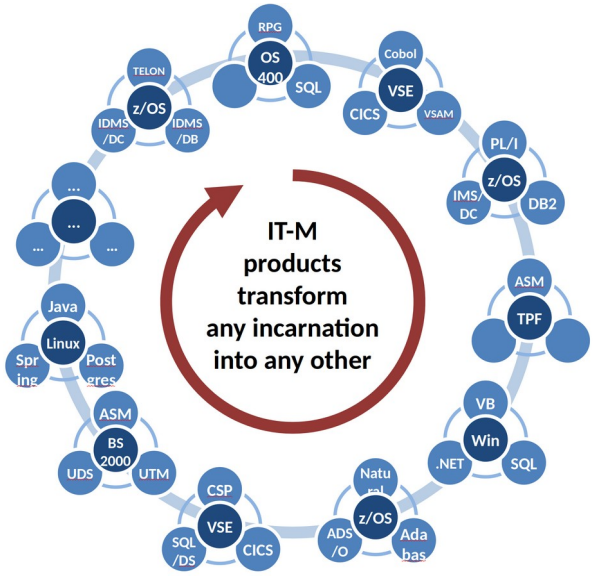
Industrialized Modernization of IT Applications

The pressure to modernize IT application systems is enormous: it is caused by prohibitive costs, lack of staff with legacy programming skills, architectural differences that prevent integration with modern systems, and data held in outdated formats and systems.
ITM has developed Njema, which enables secure, predictable and affordable modernization projects that cover all aspects of modernizing proprietary IT application systems.

Rationale

An existing IT system is only one of many possible embodiments of the inherent business logic: you can change the DB system, the programming language, the operating system, etc. and still achieve the same functionality.



Njema was designed and implemented to support this philosophy.

Code Assessment

Njema creates a comprehensive overview of the application system in a very short time, e.g. 56 million lines were analyzed in just 3.5 hours. Once the code, including job control, scheduler files or third-party product parameters, has been entered into Njema, it knows exactly what is missing or obsolete, where critical code is located and how the objects are connected to each other.

Data Transformation

The assessment revealed everything about the data structures, schemata, views, files and fields. On this basis, Njema is able to move data from classic files to in-memory files or to a database. It can replace the database system and much more, including changes to field names, data formats and field contents.

Supported Products

Over the years, ITM has constantly expanded the range of supported components. We currently support

- Origin languages including PL/I, Cobol, CA-Gen, REXX, NatStar, Informix, Telon,, CA-Earl, Easytrieve, Natural, Pacbase, Uniface
- MVS, VSE, TPF, and BS2000
- Target languages: Java, C++, Cobol
- Mainframe O/S, Linux or Windows.
- CICS, IMS/DC, UTM
- Mainframe Assembler to Cobol[1]
- MVS or VSE Job Control to Bash or another script language
- Schedulers TWS, CA-Schedule, TOM, JobTrac, HS5000
- Oracle, DB2, MySQL, IMS/DB, PostgreSQL, MSSQL, Informix, Sesam, UDS, Adabas
- All classic file systems incl. GDG
- Target frameworks like Spring, WildFly, TomCat, JSP and more

Njema was developed in such a way that new languages, databases, GUI frameworks or customer-specific requirements can be implemented efficiently..

Code Quality

Njema generates clean and easily readable target code: Re-engineering and elimination of dead code are just some of the features applied.
There are no hidden third-party products: all results are delivered as source code.

Flexibility, no one size fits all

Because Njema is rule-based, it can be easily customized to user macros, naming conventions, formatting standards, customer requirements or special situations.

1 See separate flyer.

info@it-modernisation.com

Transformation Technology

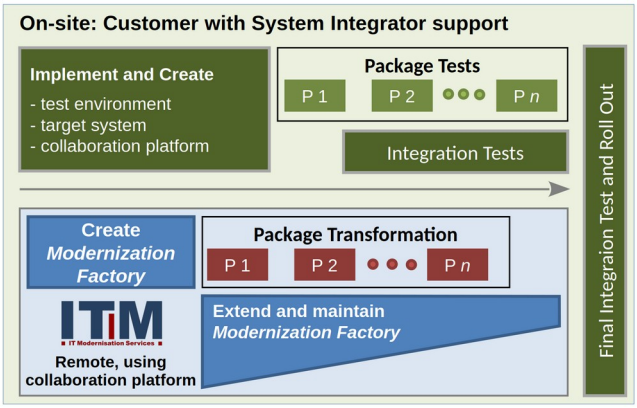
Njema uses a technology package that was expensive and time-consuming to develop. However, the benefits in terms of automation, flexibility and speed in integrating new functions, languages and systems are second to none. The most important factors are:

- Njema does not simply convert one language into another. Instead, Njema converts every single incoming source code statement into the General Intermediate Language (GIL) of ITM.
- GIL is a tree-like structure that addresses all elements from the application to packages and programs to individual instructions.
- In contrast to typical "converters", Njema only starts the transformation after it has analyzed all language objects. The resulting information is entered into the Njema metadata repository, which controls the transformation of source code and data.
- Njema is rule based, which offers enormous flexibility.
- Njema contains a huge and constantly growing collection of rules for processing the code in terms of language, operating system, TP environment, target framework and so on.

Up to date, Njema has processed more than 300 million code lines.

Njema in Projects

Depending on the specific project or customer situation, there are various ways to set up the



modernization project as such.
Here are some general scenarios:

- **Packaged**
  The classic approach, which is particularly unavoidable if the modernization project is not based on a fully automated approach.
  Code and data are split into packages that are transformed and tested independently of each other.
- **Cross-Compile**
  The code is retained in the old language. Njema works like a compiler that converts the code into another language and another environment in which the code can be executed productively.
- **Stepwise Transition**
  Like *Cross-Compile*, but a new development team takes over the maintenance of the converted packages, one by one. This enables a slow and controlled phase-out of the old technology.
- **Big Bang**
  Another classic modernization approach that is often required because code and data are highly intertwined and/or time is short for the transition. The Njema methodology makes this scenario even more attractive than the *Packaged* approach, as it shortens the overall project time and significantly reduces risk and costs.

Each of these scenarios has its advantages and disadvantages.
The project-specific variant is selected on the basis of the analysis results, the future IT architecture, time constraints and many other customer-related aspects.

Terms and Conditions

ITM services include the following:

1. Quick assessment of the costs and effort of the modernization project - calculated by analyzing the customer's entire source code.
2. Complete assessment of all customer code. This identifies missing or obsolete elements and critical code sequences so that you can decide on the future architecture, budget human and technical resources and create a reliable project plan.
3. On-site workshop(s) to explain the Njema method and to support the project.
4. Construction of the project-specific *Transformation Engine,* which provides fully automated and error-free transformation.
5. Support of the customer's on-site test team.

Item 1 of the above list is charged at a fixed price, all others according to time and material.