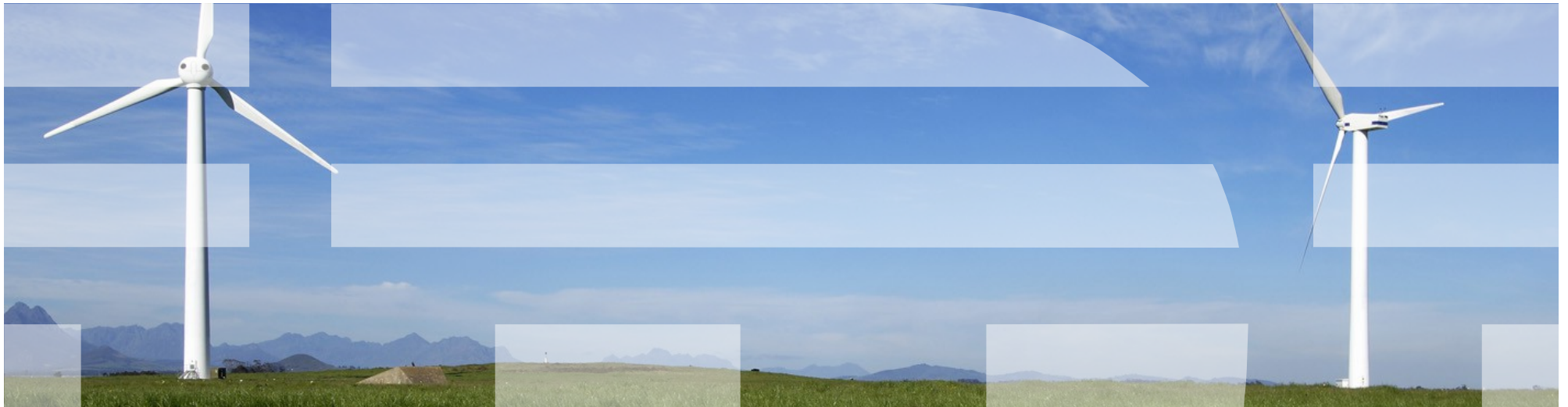
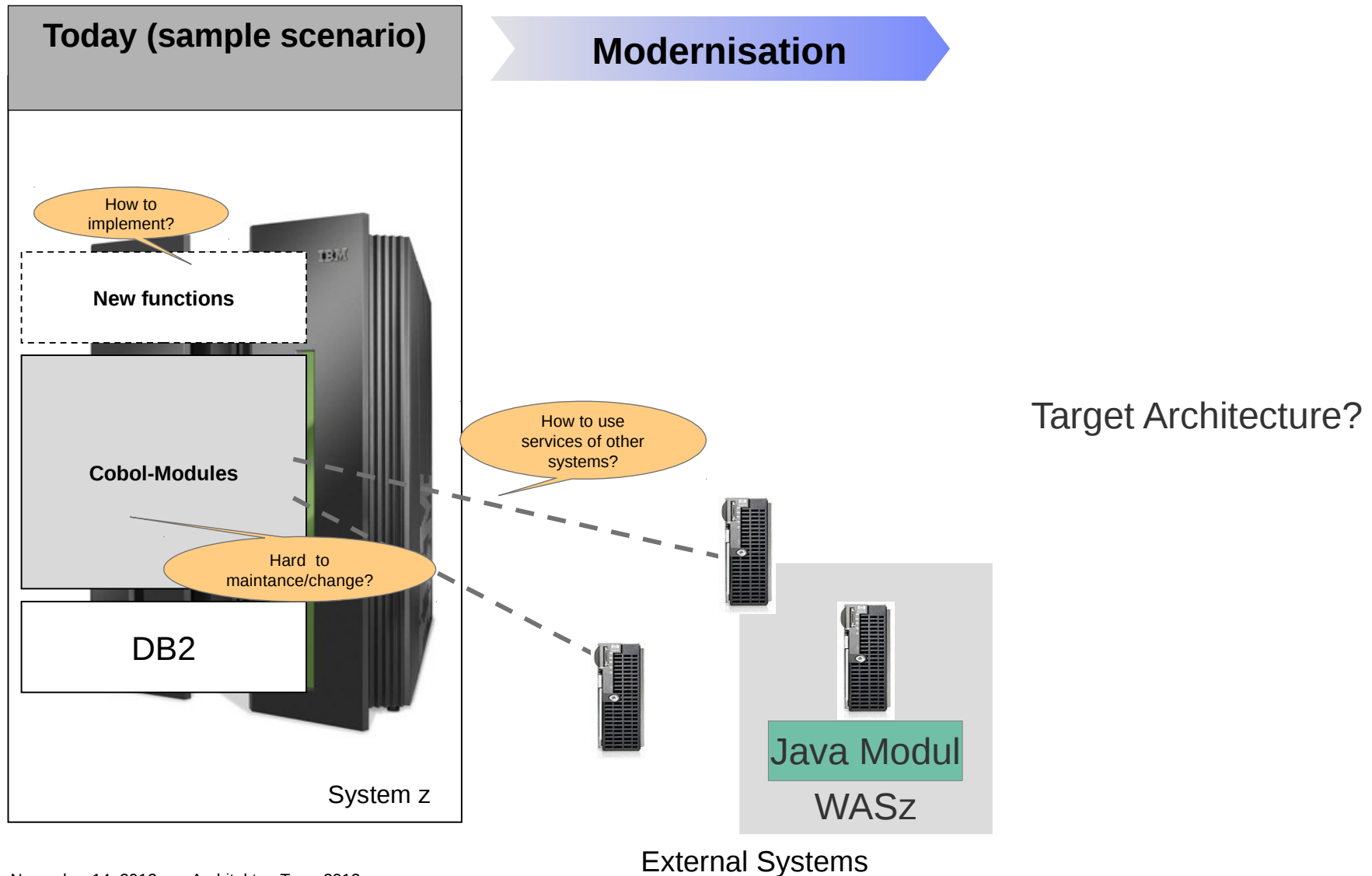


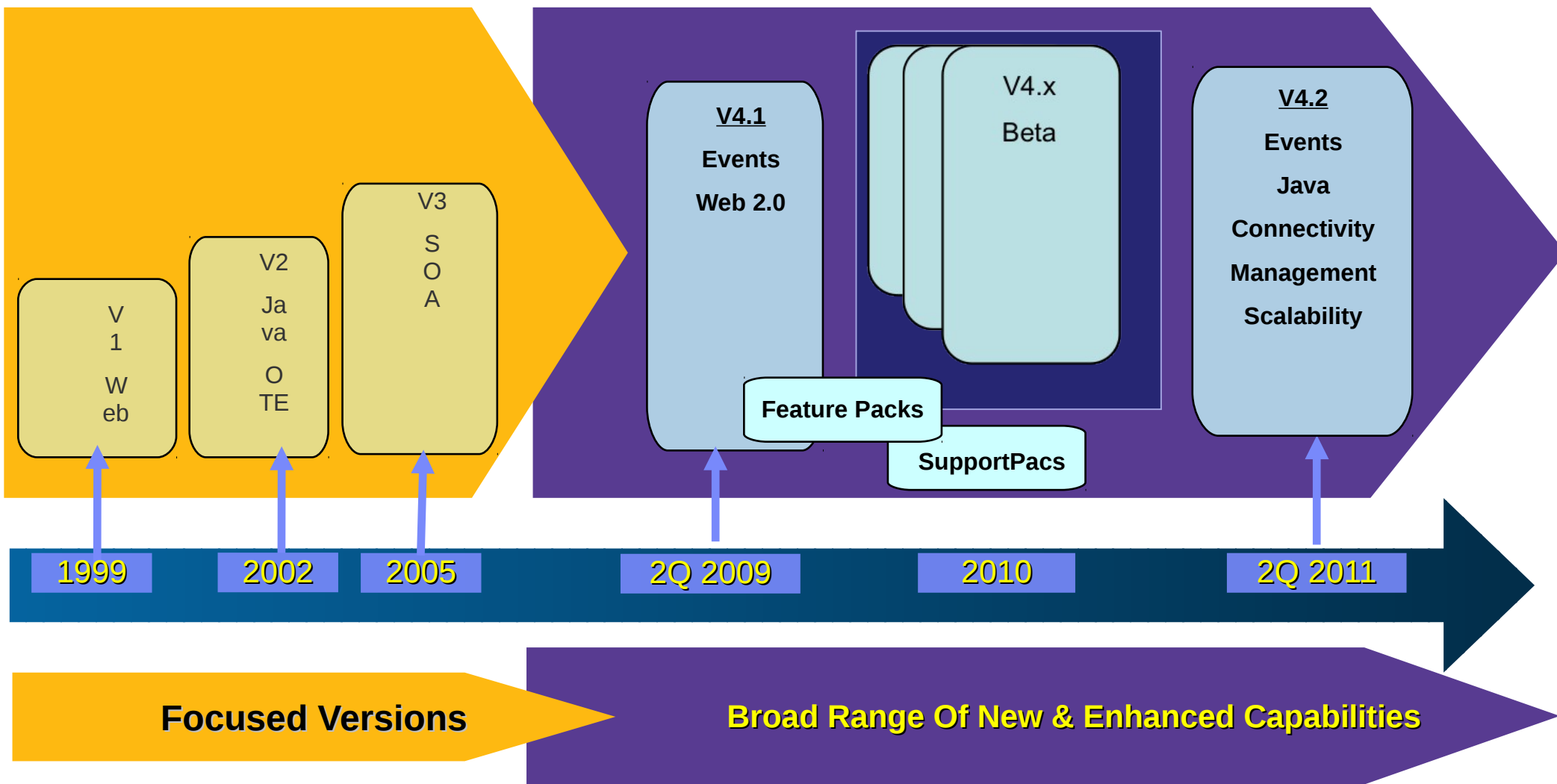
CICS 4.2 – Highlights and Technical News



Big Picture: Where to go with the CICS Development



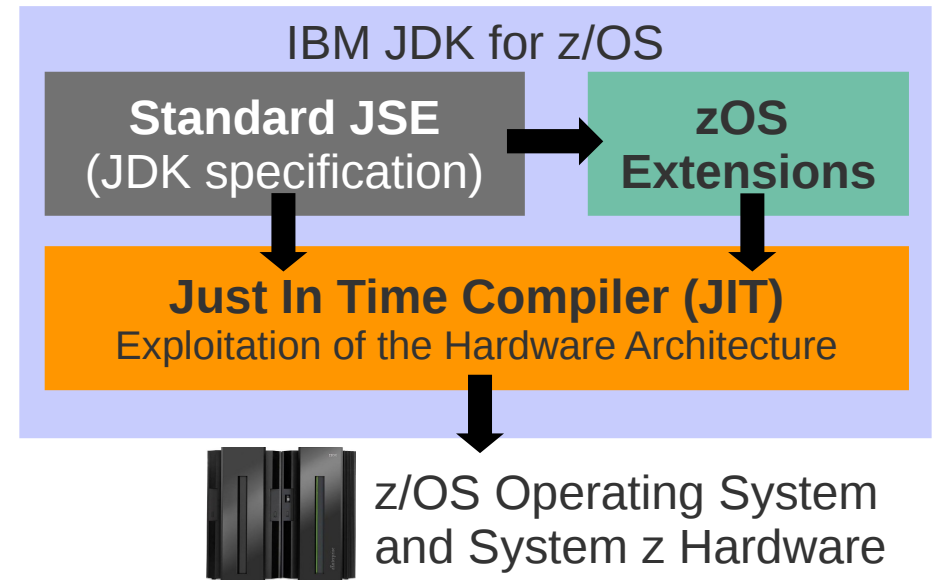
CICS Transaction Server : The Story So Far...



***Java* is just another Language which is running on the Mainframe**

Java on Mainframes - what is different and why here?

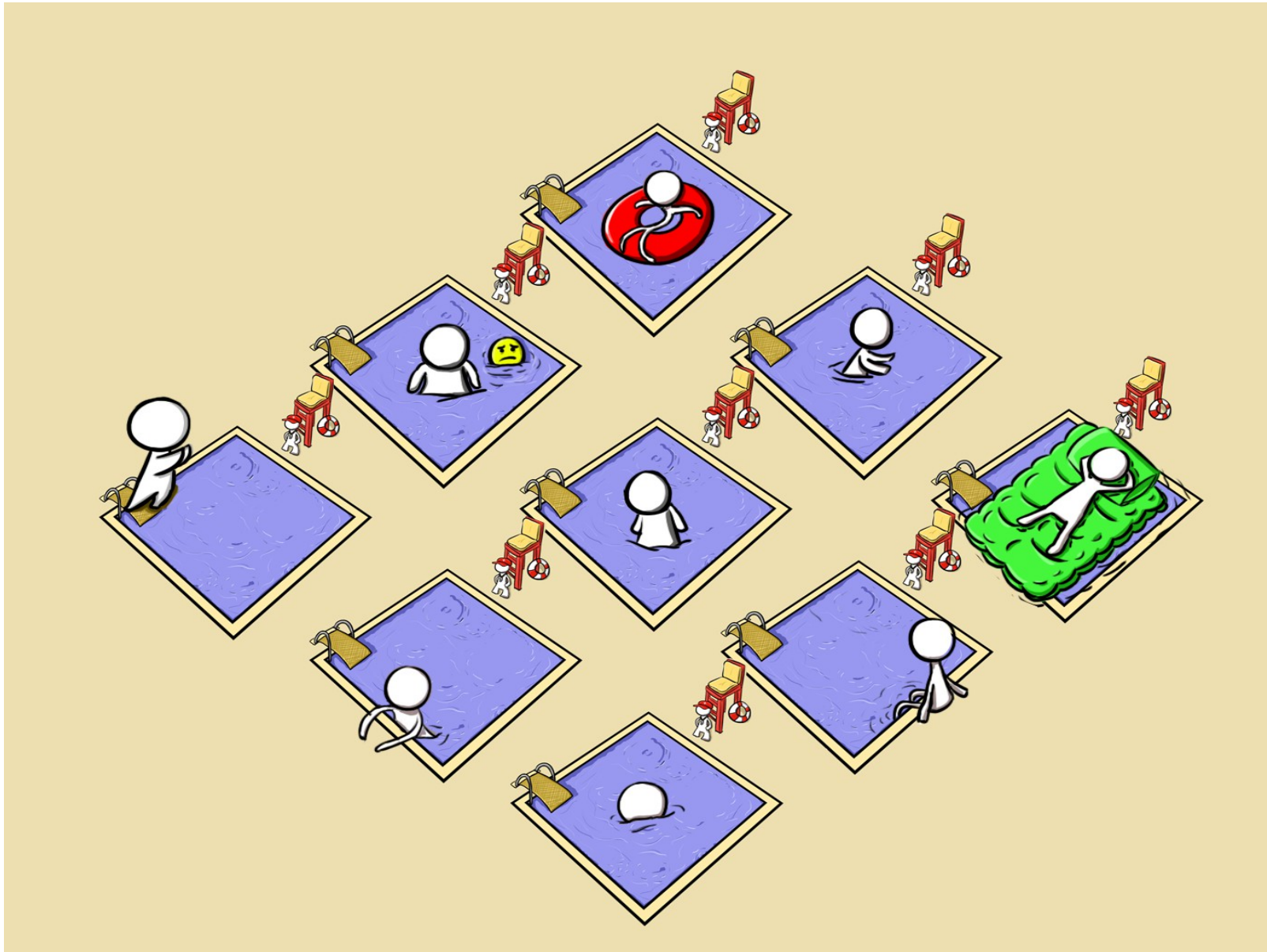
- IBM uses its own implementation of a JVM on mainframes that uses the underlying platform architecture
- The Java workload can be offloaded to a zAAP processor
- The Java logic can be a bridge between the mainframe and the distributed world
- Java is a common language on many platforms that can help to find a dialog between the departments
- Java is a language that is well known by many new professionals and so a good common ground, when they start to develop for mainframe applications
- The language encourages a good design and loose coupling of components



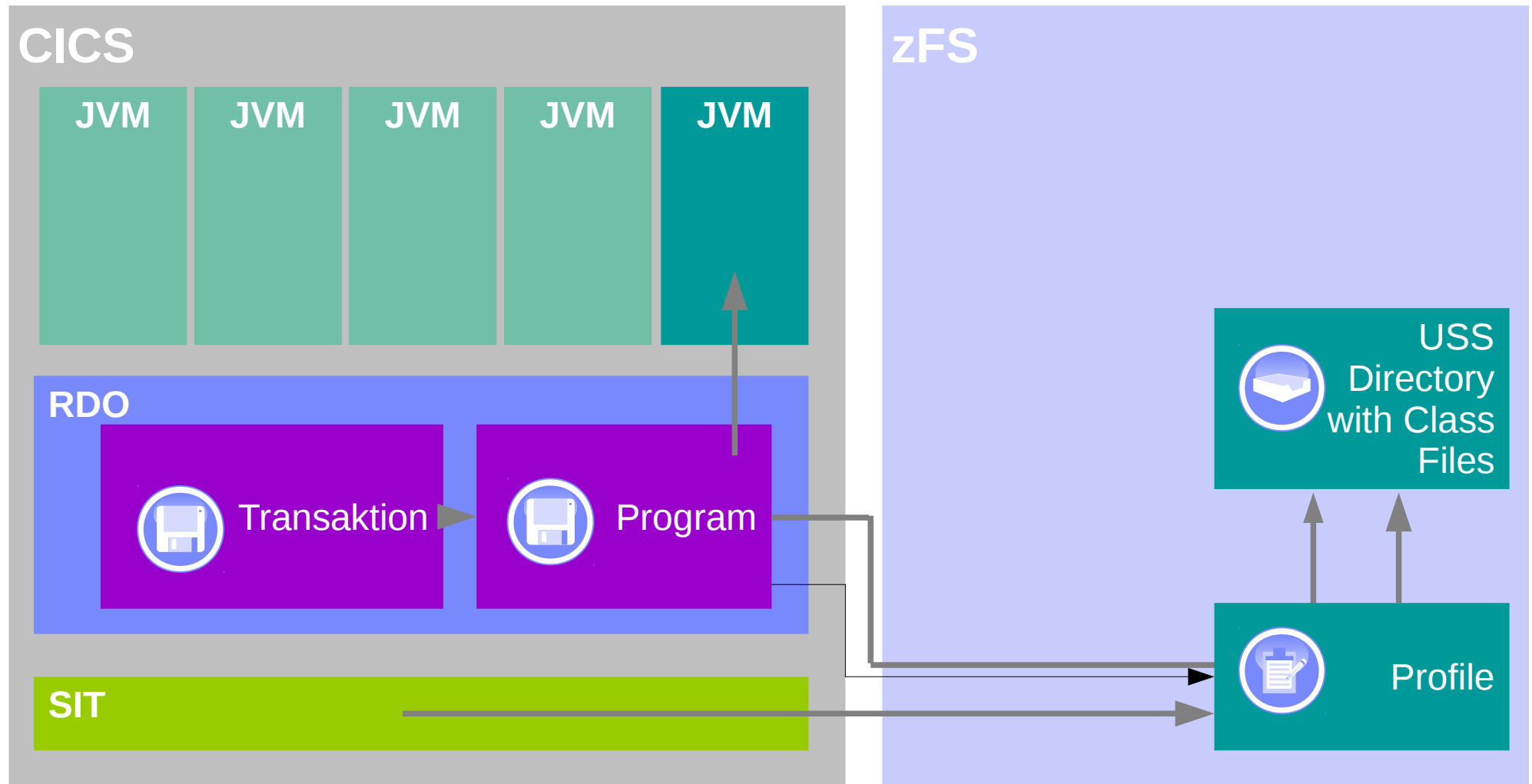
A Java Analogy



Flashback: CICS Pooled Server Architecture



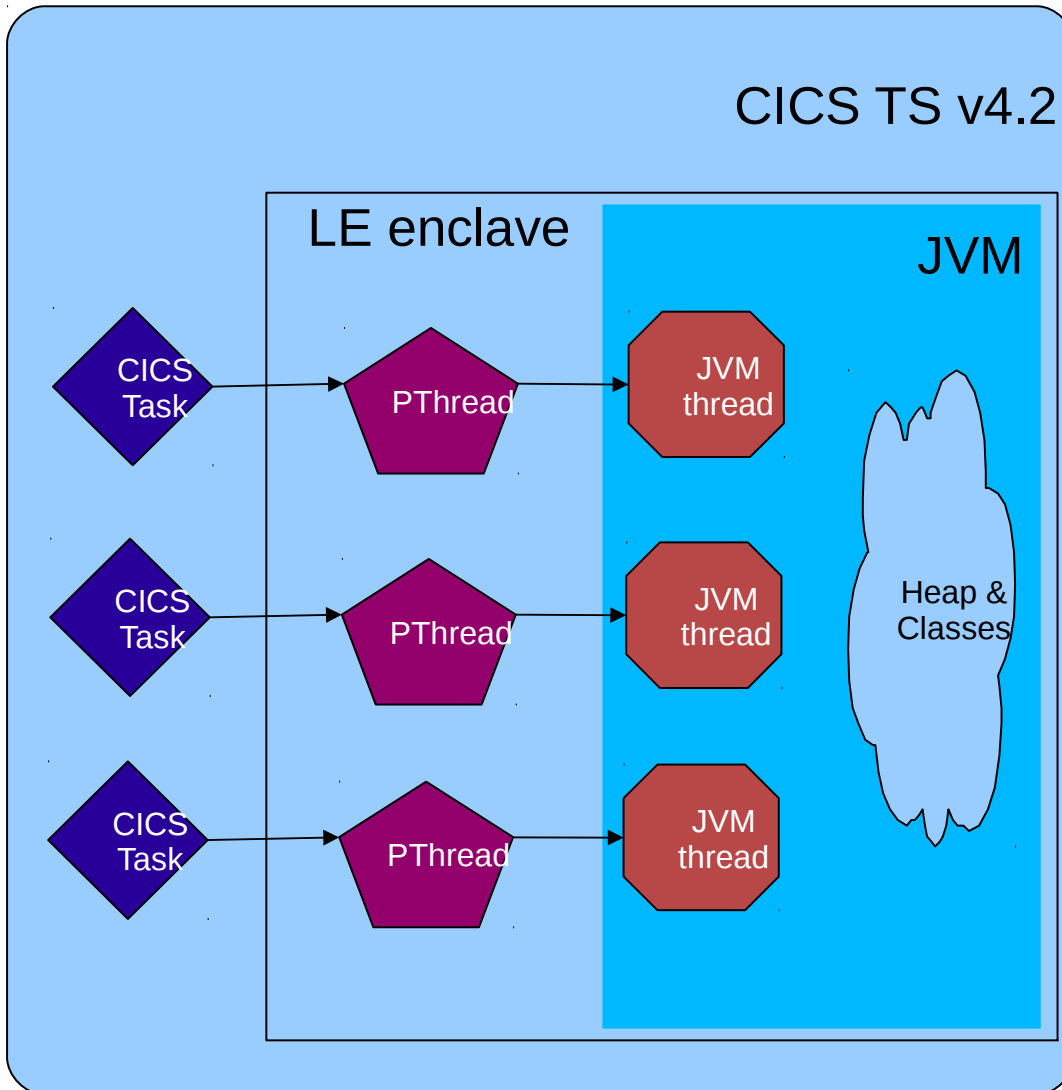
Flashback II: Old pooled Architecture



The JVM Server

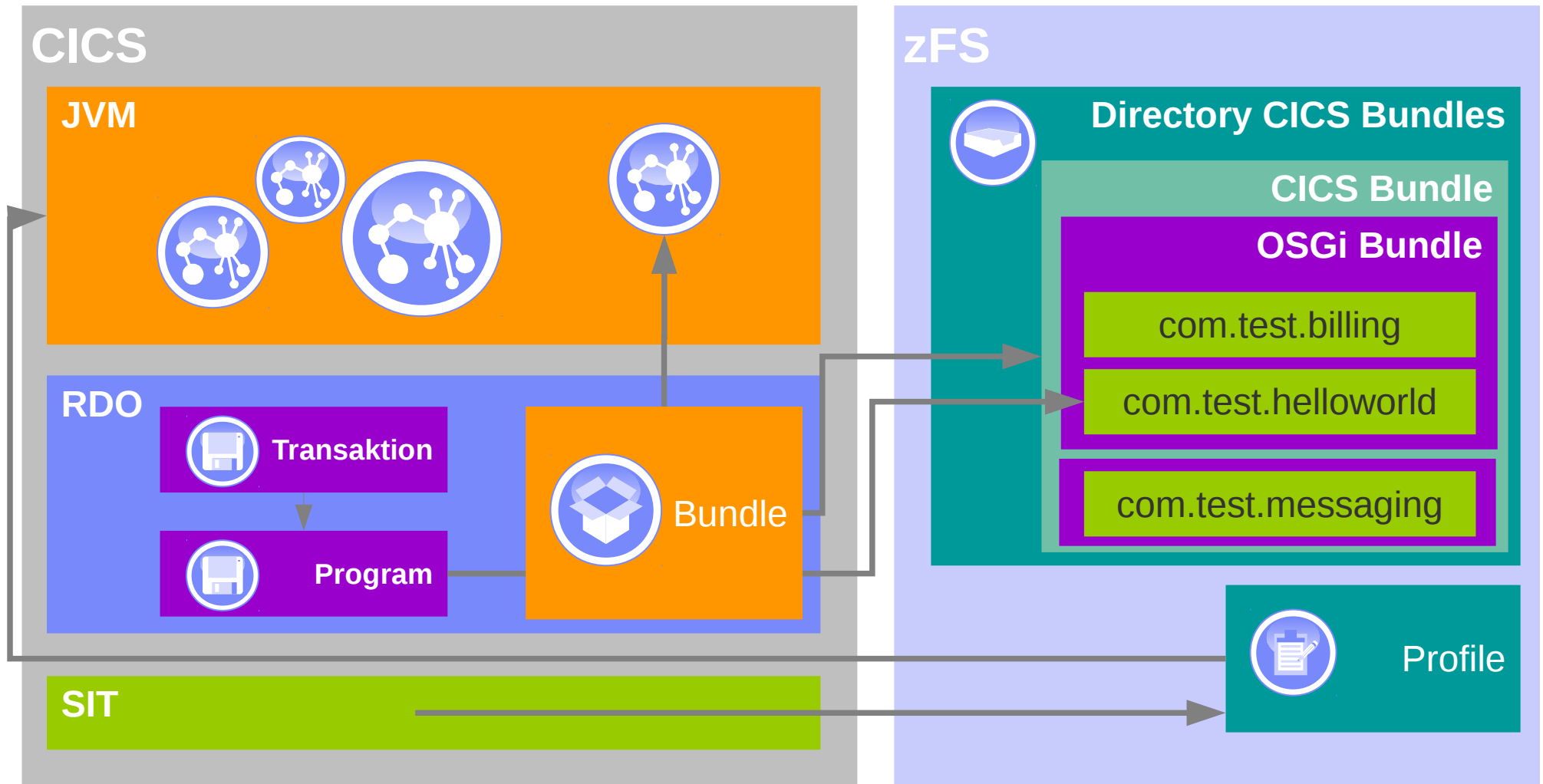


JVM servers



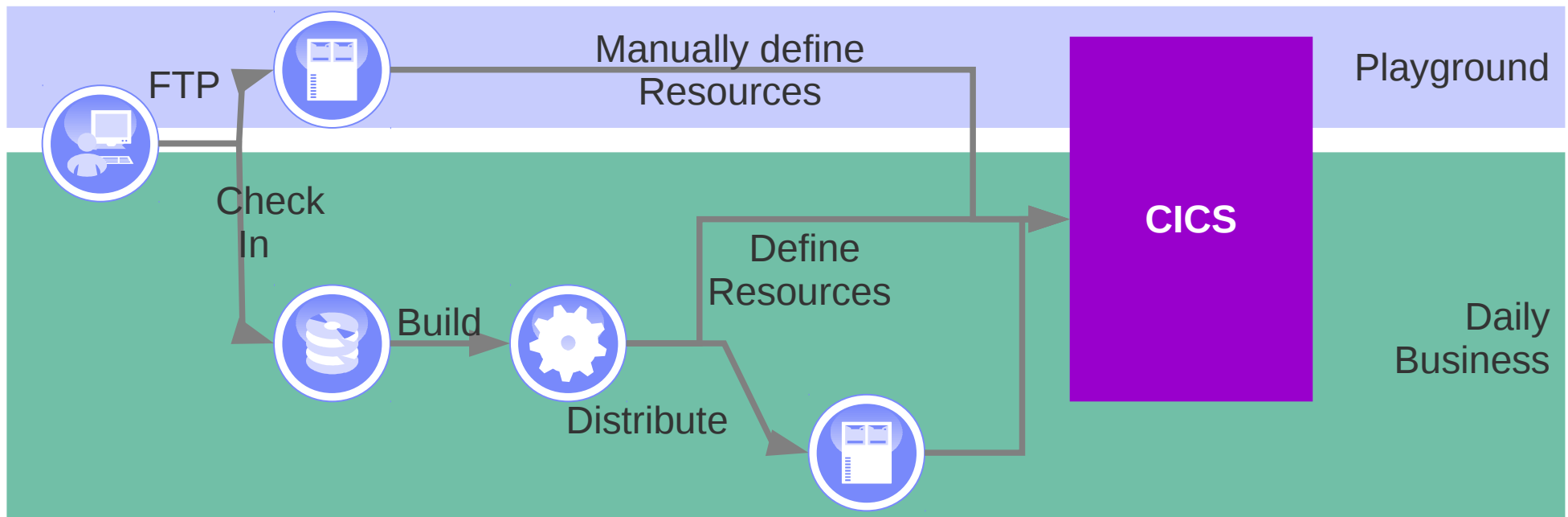
- CICS requests storage from MVS, sets up a Language Environment enclave, and launches the 64-bit JVM in the enclave.
- IBM® 64-bit SDK for z/OS, Java Technology Edition, Version 6.0.1
- Up to 256 parallel tasks/JVM & 1024/CICS
- Applications
 - Must be threadsafe
 - deployed as OSGi bundles (in CICS bundles)
- Dynamic updates without restart
- No EJB support

New JVM-Server Architecture



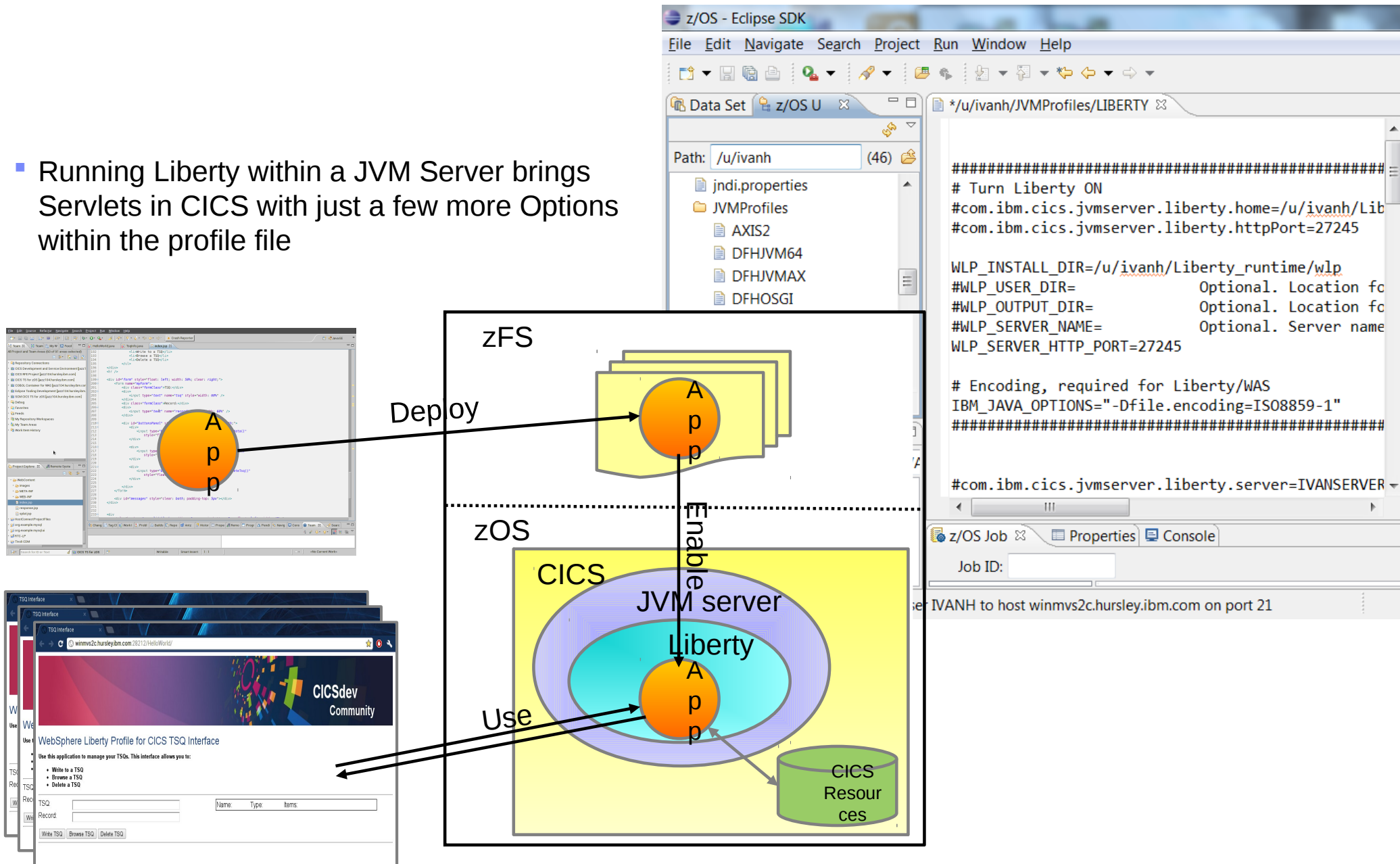
How are the Java Resources Managed within CICS and zOS in general

- With Java CICS leaves once again the traditional way of definitions within the CSD, like it does with the Event Bindings.
- The reasons are manifold, but the biggest impact comes from the increased complex artifacts
- So how are bundles managed?
 - Like the resources in the distributed world within zFS files:



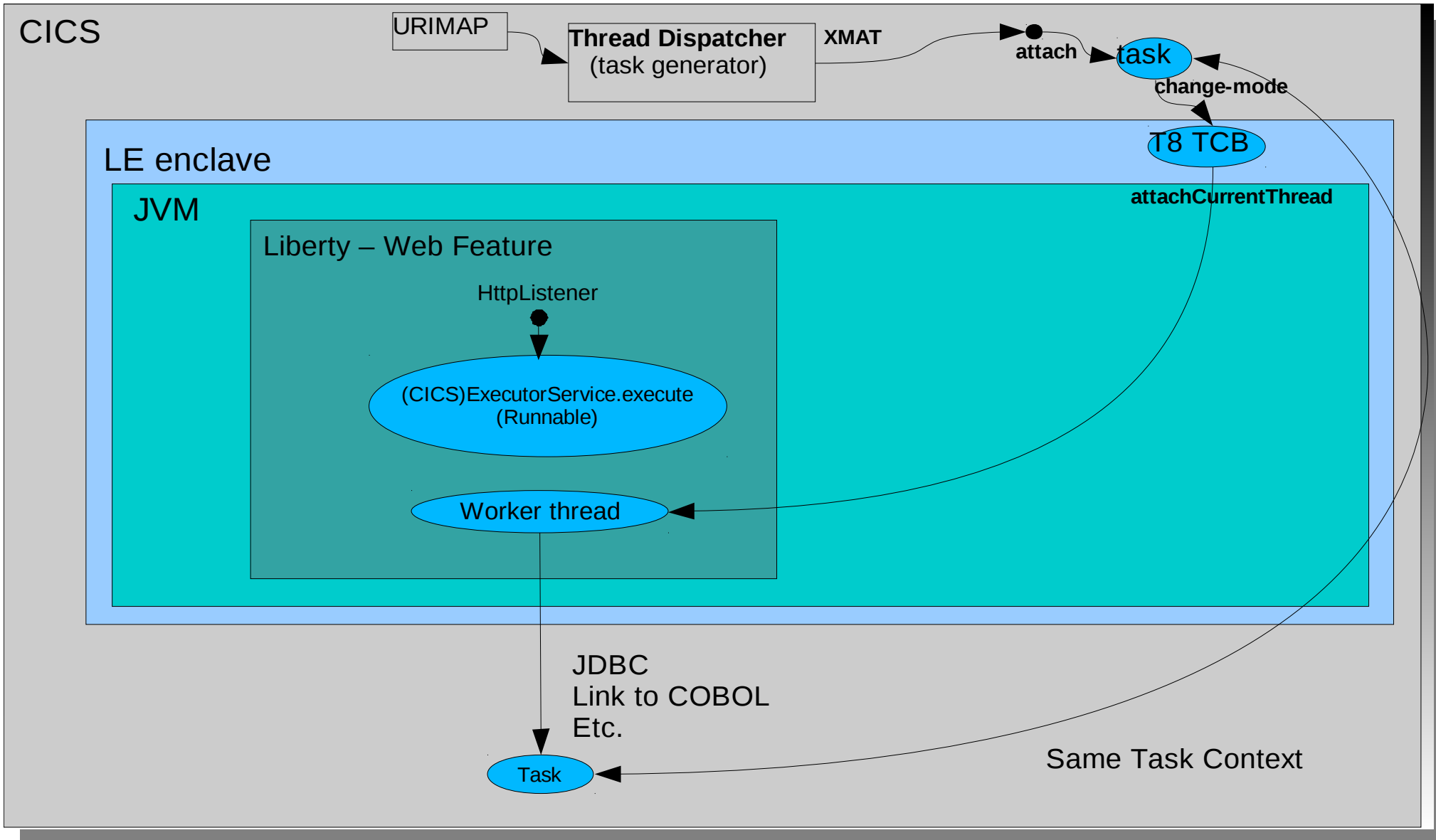
What's next? The Liberty Profile in the CICS 5.1

- Running Liberty within a JVM Server brings Servlets in CICS with just a few more Options within the profile file

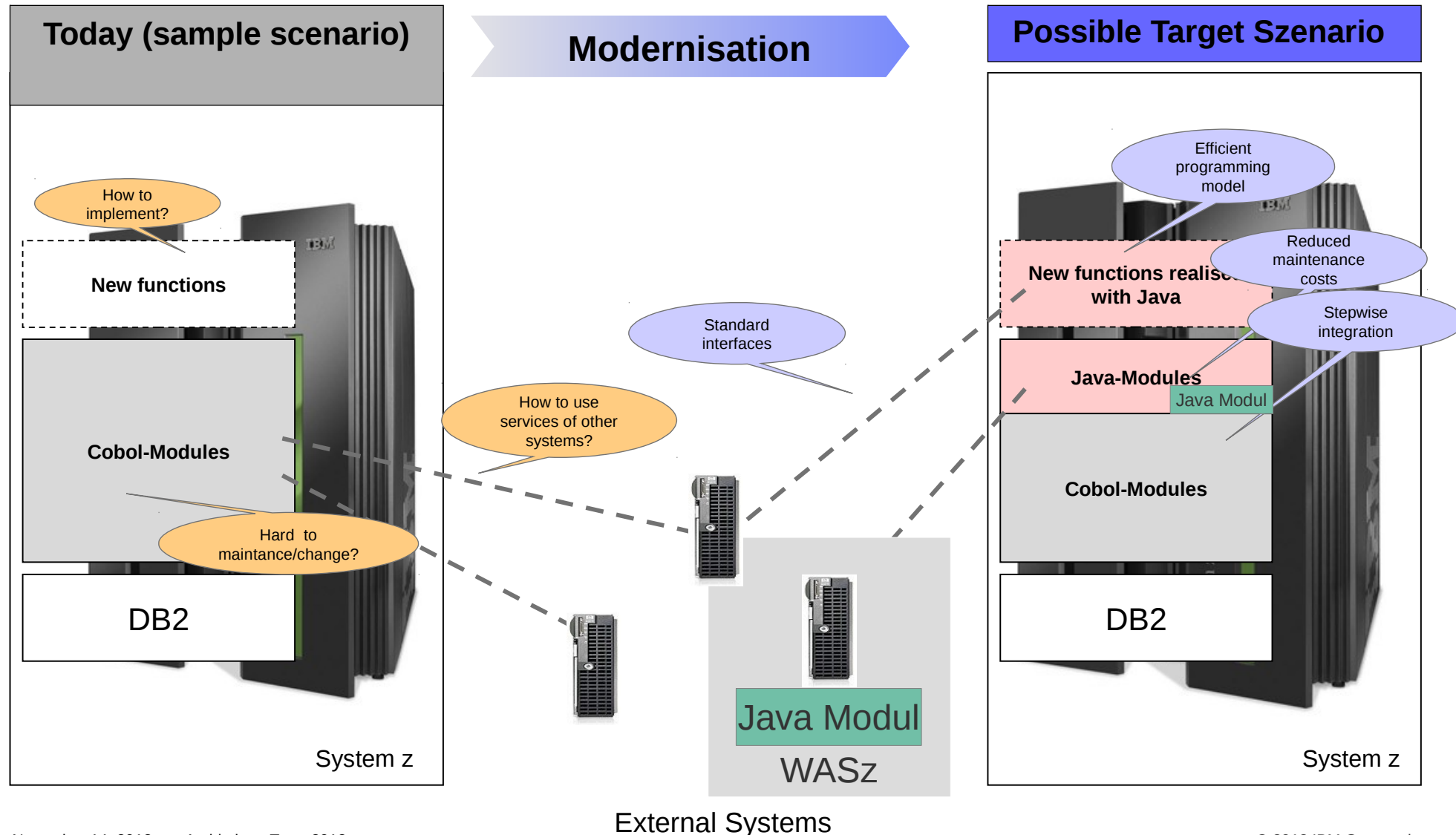


How does Liberty within CICS work

Liberty Code was NOT changed, just extended via extension Points



Big Picture: Where to go with the CICS Development



Porting/Deployment of Business-Rule-Service to Java in CICS

CICS Guide

Herbsttagung Darmstadt 2012

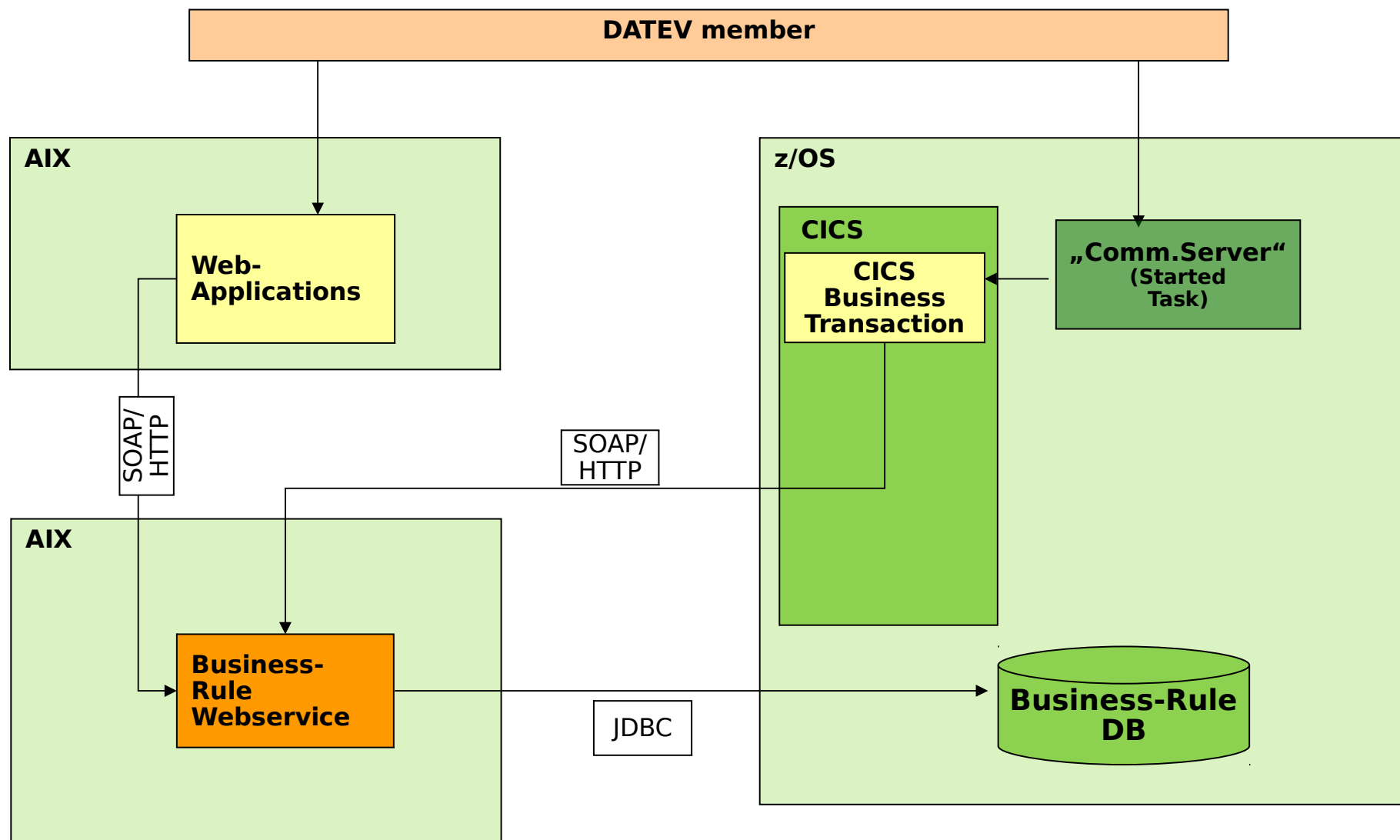
Jürgen Miehling

CICS-Systemprogrammierung

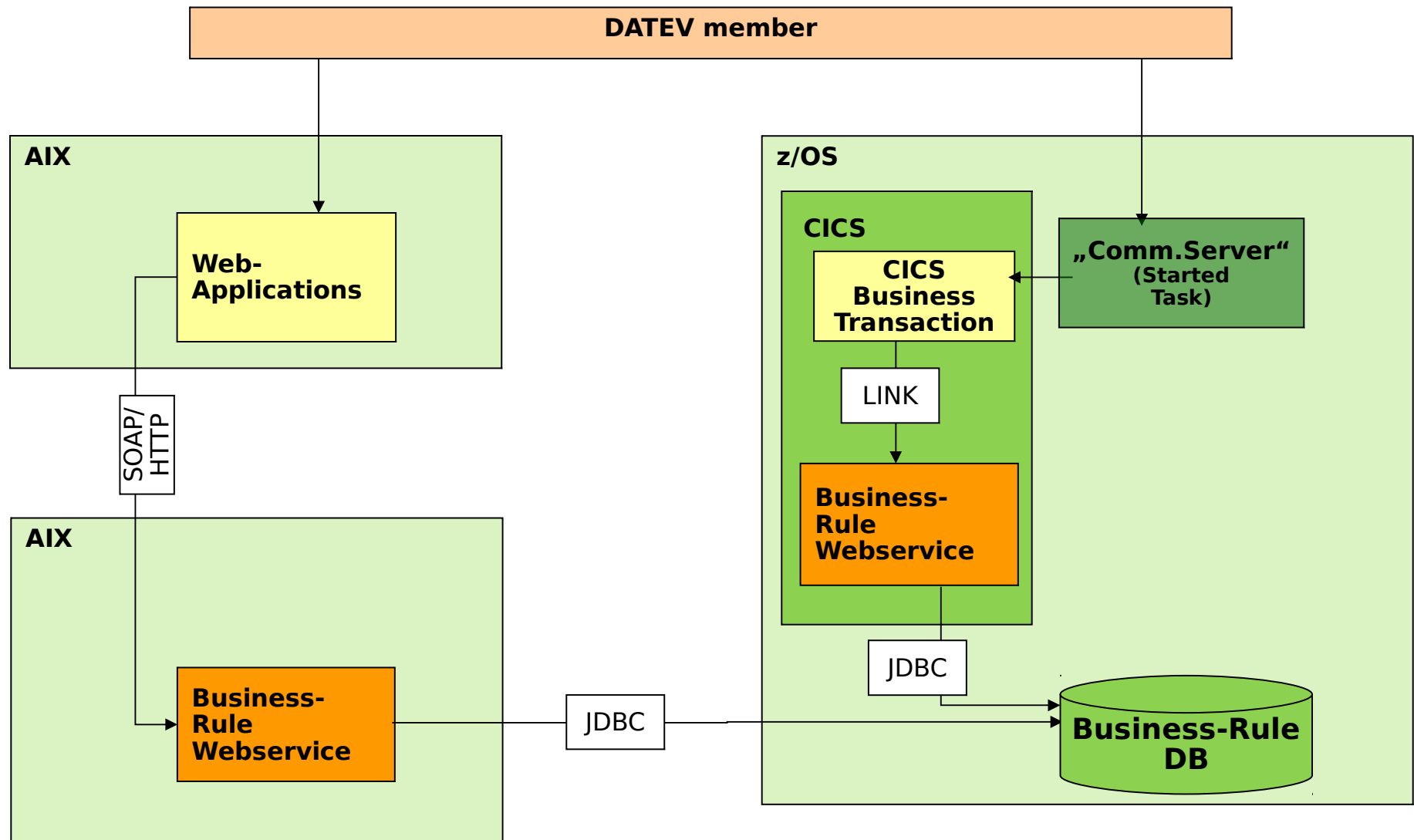
DATEV e. G.



Business-Rule Webservice: Old Architecture



Business-Rule Webservice: Current Architecture



Steps for Porting WebService to CICS



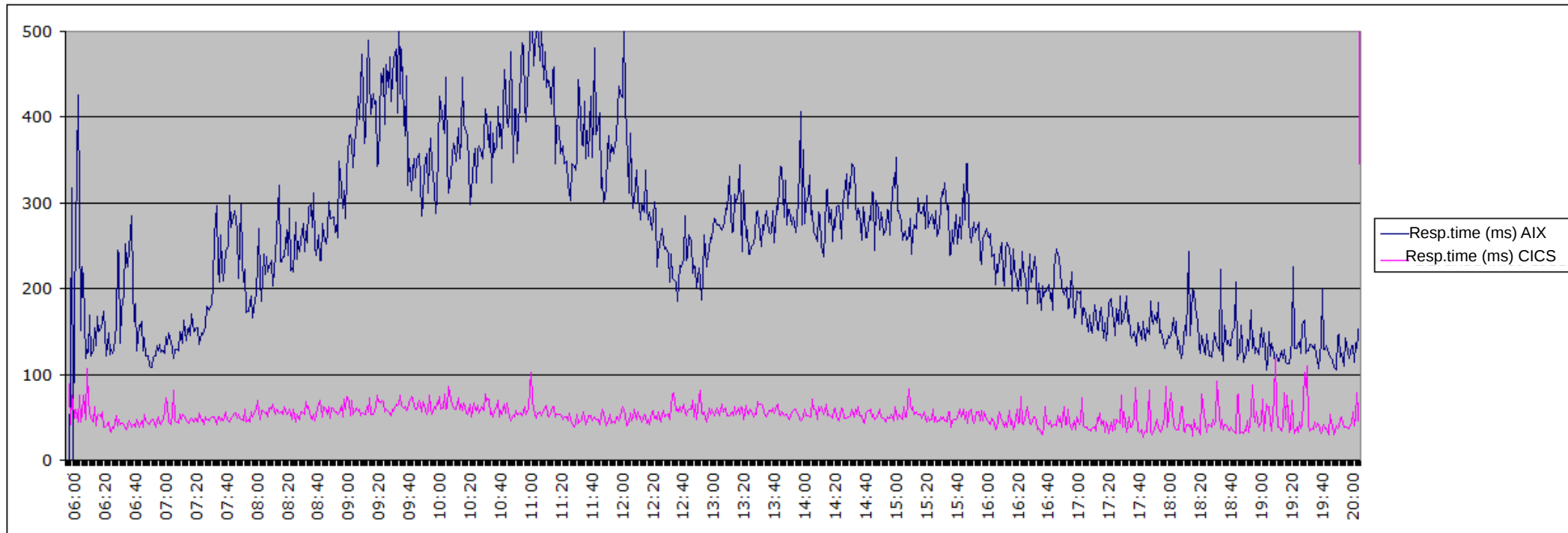
- Create C-Headerfiles via DFHWS2LS
- Use J2C-Records for Mapping C-Header to Java-Objects
- Create Java-Main-Class
 - Extract CICS-Container and map to Java-Objects
 - Call POJO
- Create DataSource using DB2SimpleDataSource, instead of JNDI-Lookup
 - Change from JDBC Typ-4 to JDBC Typ-2 Driver
- Porting Business-Logic to CICS **without any changes**
- Profiling with JInsight for Optimising

Results from Production-Environment

Scenario: CICS TX calls Service in CICS



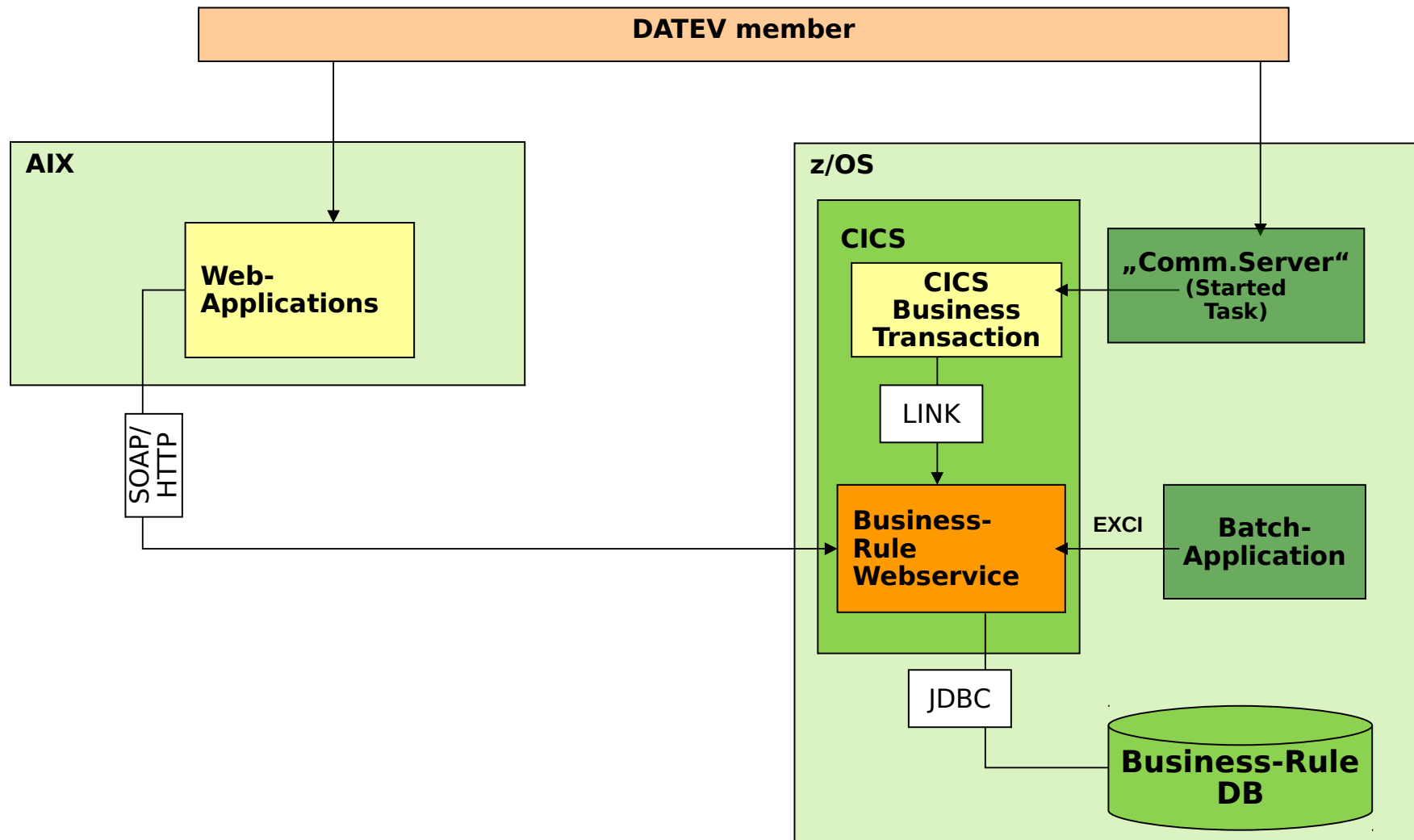
- Better Performance in production due to fast (local) access to DB2



- Easier to maintain (application-updates only in one place)
- Less involved systems => more stable
- One implementation for all platforms

Ann.: The above CICS performance-numbers were measured on a 2097-E64 – in the meantime, DATEV has migrated to a 2817-M49

Business-Rule Webservice: Target Architecture



Summary of porting WAS Java-Applications to CICS



- Most young developers at DATEV prefer Java instead of COBOL/C
- For fast response-times and high reliability it makes sense to have the business-application close to the data (DB2)
- DATEV has ported one application (special authorization-checks) from WAS to CICS. Result: Responsetime up to 5-times better with Java in CICS
- Other „Java legacy“ applications may follow
- Problem before CICS TS 4.2 with porting existing applications to CICS:
 - Clash on the Classpath of the JVM-Profile (each of the ported applications may e.g. use different version of log4j)
 - JVM-memory-requirements (each JVM-instance requires a „basic footprint“ of memory)
- Solution with CICS TS 4.2:
 - OSGi-support to get „application-specific classloaders“
 - CICS Java-Server allowed also for customer-applications

Kontakt

Tobias Leicher

Client Technical Professional
– CICS & CICS Tools

IBM Allee 1
D-71139 Ehningen
Mobil: 0151 – 15 16 24 89
Mail: tobias.leicher@de.ibm.com



VIELEN DANK FÜR IHRE AUFMERKSAMKEIT.