



**SOZIALVERSICHERUNGSANSTALT
DER GEWERBLICHEN WIRTSCHAFT**

REIFEN WECHSELN AM FAHRENDEN AUTO

Kostenoptimierung durch Migration

Thomas Mück

Sozialversicherungsanstalt der gewerblichen Wirtschaft

23. März 2010

Projektmotivation und Ausgangssituation in der SVA

- **Bereich Infrastruktur HW:** zOS Komponenten erreichten das Ende des Einsatzzeitraumes
- **Bereich Infrastruktur SW Stack:** hohe Lizenzkosten, auch signifikante upgrade Notwendigkeiten
- **Bereich Entwicklung:** die verwendeten Entwicklungswerkzeuge waren “Exoten” oder wurden vom Hersteller bereits aus der Wartung genommen. Die Anwendung agiler Methoden war nicht möglich.

daraus resultierten:

- **spürbare Herstellerabhängigkeiten:** Erfordernis von Spezialwissen, das nicht mehr dem aktuellen IT-Ausbildungsstand entspricht
- **somit: hohe Entwicklungskosten**
- **schleichender Know-How-Verlust:** Know-How-Träger gehen vermehrt in den Ruhestand. Die damit verbundenen Engpässe stellten ab 2005 ein gravierendes Wartungsrisiko für die die Core-Applikationen dar.

Die Analysten der Gartner Gruppe sprechen von „*inherent lack of agility in established IT-Systems*”

Fragen im Vorfeld des Migrationsprogramms:

- Wie kann das Investitionskapital, das im gesammelten Know-How der komplexen, spezialisierten Geschäftsprozesse steckt, weiter genutzt werden?
 - reines re-platforming?
 - Codetransformation? Wie denn?

- Wie kann das “unternehmessoziologische” Projektrisiko minimiert werden?

- Wie kann man **methodisch** von einer nicht mehr adäquaten Plattform auf eine betriebswirtschaftlich sinnvolle Plattform umsteigen?

- Kann das re-platforming und/oder die Codetransformation dieser Systeme auch **effizient** erfolgen?

- Welche Lösung mit welchem Projektpartner deckt die aktuellen Anforderungen in einem überschaubaren Zeit-, Kosten und Risikorahmen ab?

Grundsätzlich mögliche Alternativen:

■ **Neuentwicklungen auf der Zielplattform:**

- aufwändige, schwer steuerbare Großprojekte mit ungewissem Ausgang
- ausgereifte Applikationen werden sodann durch Software ersetzt, deren Bewährungsprobe noch aussteht
- Entwicklungsabteilung im besten Fall lange blockiert, im schlechtesten Fall überfordert

■ **Einführung einer Standardlösung:**

- spürbare Kosten für Implementierung und Anpassung
- in weiten Bereichen der Kernprozesse (Beispiel **Beitrag**) schlicht nicht existent

✓ **Somit: re-Platforming und Codetransformation in dieser Situation die beste Lösung**

- ✓ **Potenziale in Sachen cost cutting heben (Infrastrukturkosten!)**
- ✓ **mit knappen Projektmitteln einen möglichst großen Effekt erzielen**

Unmittelbarer Nutzen von re-Platforming und Codetransformation

- ✓ die eigentliche Applikationslandschaft (GUI-Struktur, Geschäftsprozesse, Bearbeitungsschritte, ...) bleiben erhalten.
- ✓ das Ergebnis sind funktional unveränderte, zukünftig leicht integrierbare (SOA) Applikationen
- ✓ durch die Verwendung einer open-source basierten Mainstream-Umgebung (Java, Eclipse, Jboss, ...) können
 - ✓ Lizenzkosten signifikant gesenkt, aber auch
 - ✓ voll ausgebildete Mitarbeiterinnen aquiriert werden

Die Partner-Entscheidung nach Vergabeverfahren:

SHARK GmbH als Dienstleister, der einen Codegenerator (de facto Compiler) zur Implementierung beider methodischen Ansätze anbietet:

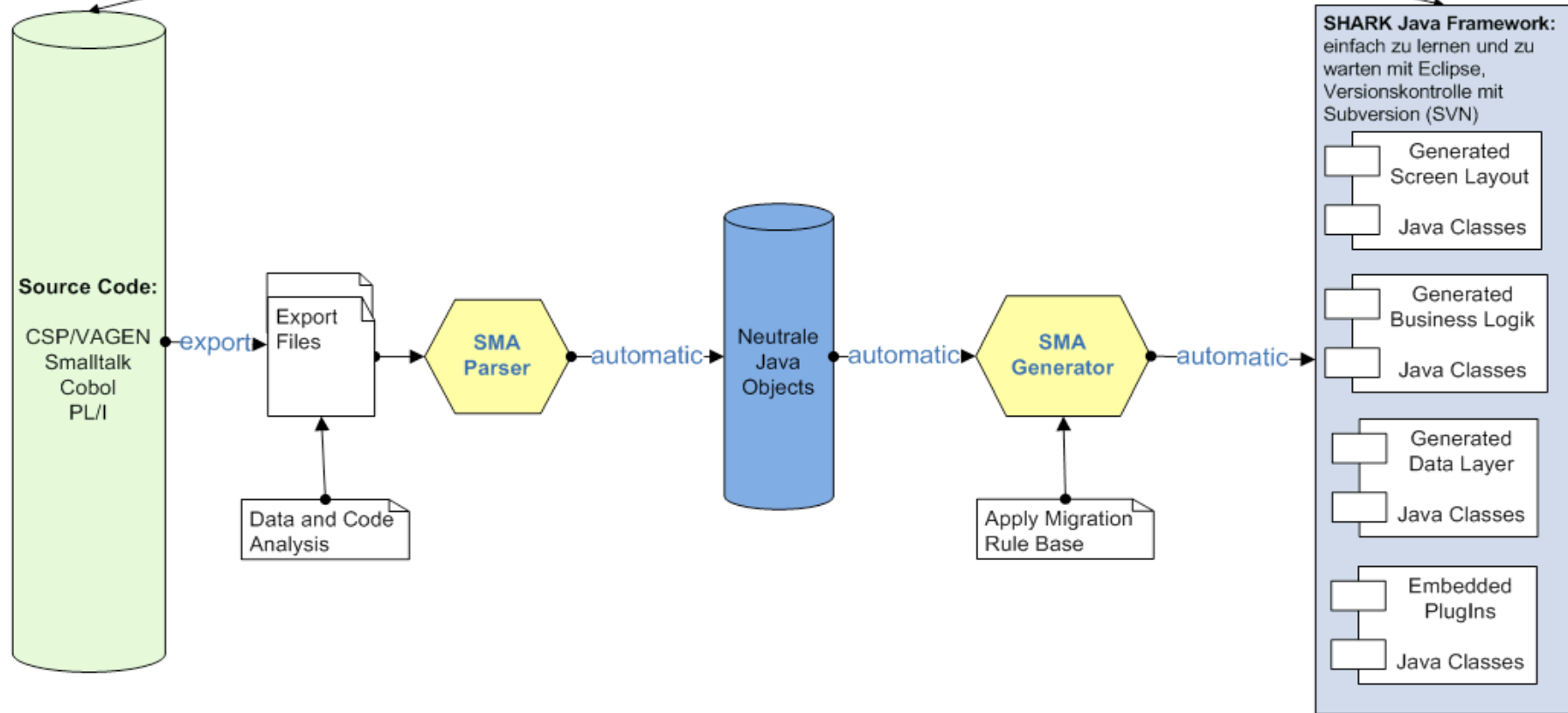
- Migration durch automatische Quellcode-Transformation (z. B. “Smalltalk” nach Java)
- Migration durch automatisches Re-Platforming

Charakteristika:

- der Generator übersetzt den Source Code in bedeutungsgleichen Java Code, der den ursprünglichen Code als Kommentar enthält. Die MitarbeiterInnen finden sich dadurch im Java-Code gut zurecht, die Lernkurve war erfreulich.
- eine wartungsfreundliche open source basierte Zielumgebung wurde in ein Architekturframework integriert, das an die Richtlinien des Hauptverbandes angepasst wurde.
- die Integration und Übernahme der zugrunde liegenden Datenbank erfolgte problemlos
- es wurden plugin Komponenten zur Verfügung gestellt, die UmsteigerInnen schnellst möglich produktives Arbeiten erlauben

Der Migrationsprozess im Überblick

100% automatische Generierung ohne manuelle Nachbearbeitung ermöglicht:
Generierungszyklus kann beliebig oft wiederholt werden
Minimale Code Freeze Perioden



.... ein standardisierter Prozess machte es möglich

Die einzelnen Migrationsschritte laufen standardisiert ab. Lediglich der regelbasierte Generator wird angepasst und garantiert ein maßgeschneidertes Ergebnis:

- ✓ **1. Schritt:** Die vorbereitende Analyse ist eine reine Daten- und Codeanalyse und deshalb weitaus kürzer als bei Neuentwicklungen. Im Zuge dessen wird der Code auf versteckte Altlasten überprüft und diese im Rahmen von automatischem Code-Rewrite beseitigt.
- ✓ **2. Schritt:** Optimierung und Anpassung der Regeln des Generators an die Architekturvorgaben
- ✓ **3. Schritt:** Die eigentliche Generierung erfolgt vollautomatisch, ohne jegliche manuellen Eingriffe in einem einzigen Durchlauf
- ✓ **4. Schritt:** Der generierte Code wird nicht manuell nachbearbeitet, sondern gegebenenfalls die Regeln des Generators angepasst

Die Generierungsabläufe können beliebig wiederholt werden (die Gesamtgenerierung von 12.000 Java Klassen in der SVA benötigte nur 12 Minuten)

... und den Projektfortschritt transparent!

- ✓ Das **Projektrisiko ist gering**, da bereits in der Vorprojektphase der gesamte Generierungsablauf an repräsentativen Code-Beispielen durchgängig demonstriert werden kann
- ✓ Das Fehlen von manuellen Eingriffen macht den Migrationsprozess gut planbar
- ✓ **Neue MitarbeiterInnen** mit aktuellem IT-Ausbildungsstand können die Zielarchitektur ohne zusätzlichen Ausbildungsaufwand einfach betreuen und warten
- ✓ **Bewährten MitarbeiterInnen** fällt der Umstieg auf die neue Entwicklungsumgebung durch die Referenzierung auf bestehende Strukturen leichter (!)
- ✓ für die **ca. 1200 EndanwenderInnen** war die Migration nahezu problemlos. Die Sachbearbeiter finden den gewohnten 'look and feel' vor und können ohne Einschulung auf der neuen Oberfläche arbeiten
- ✓ **Minimaler Testaufwand**, da nur technische Tests erforderlich sind – funktionale Tests können entfallen

Die Umsetzung: Das 3-stufige Migrationsprogramm

Im Rahmen der 3 Umsetzungsprojekte über die Ablöse und Konsolidierung der historisch gewachsenen Systemlandschaften kamen beide methodischen Ansätze zur Anwendung: Migration durch **vollautomatische Quellcode Transformation** und **Re-Platforming** auf Basis von Java unter Linux/Unix:

- **Projekt SVA-JIS: Quellcode-Transformation der SVA-IS und 3270-Online-Dialogsysteme (CSP/VAGEN → Java/Linux/JBoss)**
→ in Time and Budget: 10 / 2005 – 06 / 2006
- **Projekt PLI/Online: Re-Platforming des TP-Monitors (CICS-Middleware)**
(CICS z/OS → Linux/AIX/JBoss)
→ in Time and Budget: 07 / 2006 – 12 / 2006
- **Projekt B2U: Re-Platforming der Batchprogramme, Datenbanken und des rechenzentrumsbasierten Batchbetriebs**
(PL/I z/OS → PL/I AIX, DB2 z/os → DB2/AIX. ISPF, SDSF → UC4)
→ in Time and Budget: 02 / 2007 – 12 / 2007

Die herausragenden Merkmale der 3 Umsetzungsprojekte:

- der **Return on Investment (ROI)** blieb für alle 3 Projekte **unter einem Jahr**
- die **regelbasierte, automatische Migration** in einem einzigen Schritt
- die **minimalen Code-Freeze Perioden** (im Projekt B2U lediglich ein Tag!)
- die Gewährleistung eines **störungsfreien Produktionsbetriebes** während sämtlicher Umstellungen
- die Einhaltung der **Architekturvorgaben** (TA3.0) des Hauptverbandes der österreichischen Sozialversicherungsträger
- die **vollständige Wart- und Weiterentwickelbarkeit der migrierten Software**
- die strenge Einhaltung anerkannter Richtlinien (IPMA) im Projektmanagement und in der Qualitätssicherung

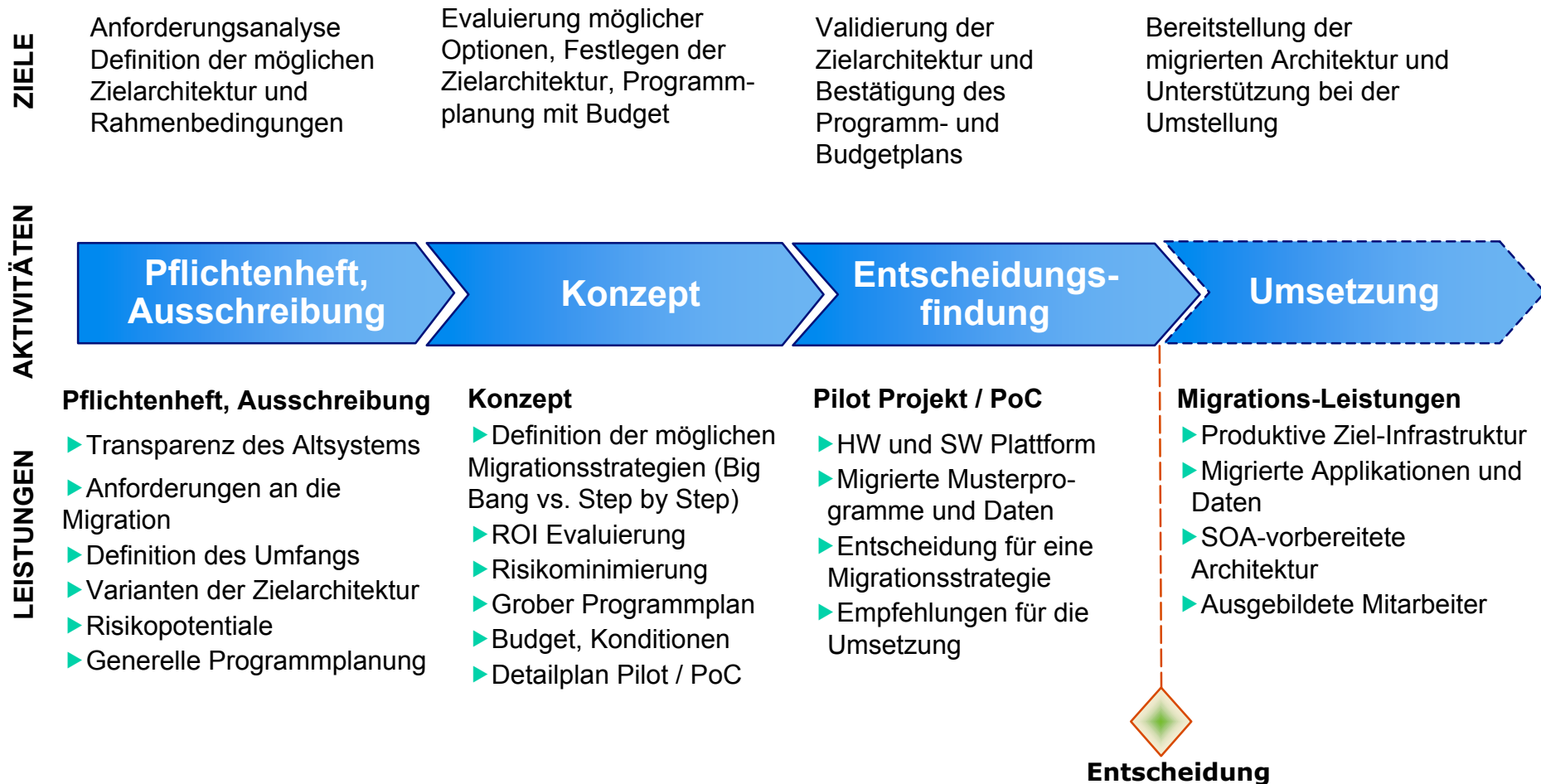
Die Systemlandschaften vor und nach der Migration

	Source Environment: IBM OS/390	Target Environment: Unix
Hardware	IBM z/OS	Bull: 4 x P5, 2.1 GHZ, 10 GB 16 x AMD 2.2, 32 GB
Operating System	OS/390	Bull AIX 5.3 / Linux RedHat AS 5.0
TP Monitor	CICS	Linux / Jboss / AIX
Databases	DB2 z/OS	DB2
Files	VSAM	Linux
Language	CSP / VAG / PL/I unter z/OS	Java / Linux, PL/I unter AIX
User Interface	3270/Smalltalk GUIs	Java GUIs, Jboss
Cmd Language	REXX, JCL	KSH / Bash, Object REXX
Job scheduler	ISPF, SDSF	UC4: Global, Ksh
Sort tool	JCL, IBM DFSORT	C-Addons Shark

Migration-Road-Map

Investition

Ertrag



Zusammenfassung

- im Hinblick auf **Kosten-Risiko-Überlegungen** und **TCO** rechnete sich der Wechsel für die SVA sehr schnell
- geschäftskritischer Code musste nicht neu geschrieben werden
- die Evaluierung erfolgte in der Vorprojektphase, das Risiko der Migration war daher sehr gering
- die Wartung des bisherigen Systems konnte während der gesamten Projektdurchlaufzeit in gewohnter Form erfolgen (**minimale Code-Freeze Perioden** im Altsystem, keine Zweigleisigkeit, störungsfreier Produktionsbetrieb während der Umstellung)
- um **Kontinuität, Sicherheit und Verfügbarkeit** der neuen Anwendung sicherzustellen wurde nach Projektabschluss der source code des individuellen SHARK-Java-Frameworks für die weitere **exklusive Nutzung** zur Verfügung gestellt.
- die Weiterentwicklung erfolgt mittels Eclipse und den darin integrierten Open Source Projekten – dadurch wird möglich **herstellerunabhängig und kostenschonend** zu agieren
- es kann sowohl ein „fat client“ (1:1 beibehalten des look and feels) als auch eine Web-Lösung generiert werden. Die jeweils dazugehörige Infrastruktur für den Betrieb wird mitgeliefert.
- **im Vergleich zum Altsystem durchwegs höhere Performance der generierten Anwendungen!**



**SOZIALVERSICHERUNGSANSTALT
DER GEWERBLICHEN WIRTSCHAFT**

Vielen Dank für Ihre Aufmerksamkeit !

Univ. Doz. Dr. Thomas Mück
Generaldirektor Stv. SVA

thomas.mueck@svagw.at