FAQ
Language Migration from COBOL to Java

1. Does this really work automatically? Isn't there a lot of manual work to do?

We have proved in several migration projects that this works (e.g. COBOL to Java at ITZBund and SüdLeasing). For COBOL to Java, large parts of the COBOL code (approx. 90–95 %) can be converted automatically. The CoJaC tool generates semantically equivalent Java code. The remaining 5–10 % are clearly indicated, for example, by appropriate comments facilitating the manual conversion.

2. Do the converted Java programs comply with an object-oriented concept?

The COBOL programs were implemented in a procedural language. A redistribution of functionality into individual classes and methods in the sense of the object-oriented concept is not automatically possible. This is not necessarily desirable since the developers must recognise the code. A future, object-oriented further development of the generated Java applications can, of course, be done.

3. Is the generated code maintainable and suitable for further development?

Yes, definitely. The basic structure of the generated Java code is analogous to the COBOL code. This increases the recognition effect. In addition, comments are inserted at the right positions of the target code. The developers of the old system (Java knowledge being the prerequisite) will find their way quickly. There are enough reference projects for automatic language transformation from COBOL to Java, which practically prove that the generated code is maintainable and performant. The available, modern IDEs (e.g. IntelliJ and Eclipse) then allow a comfortable further development.

Concerns are often expressed by competitors who market their own COBOL development environments and are, therefore, not interested in migrating "away from COBOL".

4. Which COBOL statements cannot be converted completely?

Those statements, for which Java does not provide the corresponding language resources, are not completely convertible. This mainly concerns the so-called "compiler-directing statements". These are first implemented during the translation of the COBOL program so to say in a preprocessor step. Java, however, does not provide a preprocessor. Thus, these statements cannot be converted with the common mechanisms. CoJaC includes a sophisticated mechanism to convert copybooks into separate Java classes.

pro et con
Innovative Informatikanwendungen GmbH
Reichenhainer Straße 29a
09126 Chemnitz, Germany

Phone: +49 371 270951-0
Fax: +49 371 270951-29

Internet: www.proetcon.de
Email: info@proetcon.de

5. What about GO TO? There are no jump statements available in Java.

`GO TO` statements can be migrated completely automatically. First, an attempt is made to replace them by Java typical statements. Jumps to the end of a section, for example, are translated into Java by a `return` instruction. If this is not possible, the CoJaC runtime system takes over the flow control. In the program, `GO TO` is then executed by calling a `skip()` method.

6. Isn't it a fact that many "Java-atypical" constructs arise during migration, proving that the result is a COBOL program in Java notation?

The statements for modelling COBOL constructs into Java constructs were deliberately chosen in such a way that the generated code is typical for Java. For example, Java classes encapsulating all necessary information, such as length etc., and offering methods to administrate them are provided for different data types in a runtime system. Where possible, the native Java statements are directly used for loops (`while`, `for`) or `if` statements.

7. Will my programmers recognise the source code? How significant are the structural modifications?

The basic structure of a program remains identical. A program is converted into a class, data structures become the (private) data fields of this class and individual sections of the `PROCEDURE DIVISION` become class methods. In doing so, the order within the source code remains unchanged.

8. Doesn't that result in a lot of clones?

Copybooks are converted into separate classes. Even if these copybooks are used several times, only one class will be created. Clones are only created when using `COPY` with `REPLACING` clause or `COPY` in the `PROCEDURE DIVISION`. We have with JPackage a tool capable of reuniting the resultant clones.

9. Would it not be better to develop codes in Java anew?

In addition to the significantly higher costs, you should also consider the project duration in this decision. If more than 90 % of the Java code is generated automatically, the program is of course converted more rapidly than manually. Code freezes and potential blocking times during further development/program maintenance will thus be minimised. Our experience has shown that the software migration in relation to a new development lies at 1:8, which means that if you calculate 5 man-years for a migration project using transformation tools, you will need 40 man-years for a redevelopment of the same program system.

There are two alternative approaches for a redevelopment: either you rewrite the original program 1:1 in Java, in which case the code will be similar to the automatically migrated code, and you can also automatically migrate; or you extract the business logic of the program, create a specification and then develop the program anew. The extraction of the business logic is however a very time-consuming and error-prone process.

pro et con
Innovative Informatikanwendungen GmbH
Reichenhainer Straße 29a
09126 Chemnitz, Germany

Phone: +49 371 270951-0
Fax: +49 371 270951-29

Internet: www.proetcon.de
Email: info@proetcon.de

10. Can Java developers who did not know the COBOL code before and cannot handle it get used to working with the migrated programs?

Of course, yes. This is another advantage of software migration. The developers maintaining the converted programs in future do not have to know COBOL. They can familiarise themselves with the environment solely on the basis of the Java code.

11. I want to retain my COBOL programmers. How high is the effort for training when switching over?

Well, they have to learn Java. There is no way around this. These programmers, however, have the advantage that they are already familiar with the technical background of the programs, whereas newcomers have, of course, to first acquire the skills.

12. Are converted programs not substantially slower and do need much more memory?

This is also an argument frequently voiced by customers but that has never been found to be true in earlier projects. Performance problems only occur limited to individual programs and could be eliminated by suitable optimisation measures. Thanks to the advanced technology of the "just-in-time-Compiler", today's JVM process Java programs only slightly slower than the processor the compiled COBOL programs.

13. How experienced is pro et con in software migration?

Tools for software migration require extensive development effort. In addition, very specific know-how in informatics (compiler technology) is necessary. We have over 25 years of experience in the development of tools for software migration. The basis is in its years of research work in the field of compiler technology at the Faculty for Informatics at the TU Chemnitz. We still carry out research and development in cooperation with the universities of Koblenz-Landau and Oldenburg. The results are directly reflected in our technologies and tools.

In the course of past years, different types of parser, code generators, formatting and meta tools have been developed which are all combined in our pecBOX (pro et con – Toolbox for Software Migration) and used in our migration projects. No migration project is like the other. For new projects, necessary, new migration tools are compiled from the pecBOX components thus reducing the development time. This generic approach makes migration projects less expensive and reduces the project duration. This has been affirmed by numerous reference customers like Amadeus Germany GmbH, MAN Truck & Bus SE, ITZBund, SüdLeasing GmbH, to name a few.

pro et con
Innovative Informatikanwendungen GmbH
Reichenhainer Straße 29a
09126 Chemnitz, Germany

Phone: +49 371 270951-0
Fax: +49 371 270951-29

Internet: www.proetcon.de
Email: info@proetcon.de