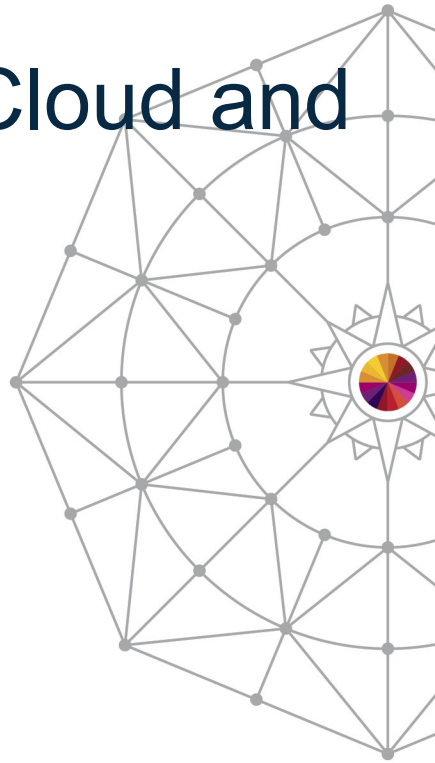
z/OS Connect: Opening up z/OS assets to the Cloud and Mobile Worlds

David Follis
IBM

August 7, 2014
Session Number 15782



#SHAREorg

SHARE in Pittsburgh 2014

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

CICS*
DB2*
GDPS*
Geographically Dispersed Parallel Sysplex
HiperSockets
IBM*
IBM eServer
IBM logo*
IMS
On Demand Business logo
Parallel Sysplex*
RACF*
System z9
WebSphere*
z/OS
zSeries*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Java and all Java-related trademarks and logos are trademarks of Oracle.
UNIX is a registered trademark of The Open Group in the United States and other countries.
Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.
MIB is a trademark of MIB Group Inc.

* All other products may be trademarks or registered trademarks of their respective companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon 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 throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

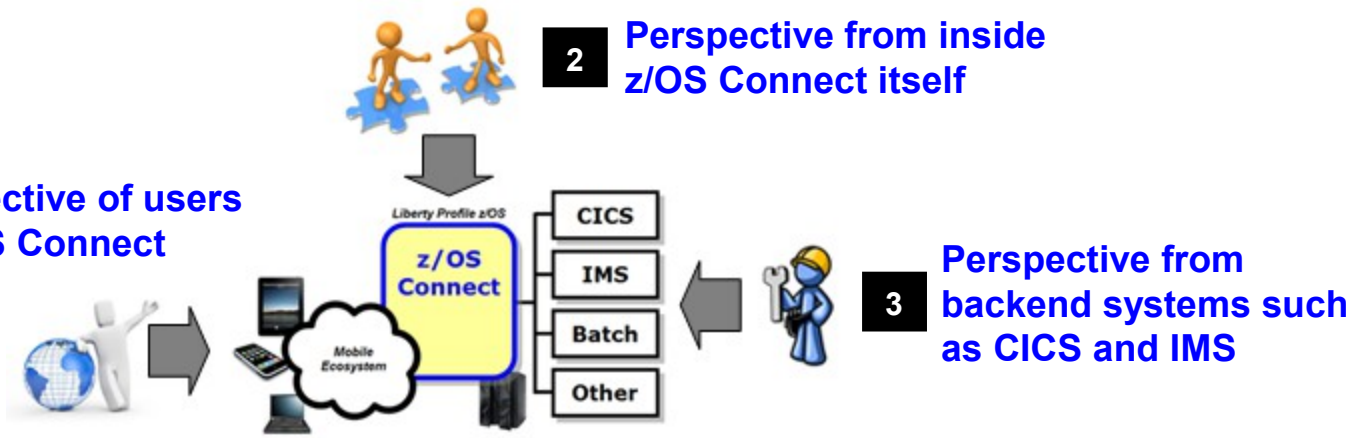
Disclaimer

- The information contained in this documentation is provided for informational purposes only. While efforts were many to verify the completeness and accuracy of the information contained in this document, it is provided “as is” without warranty of any kind, express or implied.
- This information is based on IBM’s current product plans and strategy, which are subject to change without notice. IBM will not be responsible for any damages arising out of the use of, or otherwise related to, this documentation or any other documentation.
- Nothing contained in this documentation is intended to, nor shall have the effect of , creating any warranties or representations from IBM (or its suppliers or licensors), or altering the terms and conditions of the applicable license agreement governing the use of the IBM software.
- Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon 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 throughput improvements equivalent to the performance ratios stated here.
- All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

Outline of Discussion

- **High-Level Overview**
  To establish a foundation of understanding about concepts and terminology, and where z/OS Connect fits within an overall Mobile architecture

- **Perspective Views**
  To provide an understanding of z/OS Connect from three different perspectives:



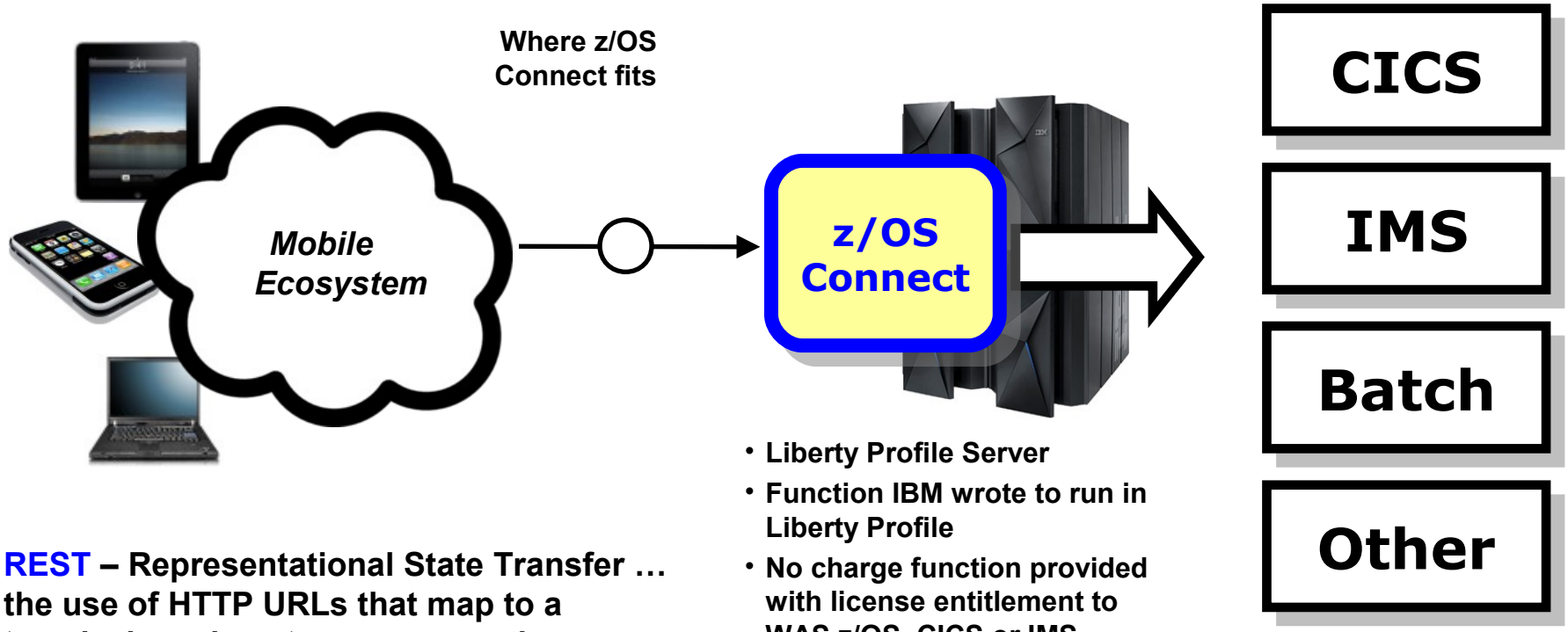
- **Summary**
  Wrap-up

High-Level Overview

Setting Context Before Going into Details

Setting Context

This is about getting REST and JSON into your mainframe environment in a way that enables you to best take advantage of the assets that exist there:



- Liberty Profile Server
- Function IBM wrote to run in Liberty Profile
- No charge function provided with license entitlement to WAS z/OS, CICS or IMS

REST – Representational State Transfer … the use of HTTP URLs that map to a 'service', such as 'query account' or 'update data'

JSON – JavaScript Object Notation … a standard of representing data as a set of name/value pairs. This is passed back and forth along with REST request/responses

Many questions arise – most notably: "Why?" and "What is it?" Then we can begin to answer "How does it work?"

Already Ways of Handling REST/JSON

We already have ways of exposing z/OS resources through REST APIs and using JSON … most notably:

Roll Your Own

REST is essentially HTTP, and JSON can be handled by WebSphere Application Server or Liberty … so you could write your own handler to take in REST/JSON and connect to backend resources

CICS

CICS and the Mobile Feature Pack provides a mechanism for a CICS region to consume REST/JSON and provide access to CICS programs

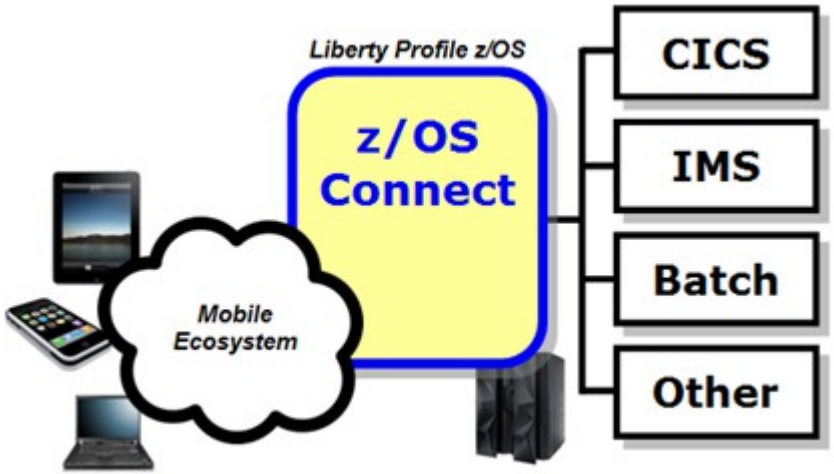
IBM Worklight

A comprehensive Mobile Enterprise Application Platform (MEAP) … but Worklight is not supported on z/OS (is supported on Linux for System z)

What z/OS Connect provides is a common and consistent REST/JSON interface to the mainframe

Why z/OS Connect?

This represents another component to configure and maintain in your environment. So what value does it bring?



You *could* enable Mobile access without z/OS Connect

z/OS Connect simplifies and makes the environment more consistent and manageable

- **Provides a common and consistent entry point for mobile access to one or many backend systems**
- **Java, so runs on specialty engines**
- **Shields backend systems from requiring awareness of RESTful URIs and JSON data formatting**
- **Provides point for authorization of user to invoke backend service**
- **Provides point for capturing usage information using SMF**
- **Simplifies front-end functions by allowing them to pass RESTful and JSON rather than be aware of or involved in data transformation**

Different *Delivery* Approaches

This is planned to be delivered with WAS z/OS, CICS and IMS … objective is to provide different approach paths depending on what you have:

WAS z/OS

Delivered as function that runs inside Liberty Profile z/OS. Initially will use WOLA (WebSphere Optimized Local Adapters) to access backend.

CICS

To be delivered as part of Liberty Profile that runs inside of CICS region. Will use JCICS interface to access CICS functions

IMS

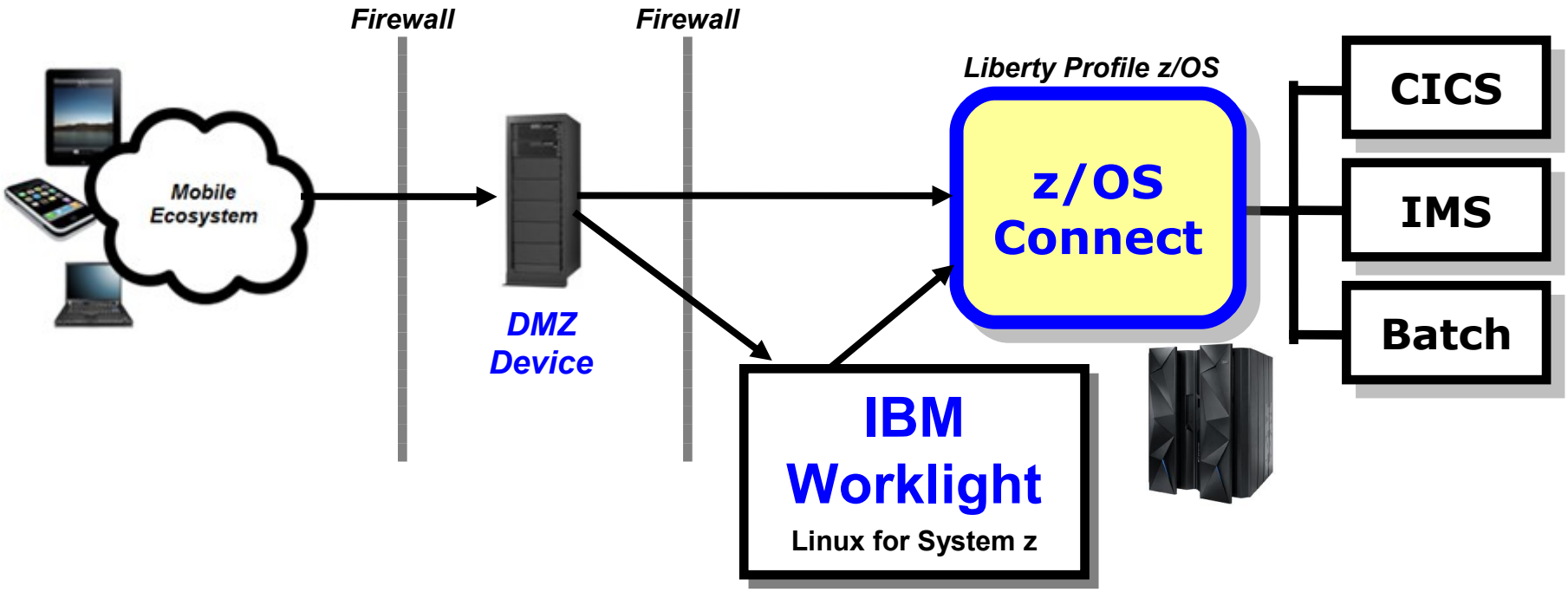
Initially this ends up looking just like the WAS z/OS approach: that is, Liberty Profile z/OS with z/OS Connect inside. Difference is this: IMS z/OS Connect uses JCA* to talk to IMS Connect to get access into IMS.

These different delivery mechanisms tend to obscure the main story of what it is and how it works, so for now let's stipulate IBM offers several ways to get this and now focus on some details

* A supplied IMS JCA resource adapter, as opposed to the local adapter support

Context Within Overall Mobile Architecture

The message here is that z/OS Connect is a *piece* of the Mobile architecture, but in most cases will not be the only component:



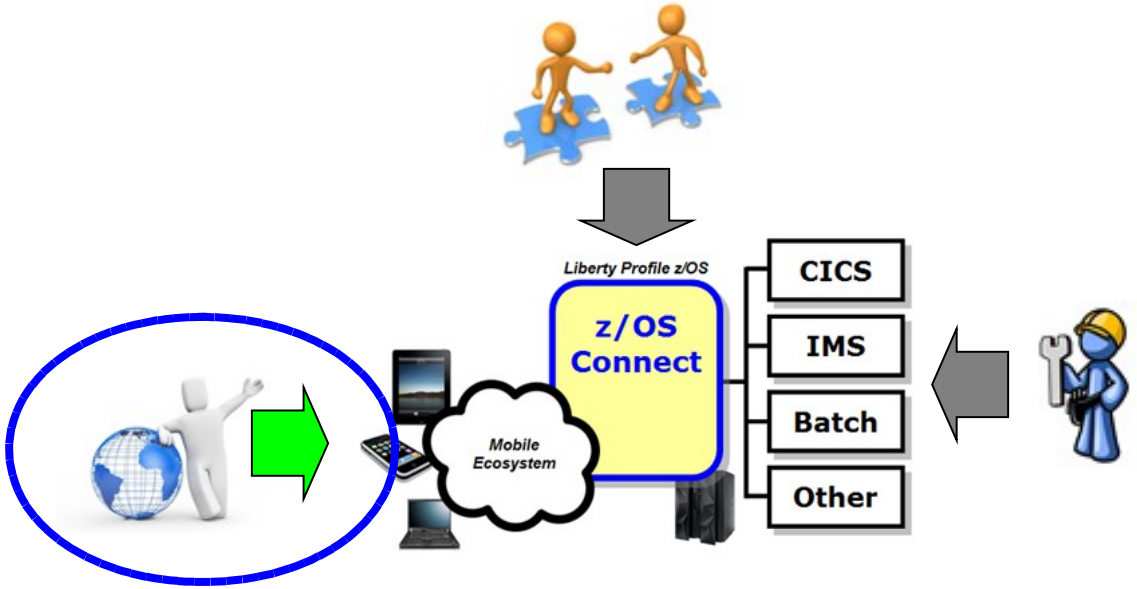
Users of z/OS Connect would access through normal corporate firewall infrastructure

IBM Worklight to provide application management, security and operational governance for mobile applications

z/OS Connect would be behind the secure firewall, and on LPARs along with backend systems

Front-End Perspective

Looking at z/OS Connect from a perspective of users of the function



Users of the z/OS Connect Function

We need to be careful to not limit users to *just* mobile phones …

- **Mobile phones**
  What people traditionally think of when "mobile computing" is discussed.
- **Tablets**
  Related to mobile phones in terms of how they function
- **Cloud-provisioned services**
  A growing consideration … these are programs that are part of a provisioned cloud service that seek information using standard protocols and data formatting. IBM's Bluemix is an example of this.
- **Traditional workstations**
  Any network-connected system or device

The key is *not* the specific type of device or program … the key is what service protocol it uses and what data format it passes

If RESTful and JSON, it can use z/OS Connect

RESTful Services

Stands for Representational State Transfer … this is a protocol built on HTTP, using HTTP verbs*, where the URI indicates the service requested:

URI = Uniform resource identifier

```
https://mysite.com/CustomerApp/getCustomer?cn=1234
```

There's no magic to this … if the URI is understood by the receiving server, then the implied action is taken. What that action is depends on how the server is configured.

Knowing what URIs the server supports is important, which is why z/OS Connect has a discovery function that can be used to query for the configured services and details on those services.

RESTful services are growing in popularity because it's easier to implement than other web service protocols such as SOAP, which involves XML and WSDL and parsing …

*** For example, GET, PUT, POST, DELETE**

JSON – JavaScript Object Notation

It is a way of passing data back and forth as a series of name/value pairs.

URI = Uniform resource identifier

```
https://mysite.com/CustomerApp/update?cn=1234
```

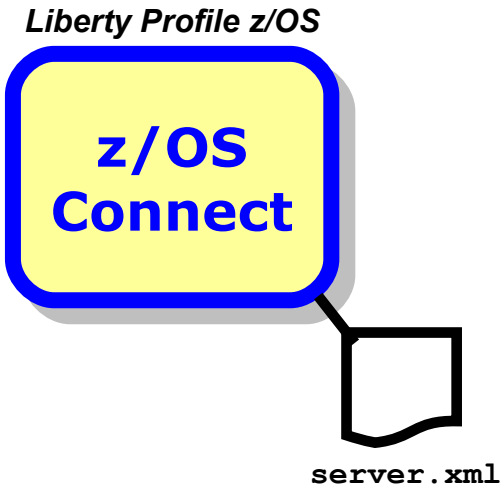
```
{
    "firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": "1234 Main Street",
        "city": "Anytown",
        "state": "NY",
        "postalCode": "10021-1234"
    },
}
```

The data being passed in is appended to the URI and passed in to the server

JSON can be passed back to the client as well.

A Peek Inside z/OS Connect Configuration

We're about to show you the discovery function, which returns a list of configured services. Let's first look inside z/OS Connect …



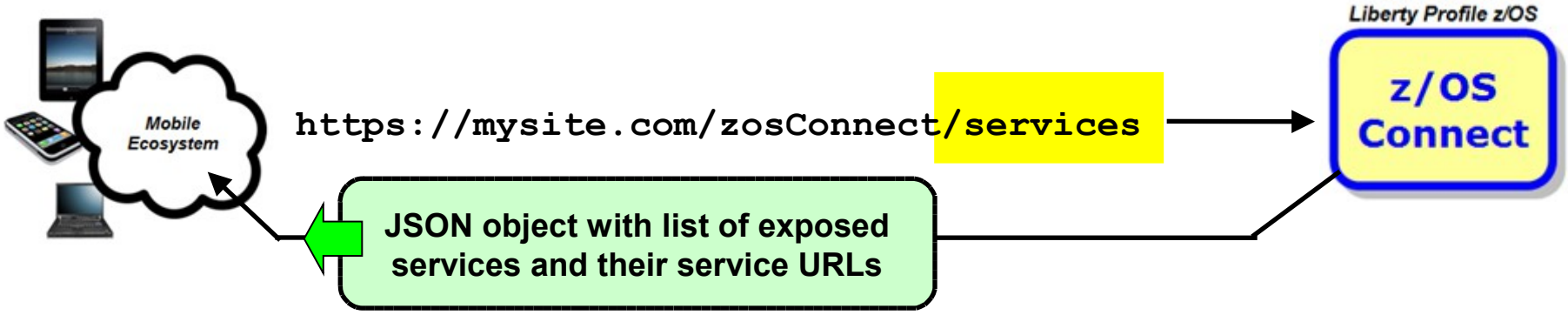

```
<feature>zosConnect-1.0</feature>

<zosConnectService serviceName="serv1" ... />
<zosConnectService serviceName="serv2" ... />
<zosConnectService serviceName="serv3" ... />
<zosConnectService serviceName="serv4" ... />
```

- **For z/OS Connect to understand what URIs to handle and how to handle them, you need to configure that information into `server.xml`**
  More information on server.xml configuration coming up
- **The sample above is a simplified representation of the configuration for multiple services**
- **z/OS Connect understands its environment based on this configuration data … *and it can provide information back using a discovery function***

z/OS Connect Discovery Function

A discovery function is provided to allow developers to query for a list of configured services, and drill down for details on a given configured service:

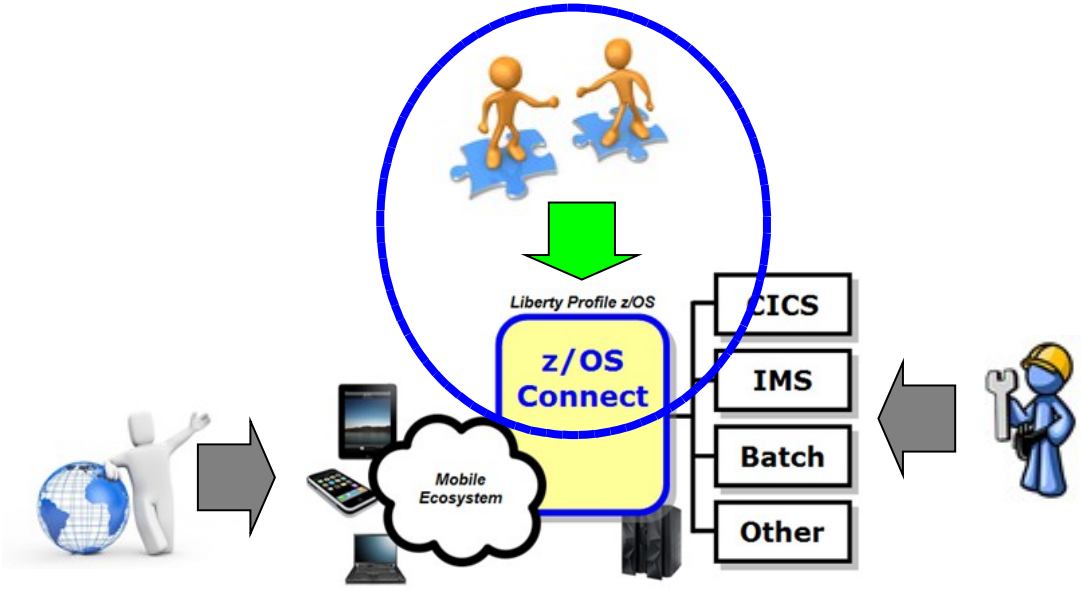
Query for configured services



Query for details on a given configured service

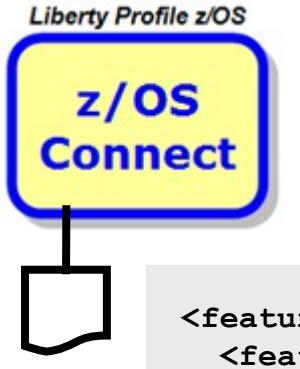
Inside Perspective

Looking at how z/OS Connect works inside the function itself



Overview of server.xml Updates for z/OS Connect

z/OS Connect behavior is defined by updates to the server.xml of the Liberty Profile in which z/OS Connect operates:



`https://myhost.com/zosConnect/services/HelloWorld`

z/OS Connect is defined as a feature in the Feature Manager section, along with WOLA used to get to the backend

```
<featureManager>
  <feature>zosConnect-1.0</feature>
  <feature>zosLocalAdapters-1.0</feature>
</featureManager>

<zosConnectService id="serv1"
  serviceName="HelloWorld"
  serviceRef="wolaHelloService"
</zosConnectService>

<authData id="cauth1" user="user1" password="{xor}LDo8Ki02KyY="/>
<connectionFactory id="wolaCF" jndiName="eis/ola" containerAuthDataRef="cauth1" >

  <properties.ola/>
</connectionFactory>

<localAdaptersConnectService id="wolaHelloService"
  wolaRegisterName="CICSREGA"
  wolaServiceName="PROGRAM1"
  connectionFactoryRef="wolaCF"
</localAdaptersConnectService>
```

This defines a service – HelloWorld – that will be exposed to Mobile users. It maps to a service provider that defines how the backend CICS will be accessed.

This defines the service provider – WOLA in this case – that provides the connectivity to the backend CICS region

Multiple Configured Services

You provide definitions for each service you wish to expose using z/OS Connect. For example, this shows Update, Create and Delete:

```
https://myhost.com/zosConnect/services/ContactCreate
https://myhost.com/zosConnect/services/ContactUpdate
https://myhost.com/zosConnect/services/ContactDelete
```

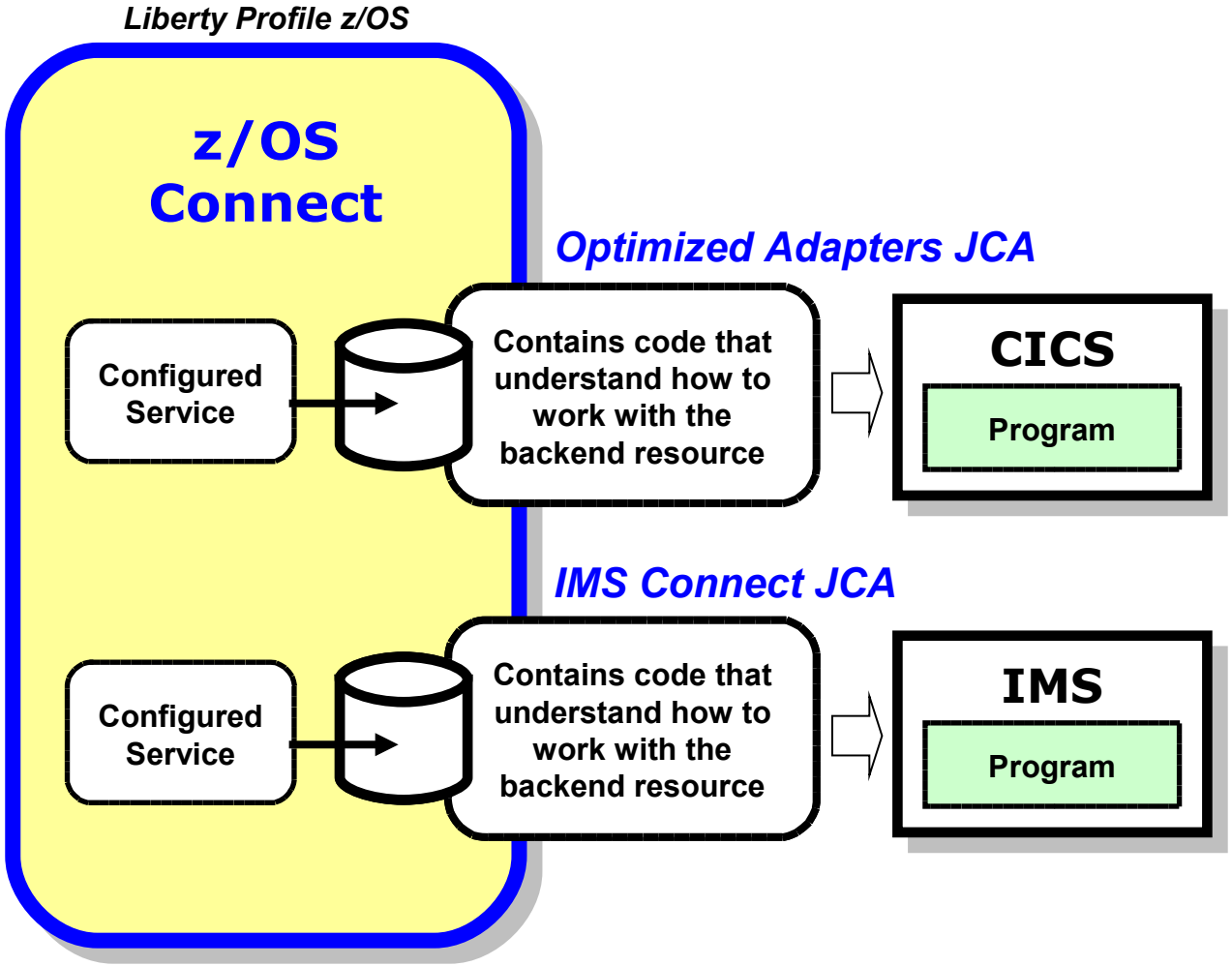
```
<zosConnectService id="zcs3" serviceName="ContactCreate"
     serviceDescription="Create"
     serviceRef="wolaKIXPHONE"
     dataXformRef="xformJSON2Byte">
</zosConnectService>

<zosConnectService id="zcs4" serviceName="ContactUpdate"
     serviceDescription="Update"
     serviceRef="wolaKIXPHONE"
     dataXformRef="xformJSON2Byte">
</zosConnectService>

<zosConnectService id="zcs5" serviceName="ContactDelete"
     serviceDescription="Delete"
     serviceRef="wolaKIXPHONE"
     dataXformRef="xformJSON2Byte">
</zosConnectService>
```

Pointer to section in XML that defines the "Service Provider," which provides connectivity to the backend system

Pointer to section in XML that provides information on how to do data transformation

Service Providers

A 'service provider' is what provides connectivity to a specific backend resource. You may have one or more service providers configured:



To connect to a backend CICS region, for example, you need to specify which CICS region and what mechanism is to be used to connect

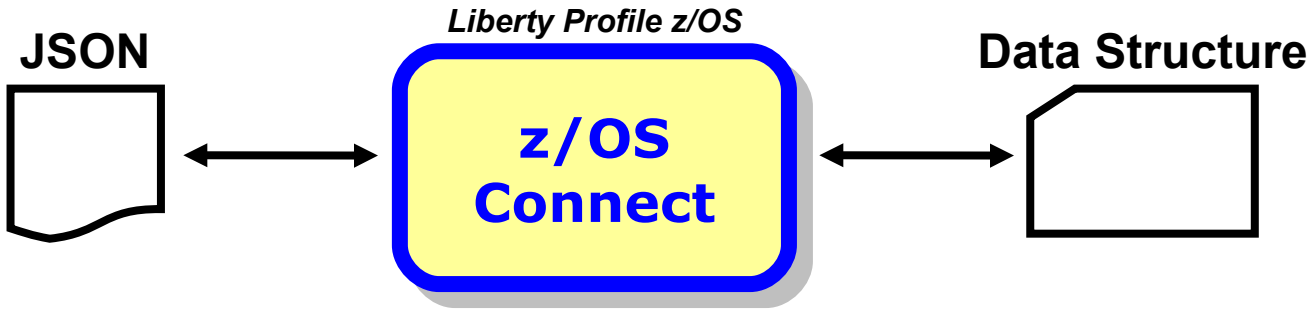
A 'service provider' definition provides that information to z/OS Connect

Configured services are tied to service provider definitions to complete the circuit

Multiple service providers are supported, as illustrated here

Data Conversion

z/OS Connect accepts JSON data, but then needs to convert that to the data format required by the backend program. Data conversion provides that:




```
<zosConnectDataXform id="xformJSON2Byte"
    bindFileLoc="/u/user1/bindfiles"
    bindFileSuffix=".bnd"
    requestSchemaLoc="/u/user1/schema"
    responseSchemaLoc="/u/user1/schema"
    requestSchemaSuffix=".json"
    responseSchemaSuffix=".json">
</zosConnectDataXform>
```

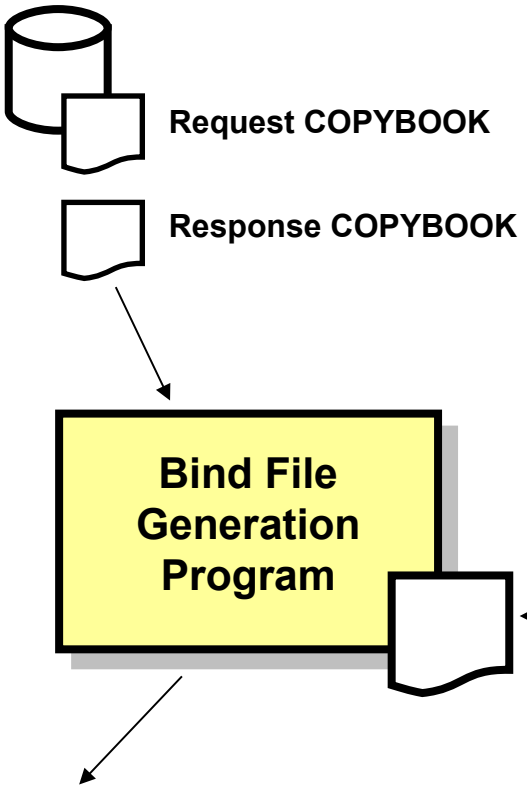
ID of transform section that was referenced in the service definition

XML that tells z/OS Connect where the bind file is located and the file suffix that is used

XML that tells z/OS Connect where the JSON schema information is located

Bind Files

Bind files are generated with a supplied utility. Bind files provide z/OS Connect with knowledge of how JSON maps to the target data structure



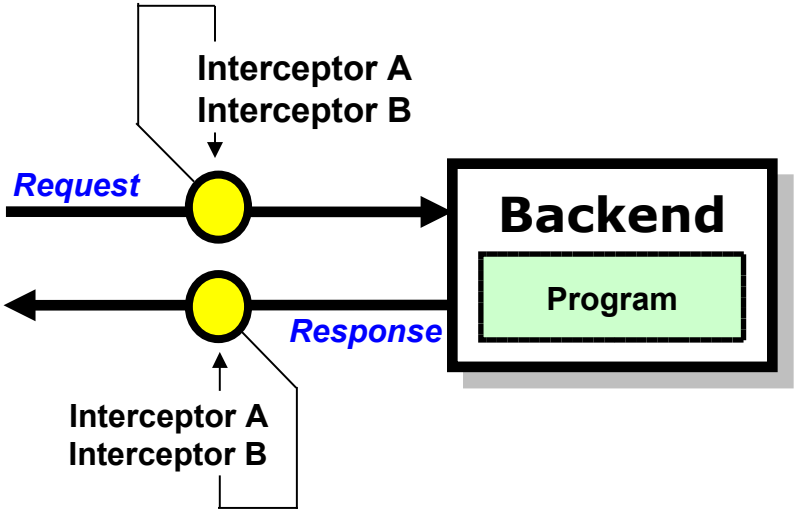

```
//JOB ...
//INPUT.SYSUT1 DD *
PDSLIB=ZCONNECT.CNTL
REQMEM=REQDATA
RESPMEM=RESPDATA
STRUCTURE=(requestData,responseData)
JSON-SCHEMA-REQUEST=/u/user1/schema/GETCUSTC_request.json
JSON-SCHEMA-RESPONSE=/u/user1/schema/GETCUSTC_response.json
PGMINT=COMMAREA
PGMNAME=GETCUSTC
WSBIND=/u/user1/bindfiles/GETCUSTC.bnd
/*
```

- **Bind File**
- **JSON Schema**

The bind file provides the JSON-to-COPYBOOK mapping; the JSON schema files are used for the `getRequestSchema` and `getResponseSchema` methods

Overview of Interceptors

The interceptor framework provides a way to call code to do pre-invoke work and then again to do post-invoke work:



In `server.xml` you can:

- **Define 'global interceptors,' which apply to all configured services**
- **Define interceptors specific to a given configured service**
- **Have services 'opt out' of global interceptors**

z/OS Connect will come with a security interceptor (for authorization) and an audit interceptor (for SMF recording)

It is also possible to write your own interceptor and have it called as part of request/response processing

Audit (SMF) Interceptor

The audit interceptor writes SMF 120.11 records with the following information captured:

Liberty Profile z/OS

- System Name
- Sysplex Name
- Jobname
- Job Prefix
- Address Space Stoken

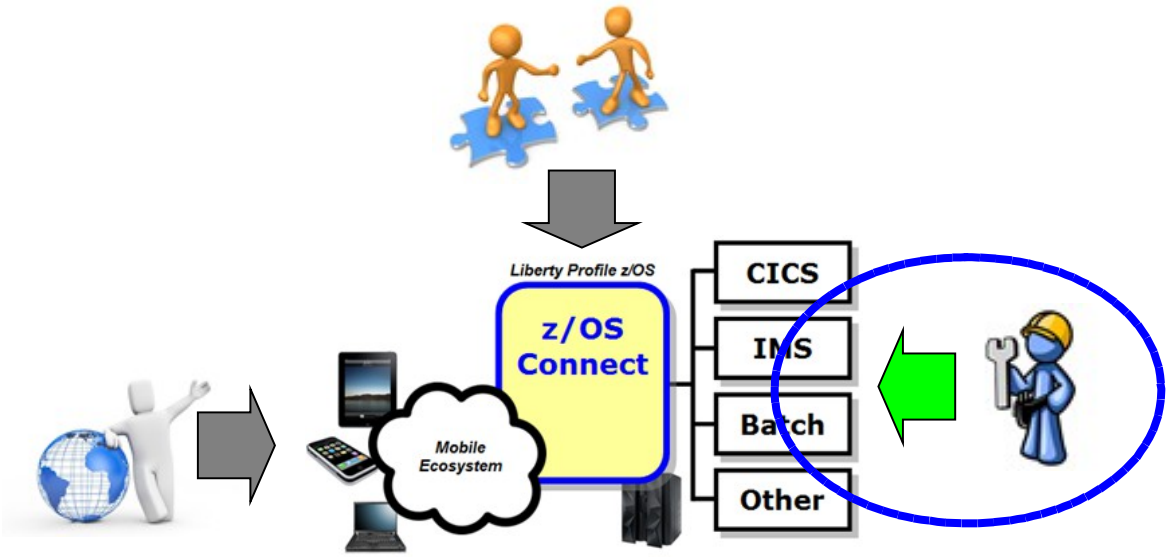
Server Identification Section

- Arrival Time
- Completion Time
- Target URI
- Input JSON Length
- Response JSON Length
- Method Name
- Service Name
- Userid

z/OS Connect User Data Section

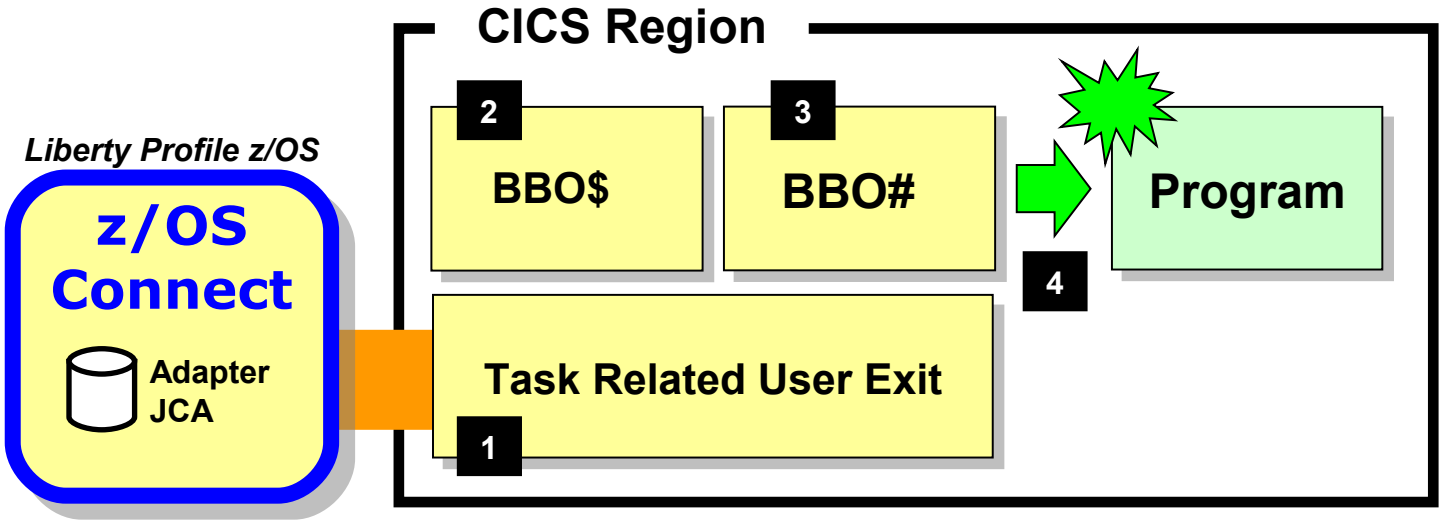
Backend Perspective

Looking at z/OS Connect from perspective of CICS, Batch and IMS



Optimized Adapters Support in CICS

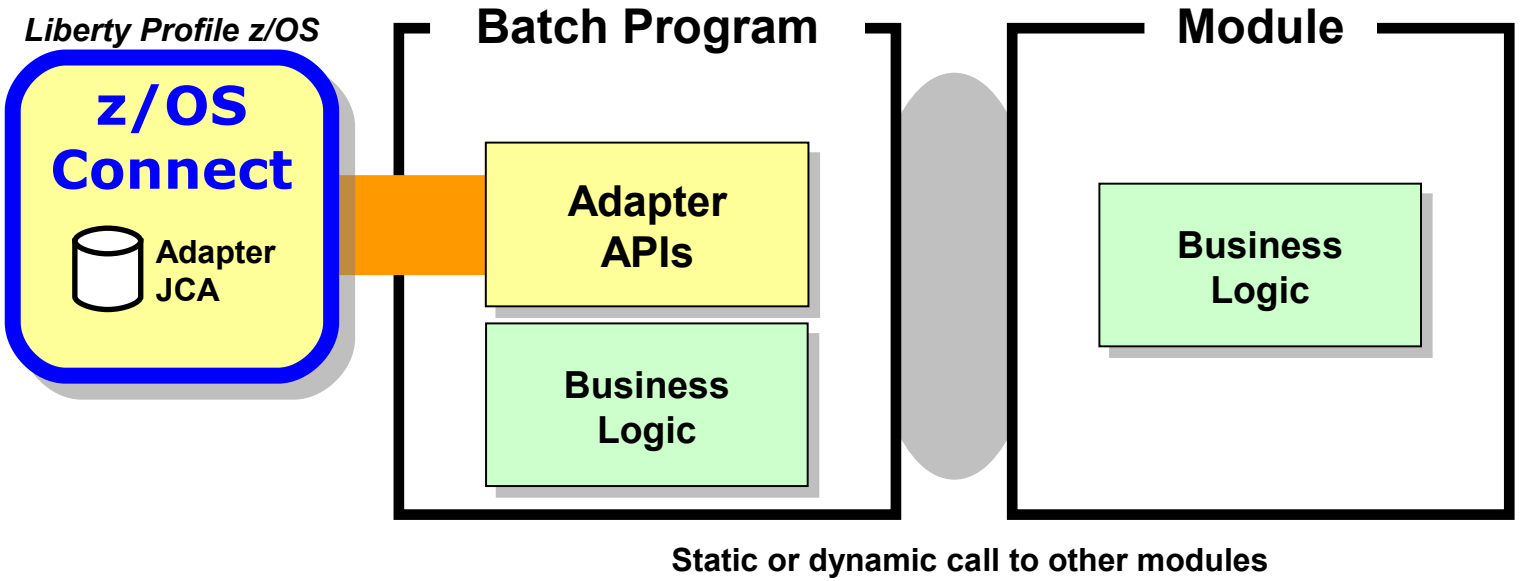
To support WOLA in CICS the following code must be installed. JCL samples provided to do the installation:



1. The WOLA TRUE provides the low-level connectivity for cross-memory communications using Optimized Adapters.
2. The BBO$ long running task handles the calls coming from z/OS Connect. The BBO$/BBO# combination shields the target program from any awareness of Optimized Adapters.
3. The BBO# invocation task is used to perform the EXEC CICS LINK to the target service.
4. Finally, an EXEC CICS LINK is performed and the target service invoked with the data from z/OS Connect passed in. The target CICS program has no knowledge of Optimized Adapters.

Optimized Adapters and Batch Programs

The Batch Program would use the APIs to "Host a Service" and wait for call from z/OS Connect:



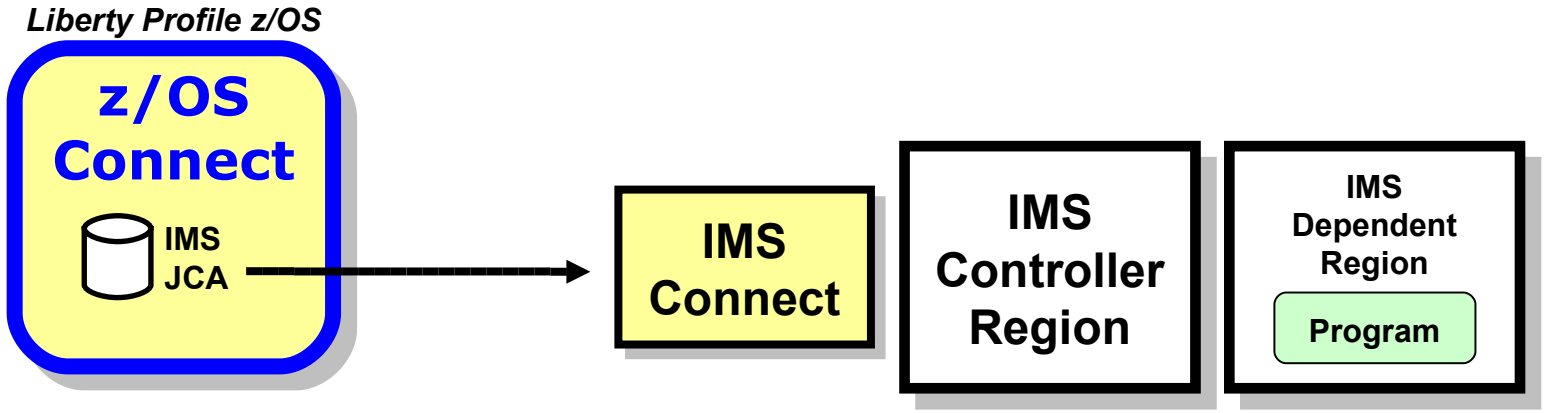
When the target of the z/OS Connect call is a Batch Program, that program will need to use WOLA APIs to "Host a Service"

BBOA1SRV, BBOA1RCA or BBOA1RCS

Business logic being called may reside in the hosting batch program, or that program may perform static or dynamic calls to another module where business logic resides

z/OS Connect and IMS

z/OS Connect interacts with IMS using function provided by IMS and announced in letter ENUS214-220:



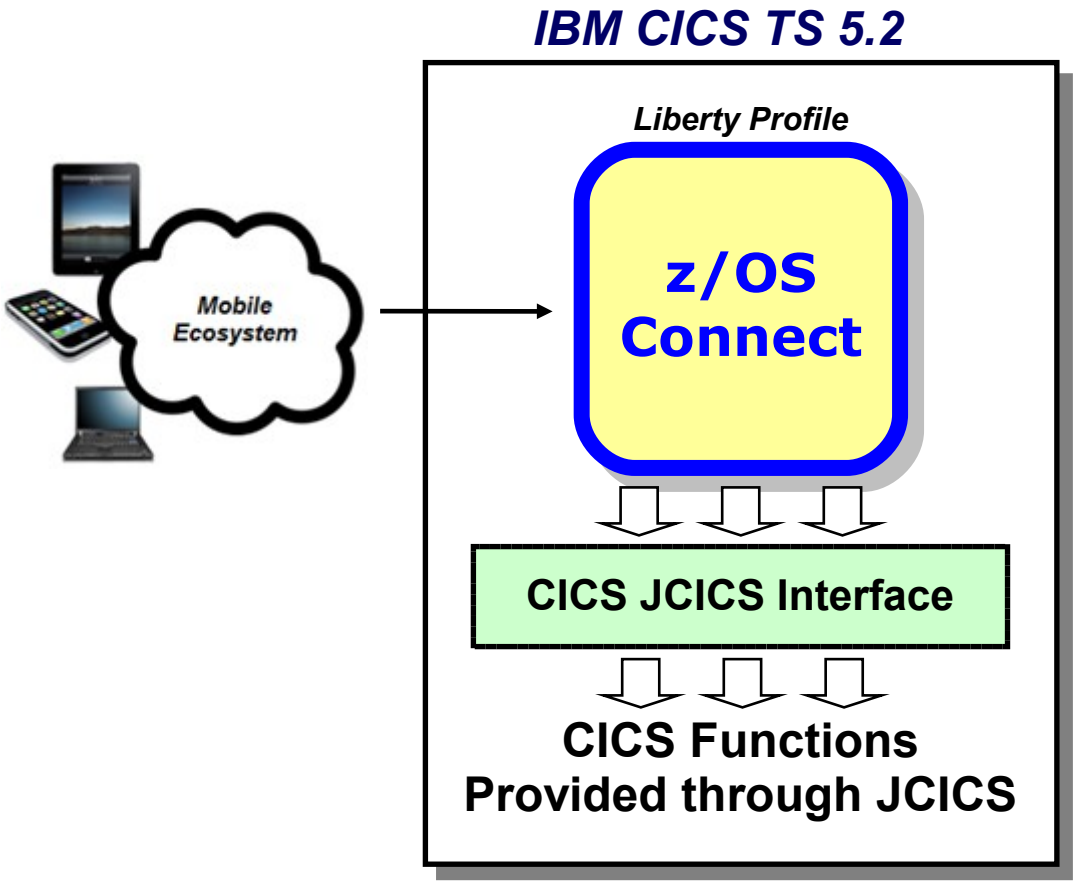
From a z/OS Connect perspective the concepts are the same as what we've shown so far. The syntax for the service provider would be different (IMS Connect vs. Optimized Adapters). And the pointer to the JCA resource adapter would specify the IMS JCA.

It is possible to use the Liberty Optimized Adapters with IMS, but only if the program uses the optimized adapter APIs to 'host a service' like described for Batch programs

ibm.com/common/ssi/cgi-bin/ssialias?infotype=AN&subtype=CA&htmlfid=897/ENUS214-220&appname=USN

z/OS Connect and CICS

z/OS Connect interacts with CICS using JCICS interface as documented in IBM US announcement letter ENUS214-107:



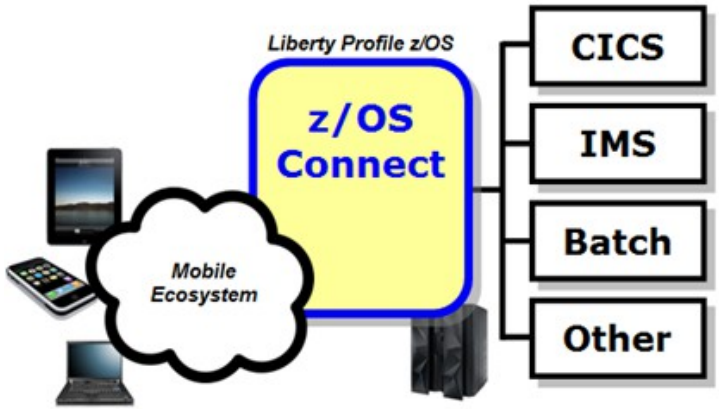
Liberty Profile runs *inside* the CICS region

The 'Service Provider' is only JCICS … not WOLA

ibm.com/common/ssi/cgi-bin/ssialias?infotype=AN&subtype=CA&htmlfid=897/ENUS214-107&appname=USN

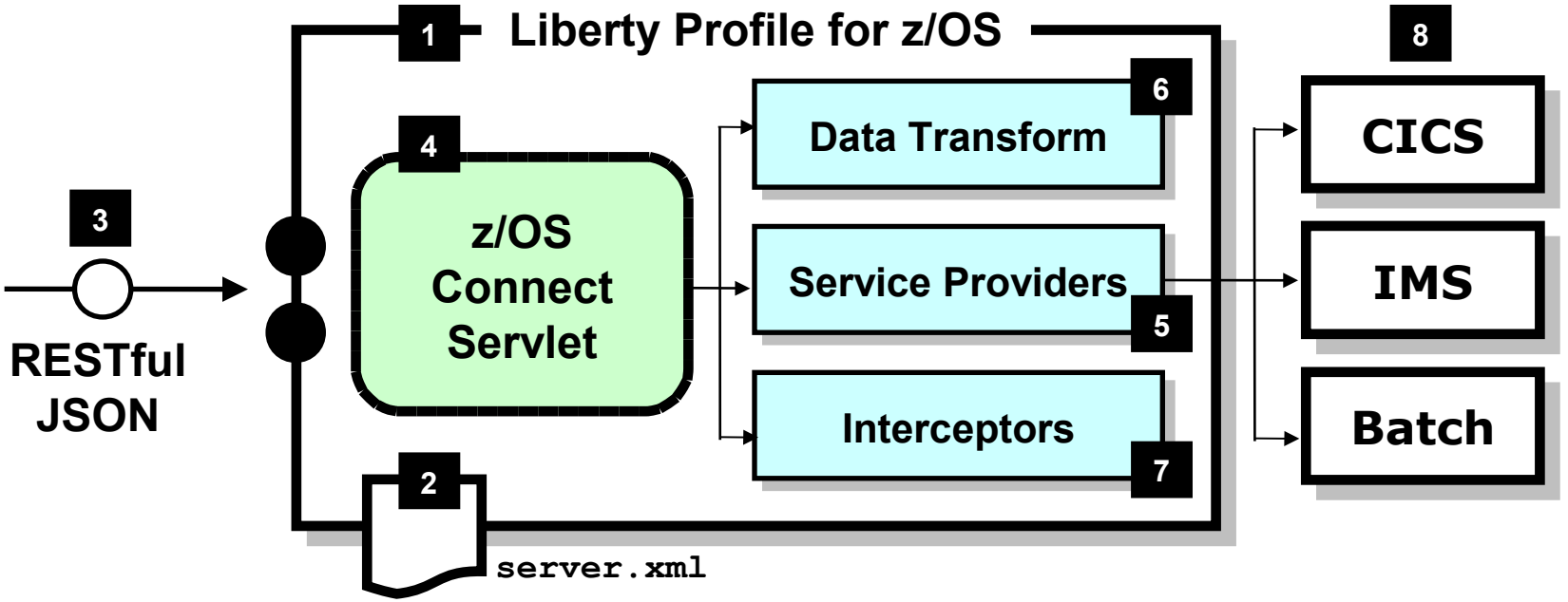
Summary

Wrap-up of Presentation



Summary in One Picture

A summary of what we covered:



1 z/OS Connect is software function that runs in Liberty Profile for z/OS.

2 z/OS Connect is described and configured in the Liberty `server.xml` file

3 z/OS Connect is designed to accept RESTful URIs with JSON data payloads

4 One part of z/OS Connect is a servlet that runs in Liberty Profile z/OS.

5 A 'Service Provider' is software that provides the connectivity to the backend system

6 z/OS Connect provides the ability to transform JSON to the layout required by backend

7 'Interceptors' are callout points where software can be invoked to do things such as SAF authorization and SMF activity recording

8 Initially the backend systems supported will be CICS, IMS and Batch

MOBILE MAINFRAME APP THROWDOWN

Will you be our mobile champ?

CICS | IMS | WAS | DB2

Open to existing System z clients

The challenge: Build a proof-of-concept demonstrating mobile enablement of your existing mainframe apps.

Get IBM help to build your mobile PoC

Call us 'Coach': We provide getting started guides and access to IBM zMobile Experts for questions and queries.

Win a week with IBM experts & more

Make it real: Win help from IBM to bring your mobile app to life.

ibm.biz/mmathrowdown

No submission of code required, only screenshots.
Entries must be complete and submitted by **17 Sept 2014**.