht. Vorteil hiervon war, dass wir sehen konnten, ob die enthaltenen Rechenoperationen korrekt ausgeführt wurden. Zugleich konnten wir prüfen, ob der verhältnismäßig lange laufende Prozess eine vergleichbare Laufzeit aufweist. Die Ergebnisse des POCs haben die Vorgaben ausreichend erfüllt. Daher ist nahtlos in die Planung und Durchführung des Gesamtprojekts übergegangen worden.

Die Migrationspakete

Um ein agiles und handhabbares Projektvorgehen zu erreichen, ist der gesamte COBOL-Code in Pakete aufgeteilt worden. Die erste Idee war es hier, die Pakete fachlich zu schneiden. Der erhoffte Vorteil war,

abgesteckte Bereiche zu erstellen, die dann wiederum unabhängig voneinander getestet werden können. Im Projekt hat sich schnell herausgestellt, dass es zu viele Überschneidungen zwischen den fachlichen Prozessen gibt, weswegen Programme nicht eindeutig einem fachlichen Prozess zugeordnet werden konnten. Die Vorgehensweise wurde somit von einer prozessbasierten in eine programmbasierte abgeändert. Die über 1.000 Cobol-Programme und Funktionen sind in eine Liste überführt worden, die dann von oben nach unten durchgearbeitet werden konnte. Ein Paket enthielt am Ende ca. 120-150 Programme und wurde innerhalb eines vierwöchigen Sprints umgesetzt. Für den gesamten Code sind 14 Migrationspakete erstellt worden. Diese sind dann innerhalb von knapp 12 Monaten umgesetzt worden. Ein großer Vorteil bei diesem Vorgehen, im Vergleich zu klassischen Reengineering Projekten, lag darin, dass wir regelmäßig Feedback zum Vorgehen und über die Qualität des generierten Java-Codes bekamen. Hierdurch konnte Fehler schnell erkannt und bei künftigen Migrationspaketen vermieden werden.

Testautomatisierung

Hunderte von Masken, Programmen und Funktionen mussten regelmäßig getestet werden. Die Entscheidung war, jedes einzelne migrierte Paket direkt im Anschluss nach Auslieferung zu testen. Auf diese Weise konnten Auffälligkeiten/Fehler direkt beseitigt und in zukünftigen Paketen vermieden werden. Aufgrund der großen Menge an benötigten Testfällen und der Notwendigkeit diese mehrfach ausführen zu können, kamen hierfür nur automatisierte Tests in Frage.

Die Umsetzung der Testautomatisierung erfolgt mittels Cucumber. Die Testfälle wurden zunächst in Zusammenarbeit mit den Fachbereichen in Gherkin formuliert und anschließend in Kotlin ausdetailliert bzw. programmiert. Die Ausführung erfolgte browserbasiert über Selenium. Zu Beginn wurde sich auf einfache Testfälle wie Maskenaufrufe und Ausführungen simpler Prozesse fokussiert. Diese wurden entsprechend der in den Paketen umgesetzten Programme parallel zur Migration erstellt, damit sie direkt nach Fertigstellung der Pakete für Tests verwendet werden konnten. Nach Abschluss sämtlicher Migrationsarbeiten belief sich das Repertoire an automatisierten Testfällen auf über 1.000 Stück. Neben den simplen Aufrufen wurden darüber hinaus im Laufe des Projekts auch immer komplexere Fachbereichstests abgebildet.

Abnahme des Java Codes

Zur Sicherstellung eines funktionierenden Systems sind ergänzend zu den automatischen Tests regelmäßige Fachbereichstests eingeplant worden. Die Abnahme der migrierten Software erfolgte über einen zweistufigen Abnahmetest. Der erste erfolgte direkt nach Beendigung der Migration des gesamten Codes und unter der Bedingung, dass sämtliche automatisierte Tests erfolgreich waren. Diese erste Feuertaufe beinhaltete ca. 250 Testfälle mit über 1.000 Testschritten. Im Anschluss an die erste Test-Phase folgte eine ca. zwei-monatige Defect-Fixing-Phase, in der Fehler beseitigt und identifizierte Performance-Probleme behoben werden konnten. Als letzte Hürde vor der finalen Abnahme gab es einen zweiten Abnahmetest. Dieser musste ohne produktionsverhindernde Defects abgeschlossen werden, damit die Fachbereiche grünes Licht für eine erste Produktivnahme mit Pilotusern geben konnten. Die nächsten Wochen und Monate werden mehr und mehr User auf die neue Produktivumgebung freigeschaltet, bis in der Endausbaustufe das alte COBOL-Backend abgeschaltet werden kann.

Zusammenfassung

Für ein Migrationsprojekt dieser Größe ist es unabdingbar, Wege zu finden, häufig und schnell neue Erkenntnisse über die migrierte Software zu erhalten. Aus diesem Grund wurde ein agiles Vorgehen gemeinsam mit der *pro et con* beschlossen. Das bedeutete konkret:

- Das Aufteilen des Source-Codes in mehrere kleinere Pakte
- Der Aufbau von Testfällen, die automatisiert bei jedem Deployment durchlaufen können
- Das schnelle Ausliefern, Deployen und Testen neuer Software-Pakete

Durch das gemeinsame Projektvorgehen zwischen der *SüdLeasing GmbH* und der *pro et con* konnte das gesamte Team kontinuierlich arbeiten und sinnvoll ausgelastet werden. Dies hat eine sehr hohe Qualität sichergestellt und das Vertrauen des Fachbereichs in das neue System gestärkt. Ebenfalls konnte der Leistungsumfang der *pro et con* direkt erweitert werden, wenn sich neue Anforderungen oder alternative Lösungsansätze im Detail herauskristallisiert haben. Dies wäre bei einem konventionellen Wasserfall-Ansatz schwer möglich gewesen. Die intensive Zusammenarbeit zusammen mit dem Dienstleister hat sich sehr bezahlt gemacht und wird auch in zukünftigen Projekten der SL im vergleichbaren Maß angestrebt.

Zum Zeitpunkt der Einreichung dieser Veröffentlichung ist das Projekt nahezu abgeschlossen. Auf dem produktiven System arbeiten bereits zwei Drittel aller Nutzer und - mit Ausnahme von einigen komplexen Massenverarbeitungsläufen - werden sämtliche Prozesse über Java verarbeitet. Der nächste Schritt ist das Abschalten von COBOL LEASCO und das Starten erster Weiterentwicklungen am neuen Java-System.