pro et con
Analyse
Reengineering
Migration

# - FAQ -

# Automatisierte Sprachtransformation von COBOL nach Java unter Nutzung des Werkzeugs „CoJaC“ im Rahmen eines Software-Migrationsprojektes


**pro et con**
Innovative Informatikanwendungen GmbH

Reichenhainer Straße 29a
09126 Chemnitz

Telefon: +49 371 270951-0
Telefax: +49 371 270951-29
Internet: www.proetcon.de
E-Mail: info@proetcon.de

## 1. Funktioniert das überhaupt automatisch? Habe ich da nicht zu viel Handarbeit?

Wir haben in mehreren Migrationsprojekten bewiesen, dass das funktioniert (z. B. SPL nach C++ bei Amadeus). Bei COBOL nach Java können große Teile des COBOL-Codes (ca. 80–90 %) automatisch konvertiert werden. Das Werkzeug „CoJaC“ (COBOL to Java Converter) erzeugt dabei semantisch äquivalenten Java-Code. Die restlichen 10–20 % werden z. B. durch geeignete Kommentare deutlich ausgewiesen, was eine manuelle Konvertierung erleichtert.

## 2. Entsprechen die konvertierten Java-Programme einem objektorientierten Entwurf?

Nein, die COBOL-Programme wurden ja in einer prozeduralen Sprache implementiert. Eine Neuaufteilung der Funktionalität auf einzelne Klassen und Methoden im Sinne des objektorientierten Entwurfs ist automatisiert nicht möglich. Das ist auch nicht unbedingt wünschenswert. Schließlich müssen die Entwickler den Code auch wiedererkennen. Eine zukünftige Weiterentwicklung der generierten Java-Applikation kann natürlich objektorientiert erfolgen.

## 3. Ist der entstehende Code wartbar und zur Weiterentwicklung geeignet?

Ja, definitiv. Der Grundaufbau des generierten Java-Codes ist analog zum COBOL-Code. Dadurch steigt der Wiedererkennungseffekt. Auch Kommentare werden optional an die korrekten Stellen des Zielcodes eingefügt. Die Entwickler des Altsystems finden sich (Java-Kenntnisse vorausgesetzt) schnell zurecht. Es gibt Referenzprojekte für die automatische Sprachtransformation (z. B. SPL nach C++), die praktisch beweisen, dass der generierte Code wartbar und performant ist.

## 4. Welche COBOL-Anweisungen sind nicht vollständig konvertierbar?

Dabei handelt es sich um solche Anweisungen, für die Java die entsprechenden sprachlichen Mittel nicht zur Verfügung stellt. Das betrifft vor allem die sogenannten „compiler-directing statements“ (`COPY`, `REPLACE`, ...). Diese werden erst bei der Übersetzung des COBOL-Programms ausgeführt, sozusagen in einer Präprozessor-Phase. Java stellt aber keinen Präprozessor zur Verfügung. Somit sind diese Anweisungen nicht mit den üblichen Mechanismen konvertierbar. Das bedeutet jedoch nicht, dass z. B. keine `COPY`-Strecken konvertiert werden können. Enthält die `COPY`-Strecke z. B. komplette Datenstrukturen (was in 90 % der Fälle zutrifft), kann sie durchaus in ein separates Java-File konvertiert werden.

## 5. Entstehen bei der Migration nicht viele „Java-untypische“ Konstrukte, so dass es sich um ein COBOL-Programm in Java-Notation handelt?

Die Abbildungsvorschriften von COBOL-Konstrukten in Java-Konstrukte wurden von uns bewusst so gewählt, dass der entstehende Java-Code Java-typisch ist. Es werden z. B. für die verschiedenen Datentypen Java-Klassen in einem Laufzeitsystem zur Verfügung gestellt, die alle notwendigen Informationen, wie Länge etc., kapseln und Methoden zu deren Verwaltung anbieten.

## 6. Erkennen meine Programmierer den Quellcode wieder? Wie groß sind die strukturellen Änderungen?

Die prinzipielle Struktur eines Programms bleibt identisch. Aus einem Programm wird eine Klasse, aus Datenstrukturen die (privaten) Datenfelder dieser Klasse und aus einzelnen `SECTION` der `PROCEDURE DIVISION` werden die Klassen-Methoden. Dabei bleibt die Reihenfolge im Quelltext bestehen. Strukturelle Änderungen gibt es nur bei den „compiler-directing statements“, welche in Java unbekannt sind.

### 7. Ist es dann nicht besser, den Code gleich in Java neu zu entwickeln?

Neben den dabei anfallenden, wesentlich höheren Kosten sollten Sie bei dieser Entscheidung auch die Projektdauer beachten. Wenn 90 % des Java-Codes automatisch generiert werden, ist ein Programm natürlich schneller umgesetzt als von Hand. „Code freezes“ und eventuelle Sperrzeiten bei der Weiterentwicklung/Wartung des Programms werden damit minimiert. Unsere Erfahrungen besagen, dass sich eine Software-Migration zu einer Neuentwicklung im Verhältnis von 1:8 verhält, d. h., wenn Sie 15 Mannjahre für ein Migrationsprojekt unter Nutzung von Transformationswerkzeugen veranschlagen, dann benötigen Sie für eine Neuentwicklung ein- und desselben Programmsystems 120 Mannjahre.

### 8. Können sich Java-Entwickler, die den COBOL-Code vorher nicht kannten und COBOL nicht beherrschen, in die migrierten Programme einarbeiten?

Natürlich. Das ist ein weiterer Vorteil der Software-Migration. Die Entwickler, welche die konvertierten Programme zukünftig warten, müssen kein COBOL kennen. Eine Einarbeitung kann allein anhand des Java-Codes erfolgen.

### 9. Ich möchte aber meine COBOL-Programmierer behalten. Wie hoch ist der Schulungsaufwand beim Umstieg?

Nun, Java müssen sie natürlich lernen. Daran führt kein Weg vorbei. Dafür haben diese Programmierer aber den Vorteil, dass sie sich bereits mit dem fachlichen Hintergrund der Programme auskennen, wogegen sich Neueinsteiger diesen natürlich erst aneignen müssen.

### 10. Sind die konvertierten Programme nicht wesentlich langsamer und verbrauchen viel mehr Speicherplatz?

Heutige JVM arbeiten dank der fortgeschrittenen Technologie der „just-in-time-Compiler“ die Java-Programme nur wenig langsamer ab als der Prozessor die kompilierten COBOL-Programme direkt. Natürlich gibt es einen gewissen Overhead, vor allem, wenn viele `.class`-Files geladen werden müssen. Aber das ist bei Java-Programmen generell so, egal, ob sie generiert oder neu entwickelt wurden. Was den Speicherplatzbedarf angeht, so ist er bei Java größer. Auch das ist unabhängig davon, ob konvertiert oder neu entwickelt wurde. Aber im Zeitalter der 64-Bit-Server und stetig sinkender Hauptspeicherkosten sollte das keine Rolle mehr spielen.

### 11. Wieso ist pro et con als relativ kleines Unternehmen in der Lage, solche Werkzeuge zu entwickeln, „Global Player“ aber nicht?

Einige große Firmen sind natürlich nicht an Migrationsprojekten interessiert, welche „von COBOL weg“ konvertieren. Sie möchten ja ihre COBOL-Entwicklungsumgebung vermarkten. Andere Firmen arbeiten meist nicht strategisch und zielorientiert, sondern strikt auftragsbezogen. Erst wenn ein konkretes Migrationsprojekt eines Kunden gewonnen wurde, wird begonnen, die dafür notwendigen Werkzeuge zu entwickeln, vorhandene anzupassen oder einzukaufen. Werkzeuge für die Software-Migration erfordern aber einen großen Entwicklungsaufwand, der nicht allein im Rahmen eines konkreten Projektes geleistet werden kann. In unserem COBOL-Parser z. B. stecken zwölf Mannjahre Entwicklungsaufwand. Fazit ist, dass die Werkzeuge von großen Firmen dann „zusammengebastelt“ werden und „irgendwie“ funktionieren. Zusätzlich benötigt man für die Werkzeugentwicklung ein ganz spezielles Informatik-Know-how (Compilertechnik). Für uns ist immer wieder interessant, dass dieses Know-how zwar in der Wissenschaft bekannt, aber von großen Firmen in kommerziellen Projekten nicht eingesetzt wird. Wir haben mehr als 20 Jahre Erfahrung in der Entwicklung von Werkzeugen für die Software-Migration. Die Basis bilden Forschungsarbeiten auf dem Gebiet der Compilertechnik an der Fakultät für Informatik der TU Chemnitz. Daraus resultierten mehrere Promotionsarbeiten von Mitarbeitern unserer Firma. Diese wissenschaftlichen Erkenntnisse fließen in die Entwicklung unserer Werkzeuge ein. Darüber hinaus betreiben wir auch heute noch Forschung und Entwicklung in Zusammenarbeit mit Universitäten (Universität Koblenz-Landau, Universität Oldenburg, TU Chemnitz), teilweise kofinanziert in Förderprojekten. Die Ergebnisse spiegeln sich direkt in unseren Technologien und Werkzeugen wider.