16562: Liberté, Égalité, Fraternité – a Mini CICS and WebSphere Revolution

Phil_Wakelin@uk.ibm.com

CICS Strategy & Design, IBM Hursley UK

#SHAREorg

SHARE is an independent volunteer-run information technology association that provides education, professional networking and industry influence.

SHARE in Seattle 2015

Abstract

“Liberté, Égalité, Fraternité” (Liberty, Equality, Fraternity), is the slogan of France, coined around the time of the French Revolution. It also seems a pretty appropriate slogan for the mini revolution that is happening right now with CICS and WebSphere. The Liberty profile is a highly composable and dynamic application server runtime environment that is shipped as a part of both WebSphere and CICS. This session will introduce Liberty in CICS, compare the capability with WebSphere (note the ‘equality’ word) and discuss how these new Liberty applications can interact with and support the established fraternity of existing CICS applications that run your core business.

Disclaimer

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

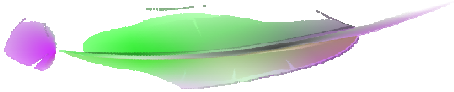
Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Agenda

- Liberty concepts
- Liberty features
- What can I do with Liberty in CICS
- CICS Liberty deployment models
- 4 simple steps to setup Liberty
- Administering Liberty runtime
- Demos

Liberté, Égalité, Fraternité" is the slogan of France, coined around the time of the French Revolution. It also seems a pretty appropriate slogan for the mini revolution that is happening right now with CICS and WebSphere. The Liberty profile is a highly composable and dynamic application server runtime environment that is shipped as a part of both WebSphere and CICS. This session will introduce Liberty in CICS, compare the capability with WebSphere (note the 'equality' word) and discuss how these new Liberty applications can interact with and support the established fraternity of existing CICS applications that run your core business.

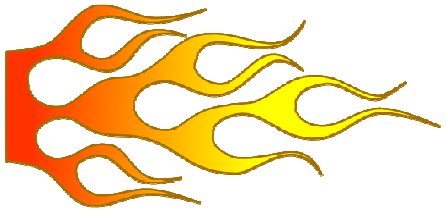
Liberty is…



A LIGHTWEIGHT

COMPOSABLE

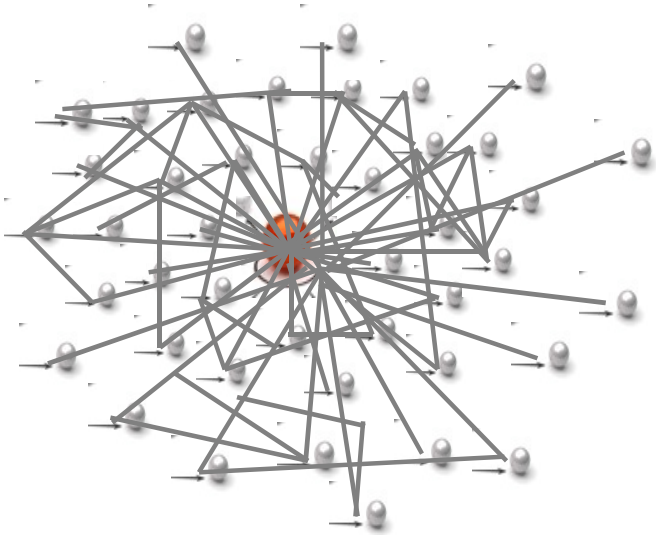
FAST



…'Profile' of WebSphere Application Server

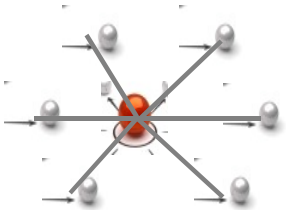
Composable – you pick the Features

If this is Traditional WAS (tWAS)...

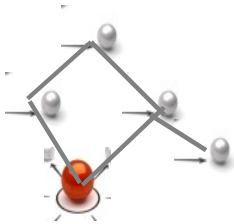


WebSphere

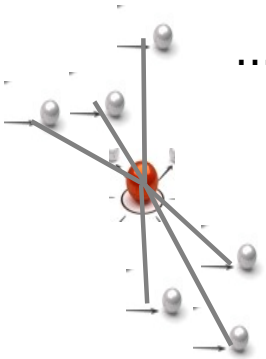
...this is Liberty (WAS)



...so is this



...or even this!



Lightweight Configuration

```
<server description="tradeLiteServer">
    <featureManager>
        <feature>jsp-2.2</feature>
        <feature>jdbc-4.0</feature>
    </featureManager>

    <logging consoleLogLevel="INFO" />

    <application type="war"
                 id="tradelite"
                 name="tradelite"
                 location="${shared.app.dir}/webcontainer/tradelite.war" />

    <include location="jdbc-drivers.xml" />
    <include location="${user.home}/custom.xml" optional="

    <dataSource id="jdbc/DerbyTradeDataSource"
                jndiName="jdbc/TradeDataSource"
                jdbcDriverRef="DerbyEmbedded">
        <properties databaseName="${shared.resource.dir}/data/tradedb" />
    </dataSource>
</server>
```

Features control what's available in the runtime.

Instance configurations allow multiple instances of resources and applications to be declared

Includes can be used to implement an extensible configuration model

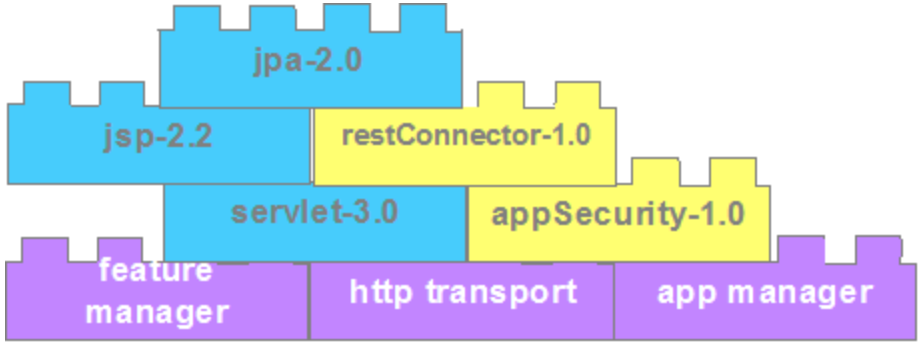
References can be used in multiple elements to point to and share a common definition

Composability – Based on features

```
<server description="composabilityIsKey">

    <featureManager>
        <feature>appSecurity-1.0</feature>
        <feature>jsp-2.2</feature>
        <feature>restConnector-1.0</feature>
        <feature>jpa-2.0</feature>
    </featureManager>

</server>
```



What's in the CICS TS Liberty Profile?

- Subset of the the WAS for z/OS Liberty profile
- Additional CICS specific integration features
- Only configure the functions you use

CICS TS V5.1

- Web/servlet/RESTful technology

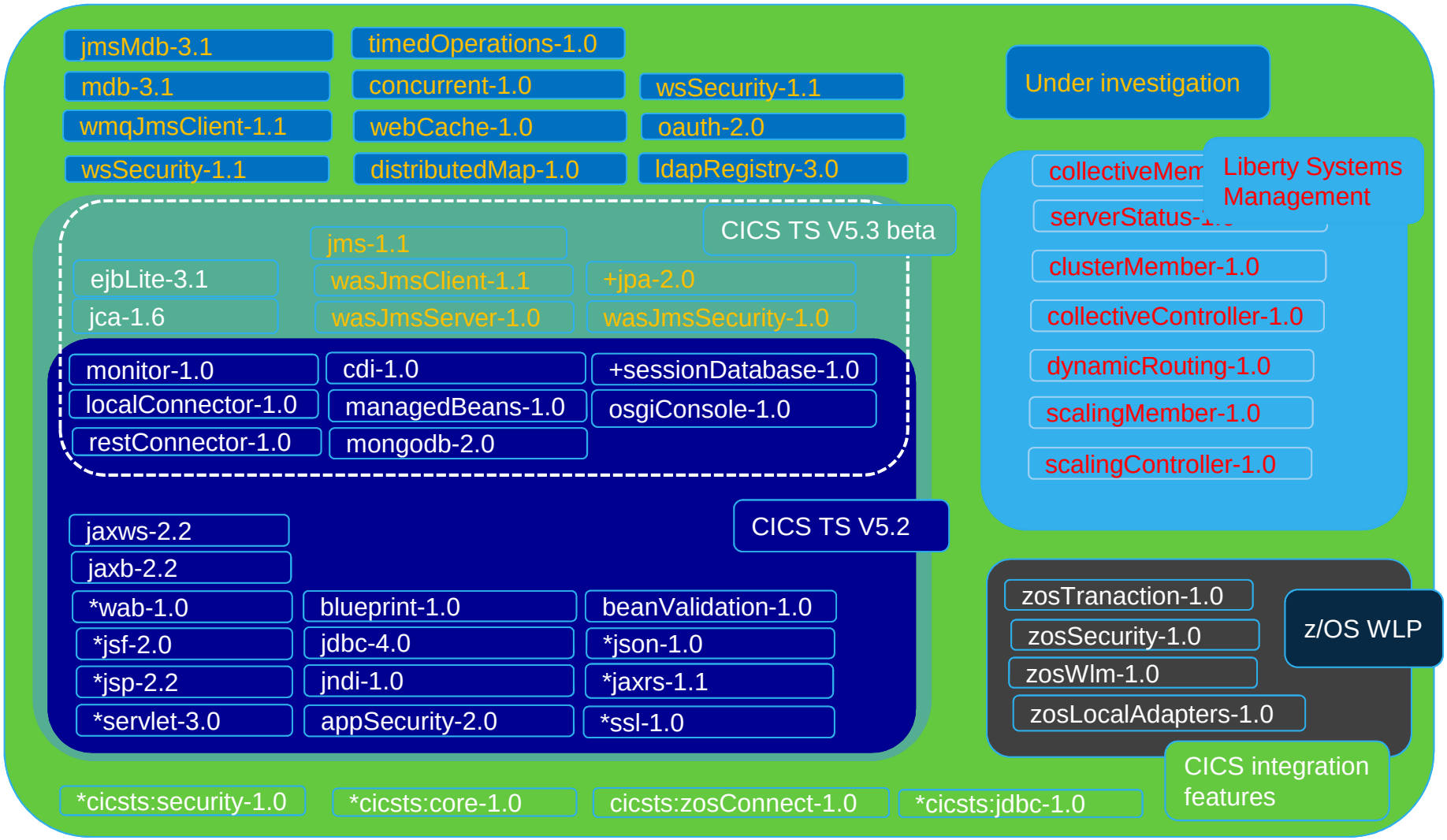
CICS TS V5.2

- JEE integration with CICS security
- Java transactions and CICS UOW integration
- Local and remote JDBC
- Java web services (JAXWS)
- z/OS Connect integration (APAR PI25503)

CICS TS V5.3 open beta

- JEE6 Web profile
- MongoDB
- JCA resource adapters
- HTTP session persistence (also in V5.2)
- EJBLite, CDI & Managed beans (also in V5.2)
- JMX monitoring (also in V5.2)

CICS TS Liberty features



* Originally available on CICS TS V5.1

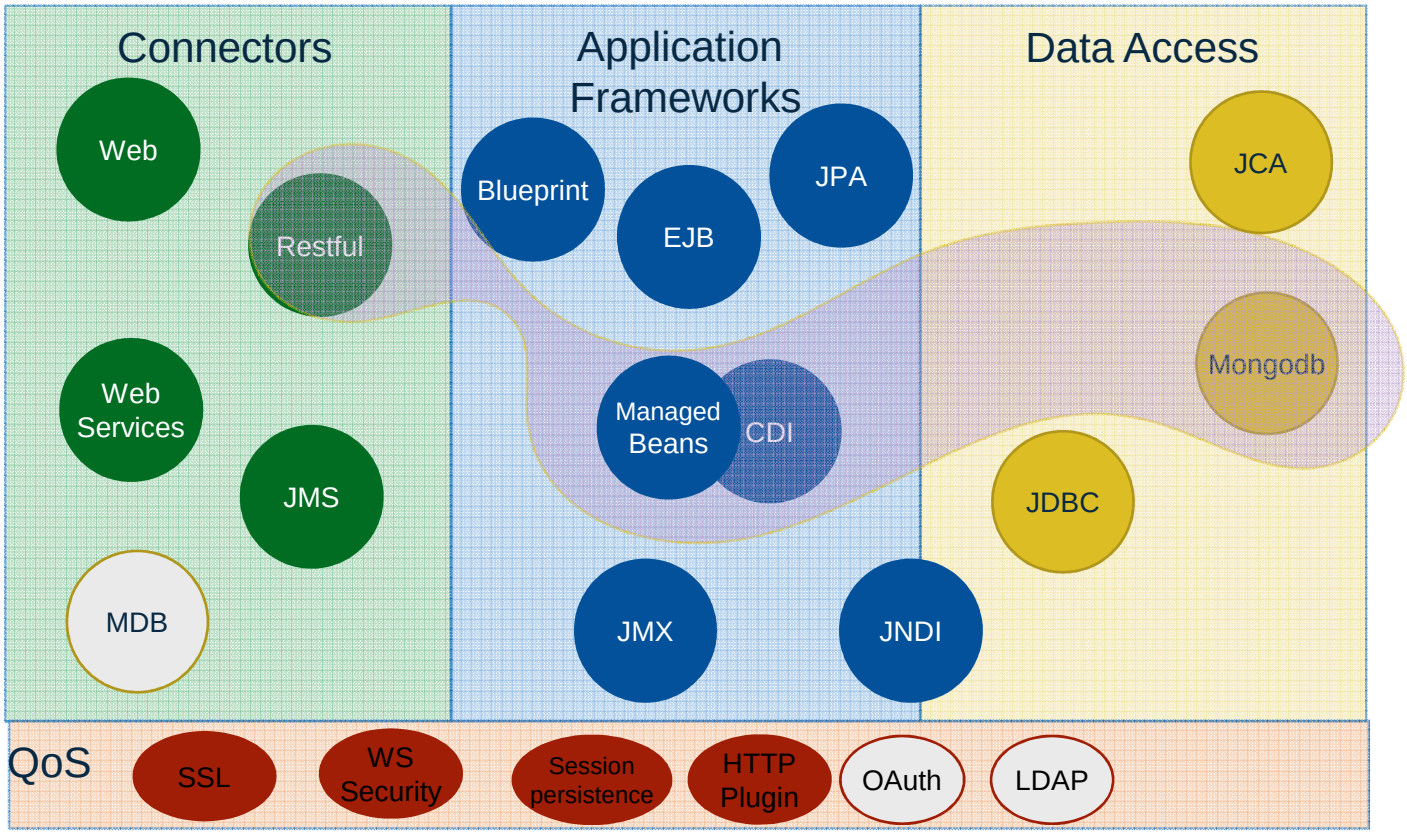
\+ Limited to T4 database drivers

CICS TS V5.3 Beta - New Liberty Features

Features	Description
jpa-2.0	Java persistence architecture (limited to T4 d/b drivers) Completes JEE6 Web profile subset
ejblite-3.1	Subset of EJB3 specification with local EJBs only CMT and BMT managed Java Transactions. No support for *NotSupported* Transaction attribute. Includes support for EAR bundle parts in CICS bundles
monitor-1.0 * restconnector-1.0 * localconnector-1.0 *	JMX monitoring of runtime and Web applications using local or remote API or JConsole
session-database-1.0 *	High availability and persistence of HTTP sessions, with a T4 database driver.
mongodb-2.0 *	Remote access to NoSQL open source database
cdi-1.0 & managedBeans-1.0*	Lightweight framework for bean management
osgiConsole-1,0 *	Debugging of OSGi framework & applications
Jms-1.1, wasJmsClient-1.1, wasJmsServer-1.1	JMS messaging to embedded WLP JMS server

*Also available on CICS TS V5.2

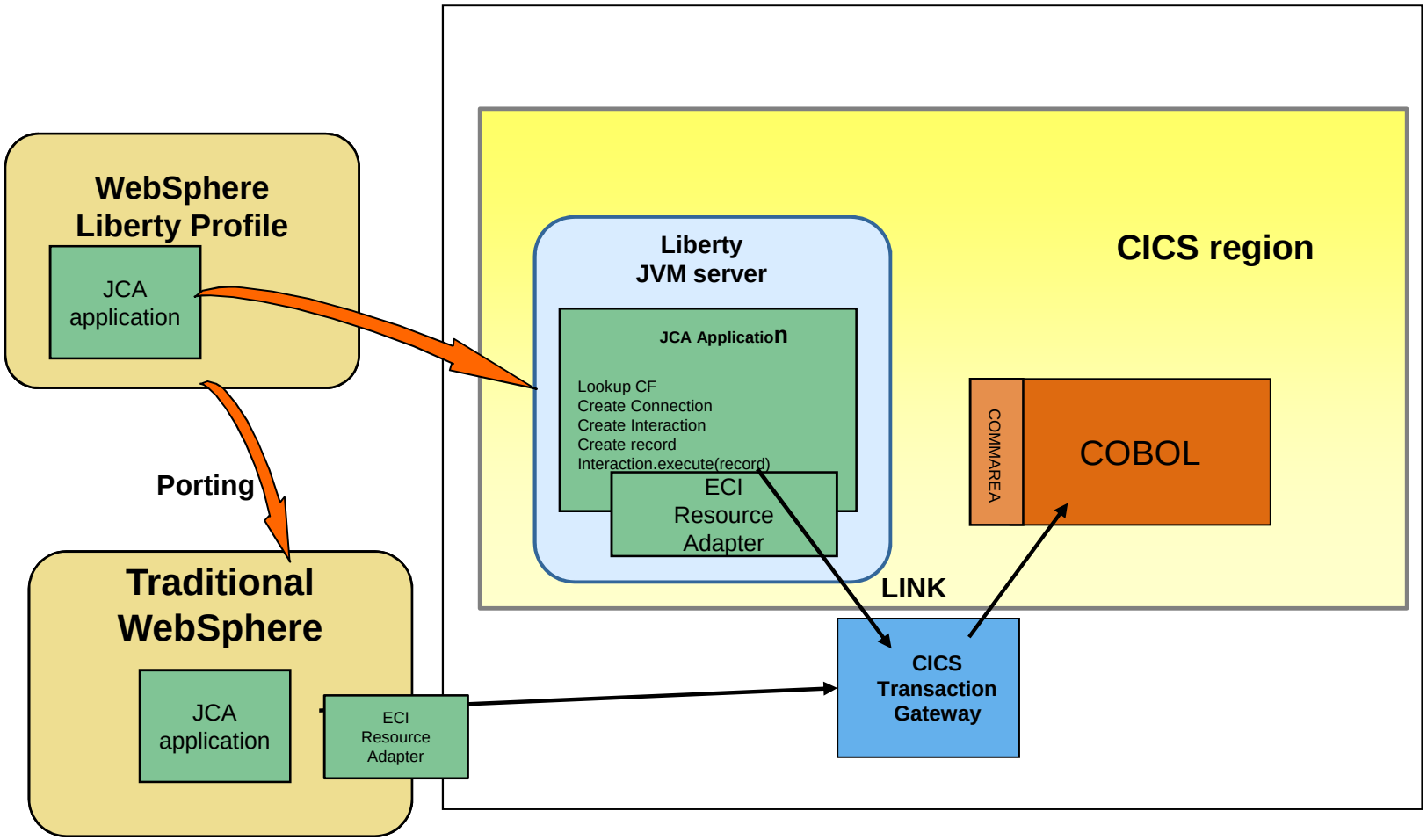
Building a JEE application



CICS TS V5.3 Beta – Enhanced Interoperability

- JCA resource adapter support
  - Portability of JCA based applications
  - Portable alternative to JCICS Program.link() when using CICS TG JCA
  - Develop/port your own resource adapter
- LINK to Liberty prototype
  - EXEC CICS LINK to main class in an OSGi bundle in Liberty JVM from non-Java programs

JCA in Liberty

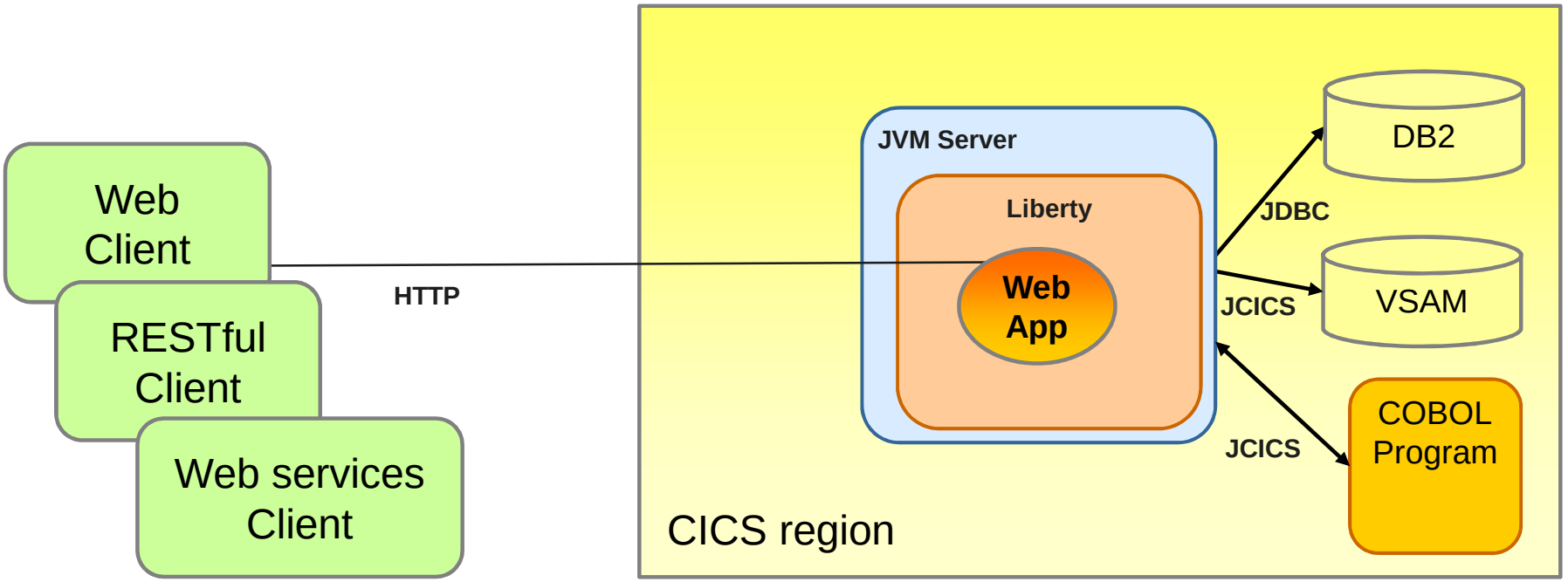


Simplified JVM server administration

- Management of number of zFS log/trace files
  `LOG_FILES_MAX=<n>`
- Redirection of zFS logs to JES (also in V5.2)
  `STDERR=//DD:STDERR`
- Localized time stamps in JVM output & file names
  - Controlled via USS TZ variable
    `TZ=GMT0BST,M3.5.0,M10.4.0`
- Common directory structure for zFS log/trace files
  - stdout/stderr/trace now located in &WORK_DIR/&APPLID/&JMVSERVER

- Default Web application
  `<cicsts_defaultApp deployed="true"/>`
  http://<server>/com.ibm.cics.wlp.defaultapp

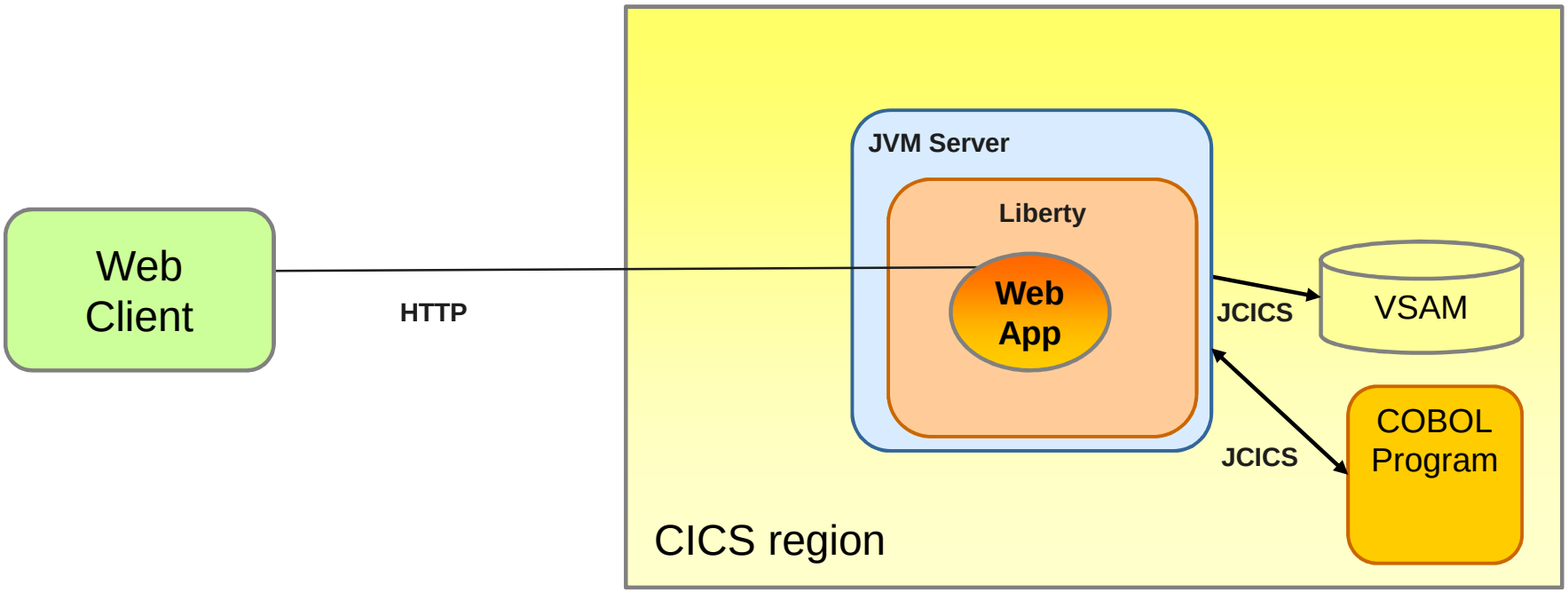
Liberty in CICS scenarios



- Typical scenarios for Web app. deployment to CICS
  1. Access to existing CICS programs or data VSAM data
  2. Sharing access to DB2 tables controlled by CICS
  3. Reducing network I/O by removing remote connector

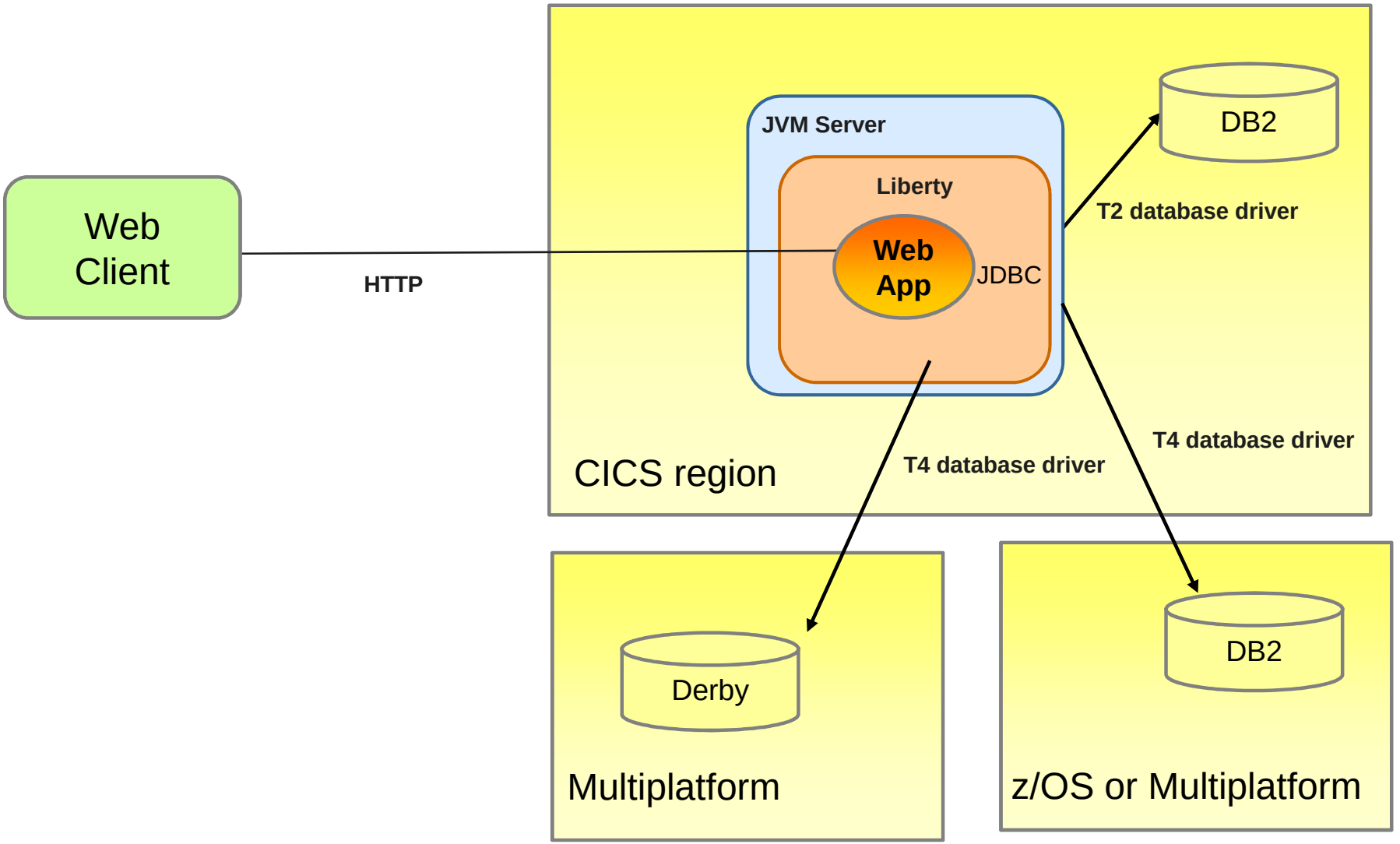
Liberty in CICS scenarios - 1

Access to existing CICS programs/data



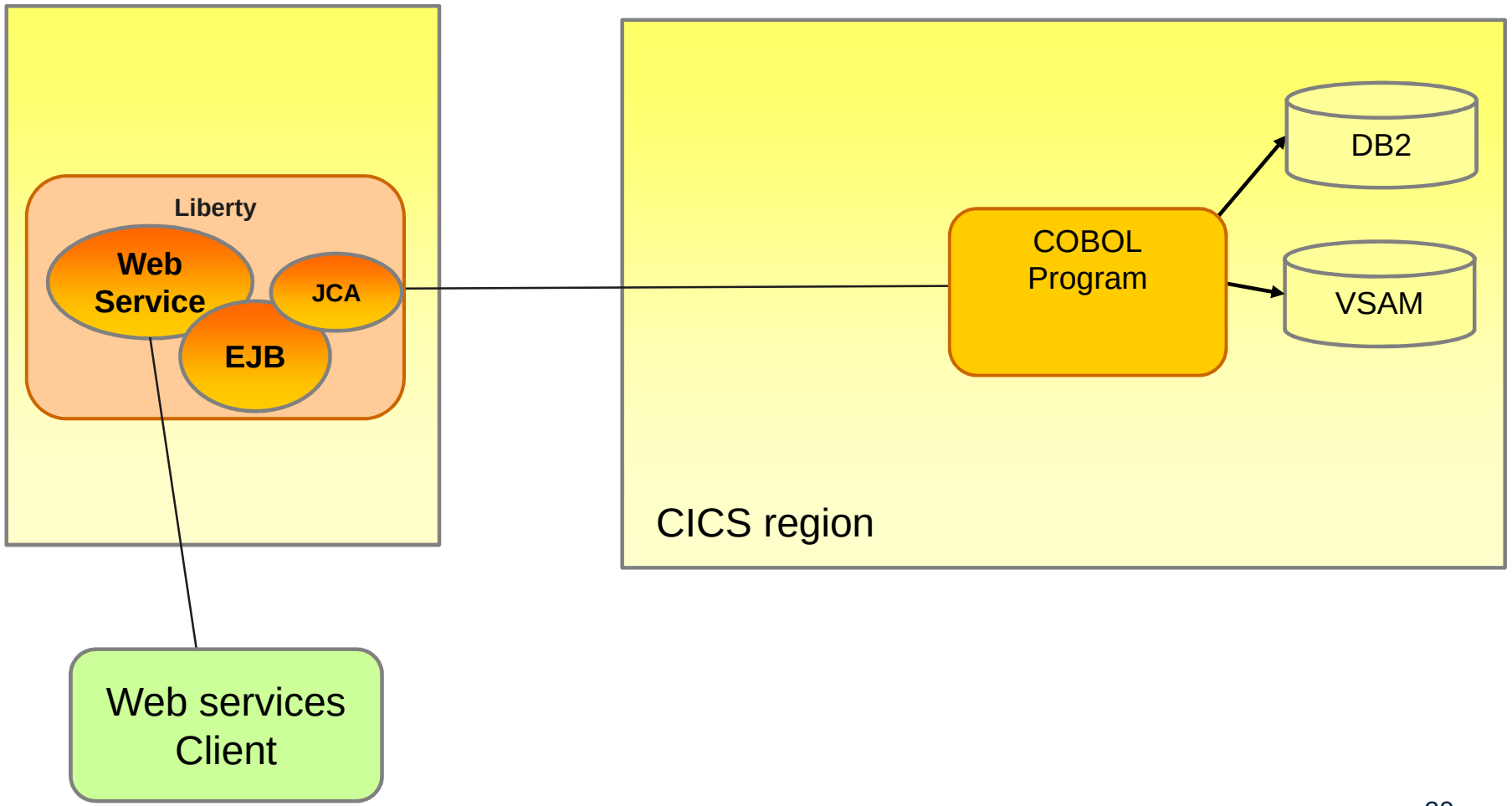
Liberty in CICS scenarios - 2

Accessing relational data



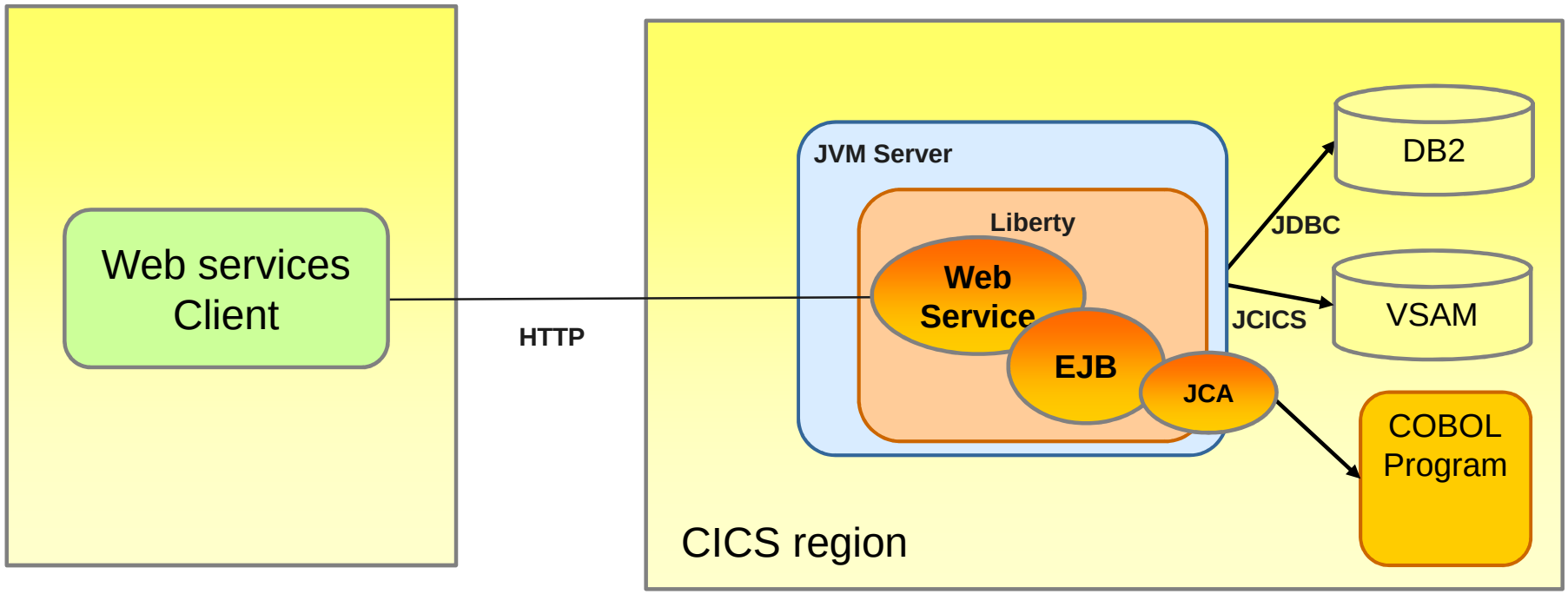
Liberty in CICS scenarios - 3

Collapsing connector path



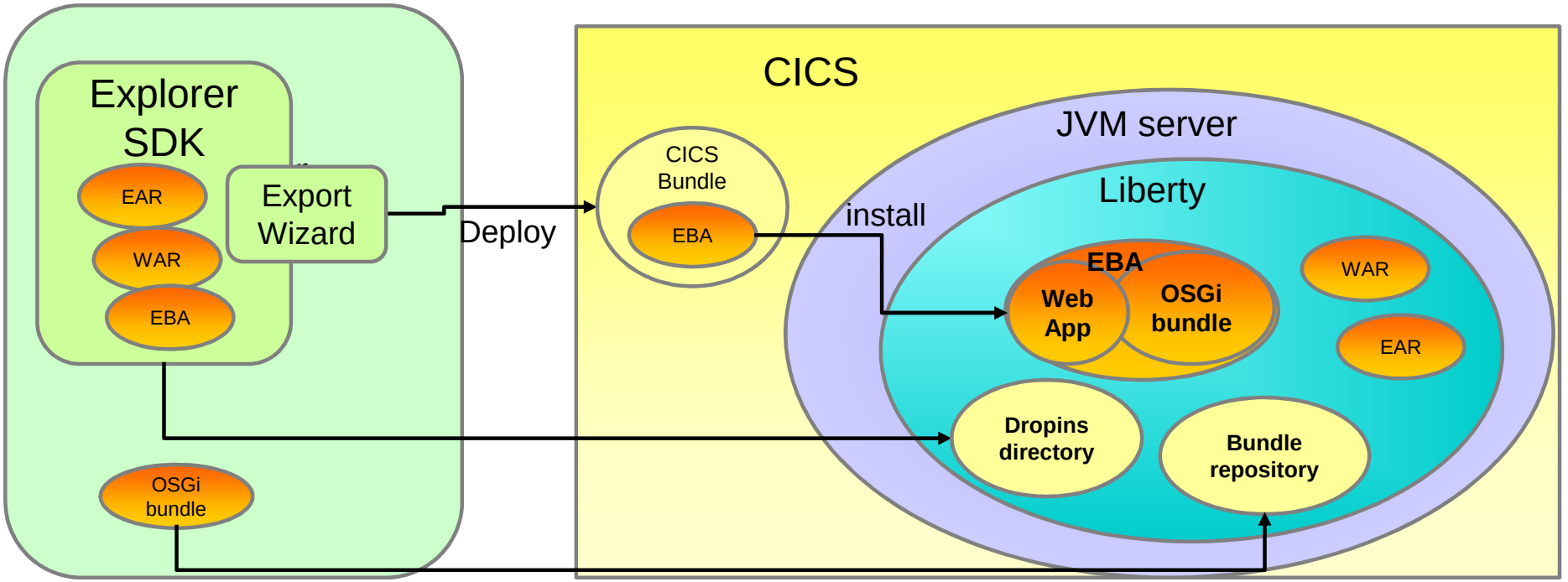
Liberty in CICS scenarios - 3

Collapsing connector path



Application deployment – Liberty JVM server

1) Liberty dropins directory
   - For development/testing
2) CICS Bundle resource
   - Web application(WAR), JEE archive (EAR), OSGi Application(EBA)
3) Liberty application definition
   - For manual deployment via server.xml
4) Liberty shared bundle repository or global library
   - For shared components



Deployment – 1. Dropins

- To use the Drop-in directory (turned off by default):
  - Update server.xml to enable dropins

```
<applicationMonitor dropins="/u/wakelin/cicsjava/dropins" dropinsEnabled="true“
    pollingRate="5s" updateTrigger="polled"/>
```

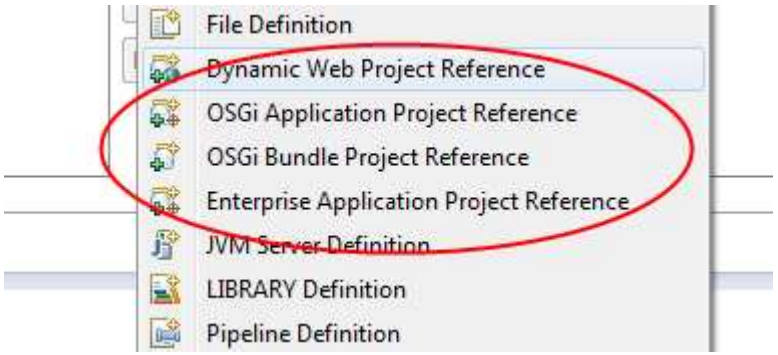
  - FTP the WAR/EAR/EBA file in binary mode to drop-ins zFS dir
    - Directory is automatically created when JVM server is created
    - In default configuration:

    /$WORK_DIR/applid/jvmserver/wlp/usr/servers/defaultServer/dropins

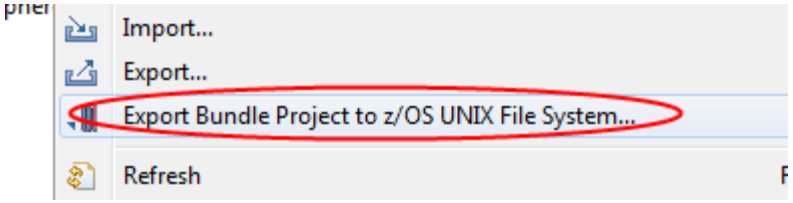
  - Liberty detects the deployed WAR file and installs/updates it
  - CICS is **not** aware of it, no CICS bundle life cycle
  - Not integrated with CICS security
    - Useful for development regions

Deployment – 2. CICS bundles

1. Create CICS bundle in CICS Explorer
2. Add EAR/WAR/EBA reference



3. Export to zFS



4. Define & install CICS BUNDLE resource with BUNDLEDIR attribute referring to #3
   - CICS Bundle disable/enable will recycle application

Deployment – 3. server.xml definitions

- Define application in server.xml or embedded file

```
<application type="war" id="LibertyWorld" name="LibertyWorld"
     location="/u/cicsjava/deploy/liberty.security.helloworld.war">
     <classloader commonLibraryRef="mqlib" />
     <application-bnd>
         <security-role name="testing">
              <user name="WAKELIN" />
         </security-role>
       </application-bnd>
</application>

<library id="mqlib">
  <fileset dir="/mqm/V7R1M0/java/lib" includes="*.jar" scanInterval="5s" />
</library>
```

Locally defined java library

testing role must be defined in application's web.xml

Userid to be given permissions,

Deployment – 4. Shared repositories

1. Shared bundle repository – available to all OSGi components (i.e components deployed in EBAs)

```
<fileset dir="/mqm/V7R1M0/java/lib/OSGi" id="mqosgilib"
   includes="com.ibm.mq.osgi.java_7.1.0.4.jar"/>
<bundleRepository filesetRef="mqosgilib"/>
```

2. Global library – available only to standalone Web applications (WARs/EARs)

```
<fileset dir="/mqm/V7R1M0/java/lib" id=“mqlib" includes="*.jar"/>
<library filesetRef=“mqlib" id="global"/>
```

Starting Liberty in CICS

- Starting a Liberty JVM server in 4 easy steps
  - CICSDev article
    https://www.ibm.com/developerworks/community/blogs/cicsdev/entry/liberty_jvm_servers_a_quickstart_guide?lang=en

1. Add SDFHAUTH and SDFJAUTH to STEPLIB
2. Setup access to output and user zFS directories
3. Create JVM server
4. Add JVM profile properties to autoconfigure Liberty server

```
WLP_INSTALL_DIR=&USSHOME;/wlp
WLP_OUTPUT_DIR=/logs/cics/&APPLID;
WLP_USER_DIR=/logs/cics/&APPLID;/&JVMSERVER;
-Dfile.encoding=ISO-8859-1
-Dcom.ibm.cics.jvmserver.wlp.autoconfigure=true
-Dcom.ibm.cics.jvmserver.wlp.server.name=LIBTEST
-Dcom.ibm.cics.jvmserver.wlp.server.http.port=12345
-Dcom.ibm.cics.jvmserver.wlp.server.https.port=12346
```

- Autoconfigure
  - Creates embedded Liberty server using properties in the CICS JVM profile
  - Requires r/w access to the WLP_USER_DIR directory tree
  - Suited to development environments

Administering Liberty in production

- Support for MQ
  - Add MQ base classes via global library (WARs/EARs) or shared bundle repository (OSGi applications)
  - CICSDev Using the WebSphere MQ classes for Java with a CICS JVM server
- JMS
  - JMS 1.1 & 2.0 support only in OSGi JVM server for IBM MQ V8 & 7.1 in CICS TS V5.2+
  - JMS 1.1 support for Liberty embedded messaging in CICS TS V5.3 open beta
- Support for DB2
  - cicsts:jdbc-1.0 feature
    - Access to Local DB2CONN using T2 d/b driver
    - DriverManager and DataSource APIa
    - Dynamic JDBC and static SQLJ
  - jdbc-4.0 feature
    - Remote database using type4 d/b driver (i.e DB2 or Derby)
    - Dynamic JDBC and DataSource API only
    - XADataSource and Java Transactions (JTA) for 2pc integration with UOW

Demos

- Deploying CICS servlet samples from Explorer
- Deployment of EJB to dropins
- Using Jconsole with JMX to monitor Liberty
- Porting JCA applications to CICS Liberty