Exposing mainframe applications and services to mobile

Track 3: Extending the mainframe to the mobile enterprise

Where we are in todays agenda

- Mobilizing the mainframe
- Modernizing mainframe applications for mobile and more
- Exposing mainframe applications and services to mobile
- Developing an IBM MobileFirst platform application for z Systems
- Optimizing applications and data for mobile workloads
- Client use cases and getting started with mobile and z Systems

Agenda

- The API Economy – Extending the enterprise
- IBM API Management
- IBM WebSphere Liberty z/OS Connect
- Messaging with IBM MQ for z/OS

The API Economy

APIs - the building blocks for apps

Application Programming Interface

The **"API economy"** has changed how developers think about building apps, and how organizations deploy software in the cloud

API Economy

Mobile App Assembly

IBM APIs
- Watson
- Cloud Provisioning
- Xtify

Insurance APIs
- Life
- Home
- Auto
- Claims

Bank APIs
- Mortgage
- Online Payment
- Loans
- Account Query
- Calculators

Auto Dealer APIs
- Price
- Availability
- Location
- Configuration

Map Provider APIs
- Address
- Locator
- Weather
- Traffic

Mobile App Assemblers

Developers & Partners

mint.com

Zillow.com

New Customers

Providers | **Consumers**

API Economy

Lifecycle

Business Assets
Exposable Enterprise Services

Web APIs
APIs providing commercial access to the Business Assets

Developers
Use APIs to create Apps

Apps
Use backend services through Web APIs

End Users
Increase revenue by using Apps with Business Assets

The API Economy

Where companies [providers] expose their (internal) digital business assets or services in the form of (Web) APIs to third parties [consumers] with the goal of unlocking additional business value through the creation of new assets

Create new business opportunities by extending "reach" with APIs for accessing enterprise assets

Opportunity

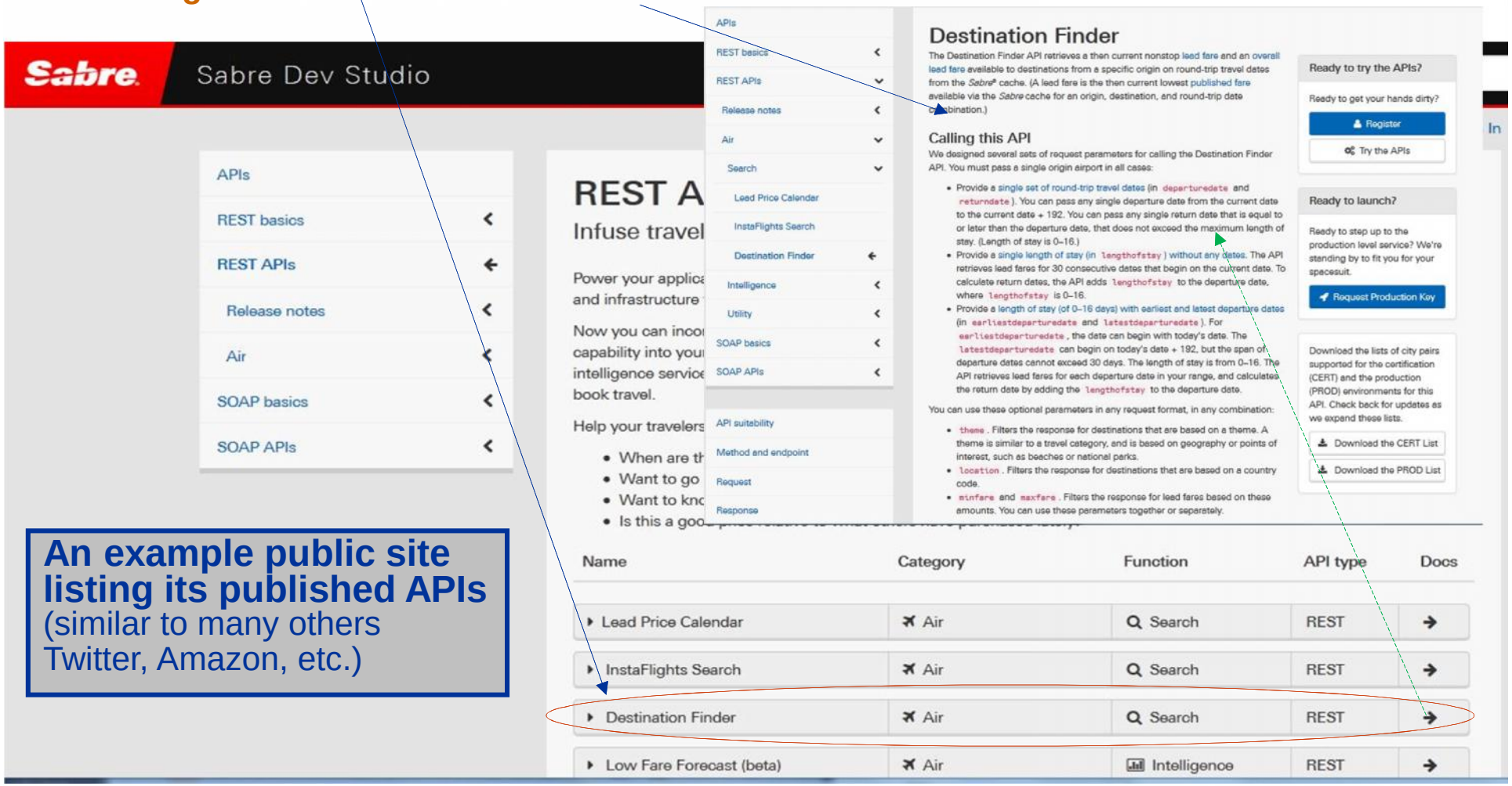
z Customers have made a significant investment over a long period of time, in developing many valuable assets supporting core business functions (and core competencies)

- Leverage these existing assets by exposing as **APIs**, improving both **consumability** and **governance**
- **Mobile, Cloud** based or **third-party** applications can invoke these APIs for accessing **core** business functions

- ✓ ***Reach new customers and markets*** *with new applications and solutions accessing core (z based) business functions, through business partners*
- ✓ ***Improve user experience of existing applications and customers with Mobile applications*** *and new services*
- ✓ ***Improve consumability and governance of enterprise services*** *with API catalog, business control (policy enforcement) and insight (e.g., accountability & chargeback) for the internal and external developers*
  - *simplify access to z assets*

Making APIs consumable requires publishing details on their use, making it easy for app developers

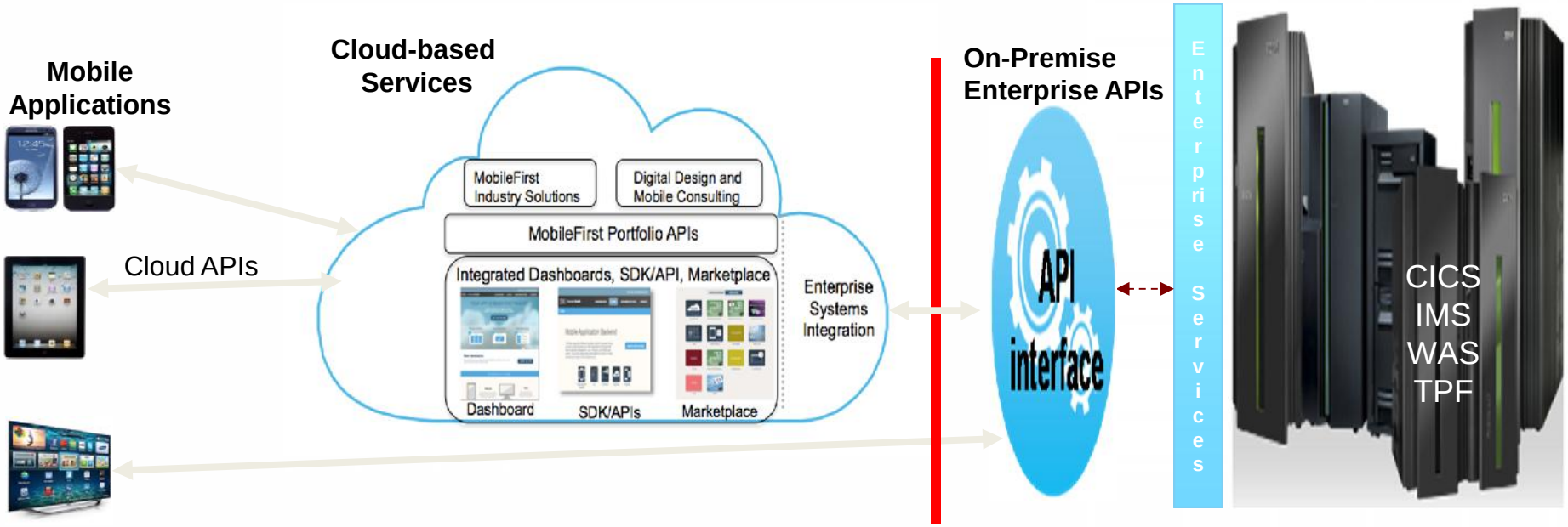
- Listing and categorizing APIs for easy to find
- Describing details on how to invoke an API



An example public site listing its published APIs
(similar to many others Twitter, Amazon, etc.)

Why API Management?

Business challenges addressed by APIM in exposing z based services/assets



Mobile, Cloud and Third-party Applications invoking z Services using APIs

1. ***Consumability of the APIs is Key:***
   - ***Easier creation and assembly of API from existing assets/services***
   - ***Visibility of APIs to internal and external developers***
   - ***Easier registration (by consuming applications) and set up including managing entitlement***
2. ***Retaining business control (e.g., enforcing entitlement, accountability/chargeback) and gaining business insight in API invocation***
   - ***Securing APIs using a secure GW from unwanted external invocations (mapping to application level security) and enforcing workload entitlement***
   - ***Business Monitoring of API access in gaining business insight on the use of APIs by external applications, and for accountability/chargeback***

Businesses are transforming themselves to participate in the API economy

IBM API Management

Expose business services securely as APIs to developer communities, and analyze API usage

Provide self-service API portals to external/internal app developers

Manage & monitor the entire API platform

Off-premise SaaS

On-premise private

Off-premise private

Hybrid

IBM API Management

IBM API Management:
"A Complete API Management Solution"

1
Create, assemble and define an API

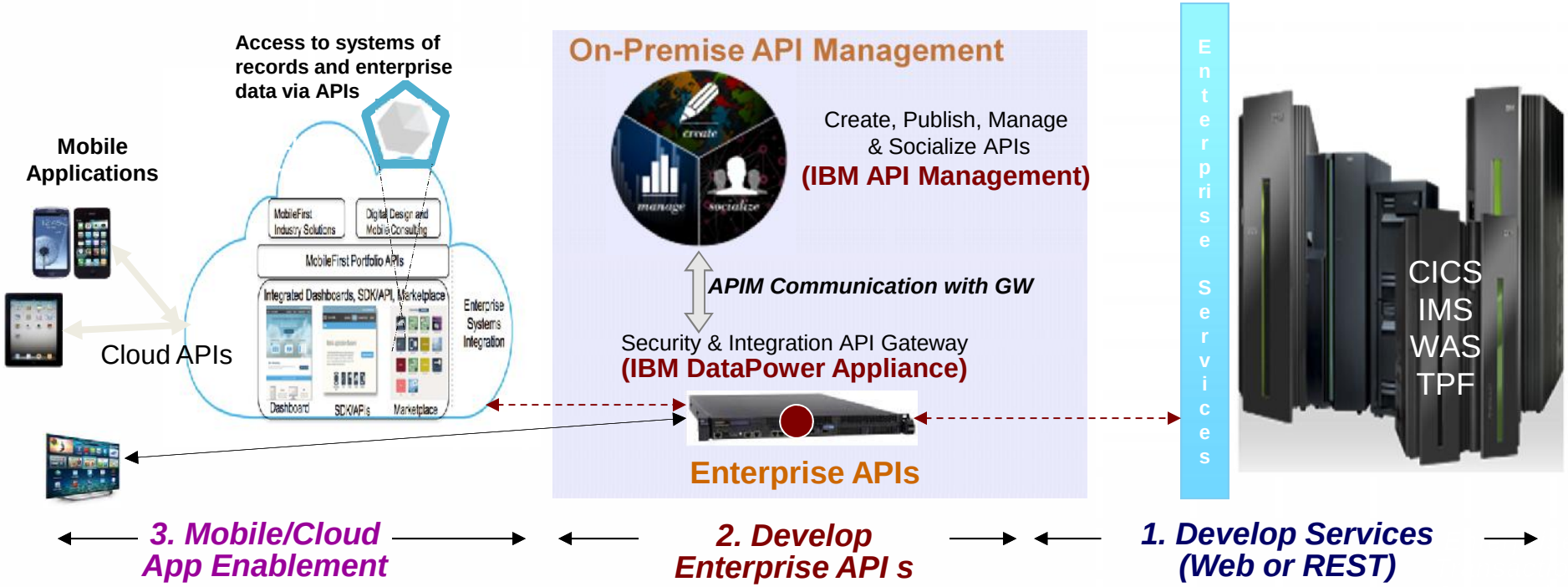
2
Secure & scale the API

3
Share with developers

4
Manage growth & analyze results

13

End-to-End Architecture for Mobile, Cloud and Third-party Applications accessing z Assets/Services using APIs



Mobile/Cloud/Third-party Application Development

- **Invokes APIs for accessing SOR**

API Management

- **Consumability by internal and external developers (creation and look up)**
- **Entitlement Management (securing, workload enforcement)**
- **Usage monitoring & Analytics**

Service Enablement

- **Enables invocation of z applications by remote applications using standard protocols (WSDL, REST)**
- **Converts SOAP or JSON into application specific (e.g., COBOL, PL/I) data and invokes applications**

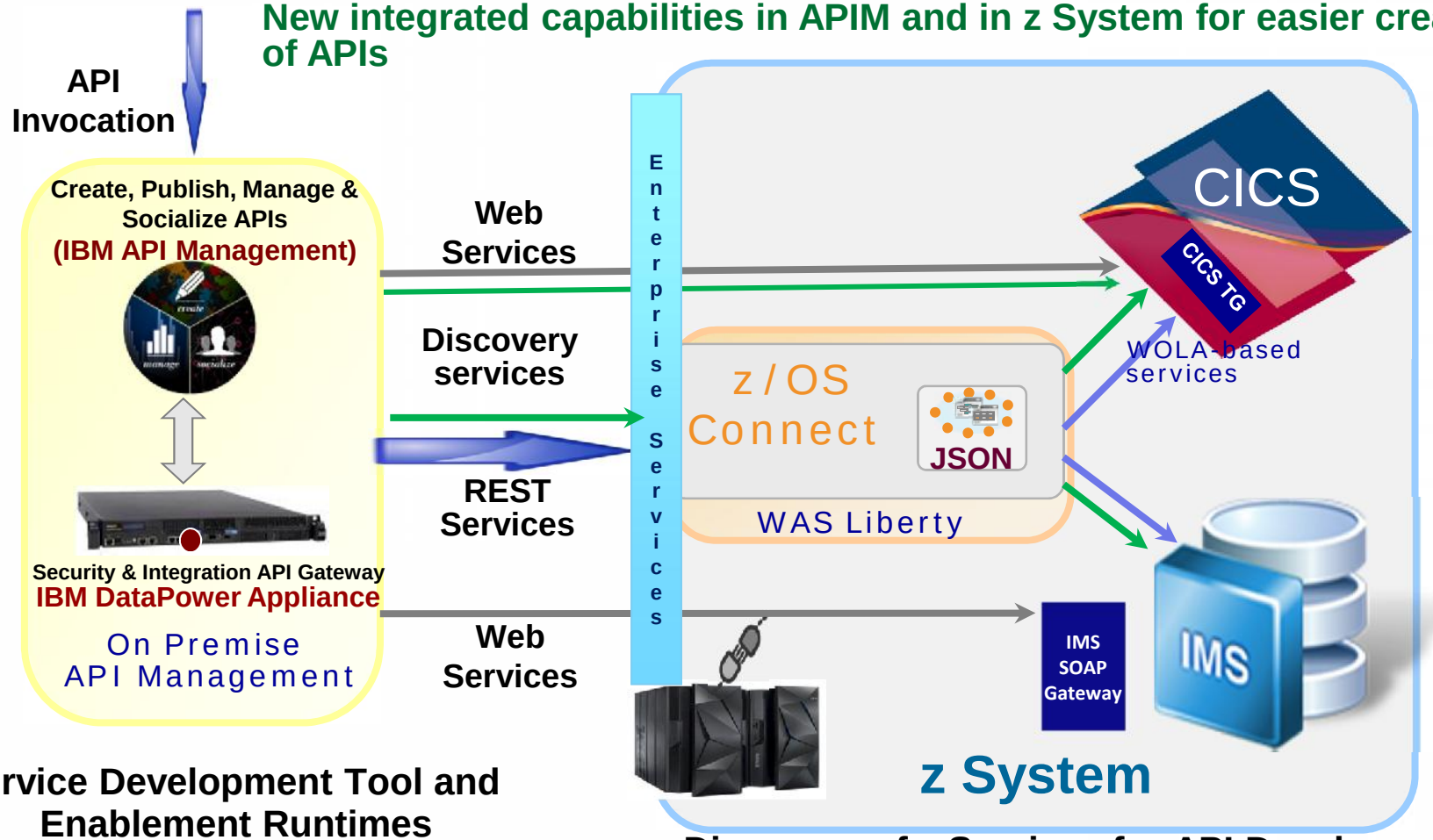
Jane - Mobile app developer
Uses APIs to access Back-End services

Shavon – API developer
Develops APIs from z based services

Bob – developer of z based Services
Develops services from CICS, IMS and other z applications

Discovery and invocation of z Systems based Services

New integrated capabilities in APIM and in z System for easier creation of APIs



Service Development Tool and Enablement Runtimes

- ✓ **Web Services**
  CICS and IMS provides separate tools and runtimes; TPF provides runtime libraries
- ✓ **REST/JSON**
  CICS and IMS use common z/OS Connect runtime

Discovery of z Services for API Development

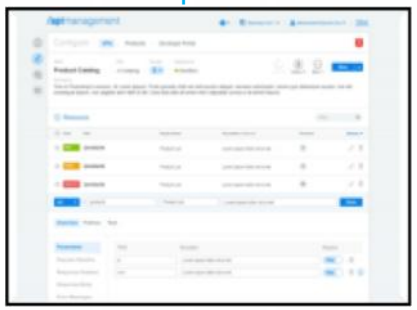
1. *Get a list of deployed services* (Service Identification)
   - *Filter based on technical and business service attributes*
2. *Get schema for a specific service* (API Definition)
3. *Get additional deployment details for a service* (API Assembly)
   - *e.g., security protocol support, invocation uri*

A single, comprehensive solution to design, secure, control, publish, monitor & manage APIs

IBM API Management

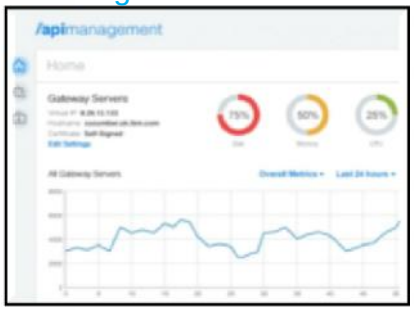
Fully on-premise, multi-tenant solution, for API providers

Developer Portal



API Manager

Management Console



IBM DataPower

API Gateway for security, control, integration & optimized access to a full range of Mobile, Web, API, SOA, B2B & Cloud workloads

Over a decade of innovation, 10,000+ units sold, 2000+ customer installations worldwide

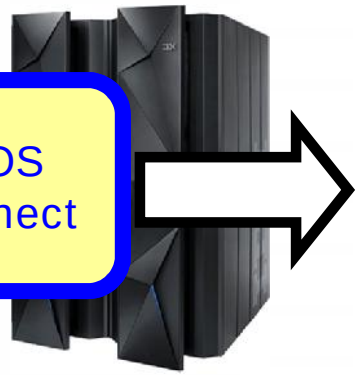
z/OS Connect

z/OS Connect – What is it?

It's 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:

Where z/OS Connect fits

z/OS Connect



- 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

CICS

IMS

Batch

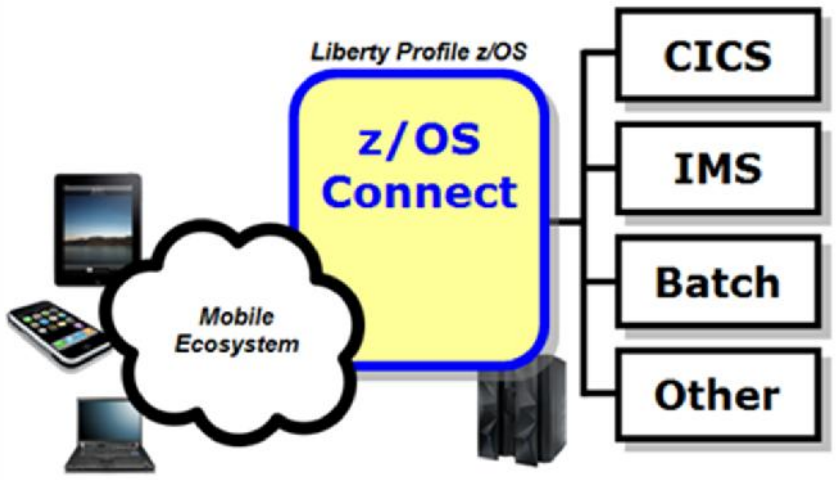
Other

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

Why z/OS Connect?

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

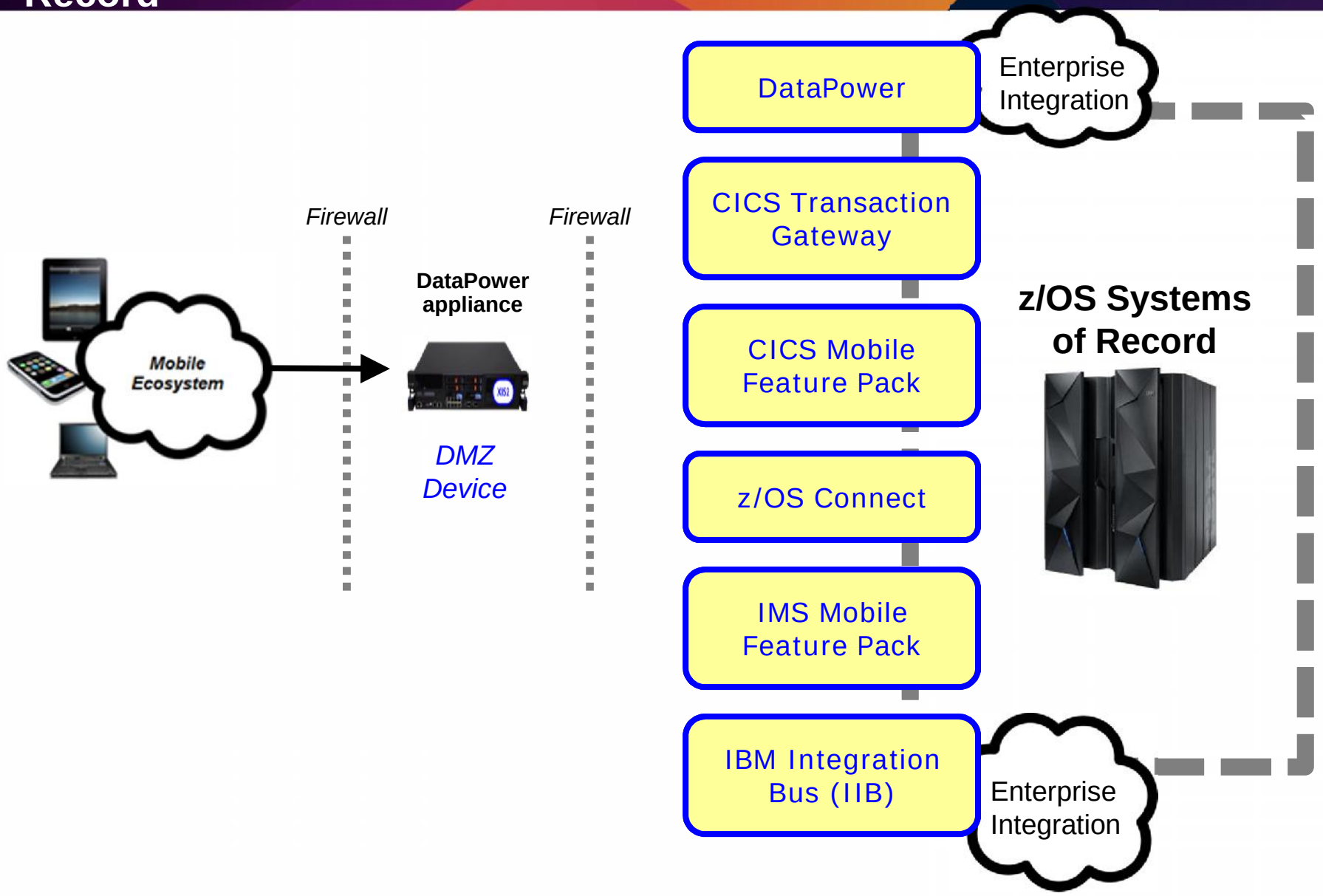


You *could* enable Mobile access without z/OS Connect

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
- Extensible Framework!

Positioning IBM solutions: Accessing z Systems of Record



Different Delivery Approaches

This is 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

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

IMS

Delivered as part of IMS Enterprise Suite (v3.1.1), or downloadable from the Liberty repository, it also has an Eclipse-based tooling platform for service creation and administration (IMS Explorer)

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

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.

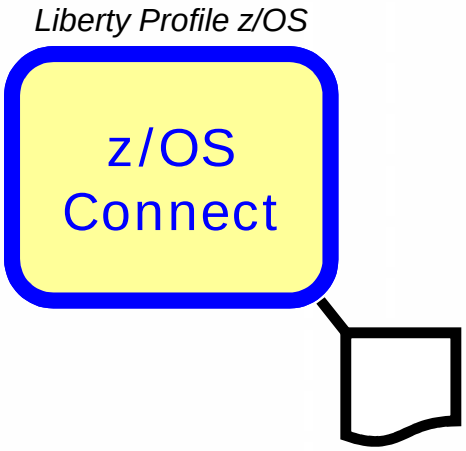
Knowing what URIs the server supports is important, which is why z/OS Connect has a discovery function that can be used to query for the configured services and details on those services.

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

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" ... />
```

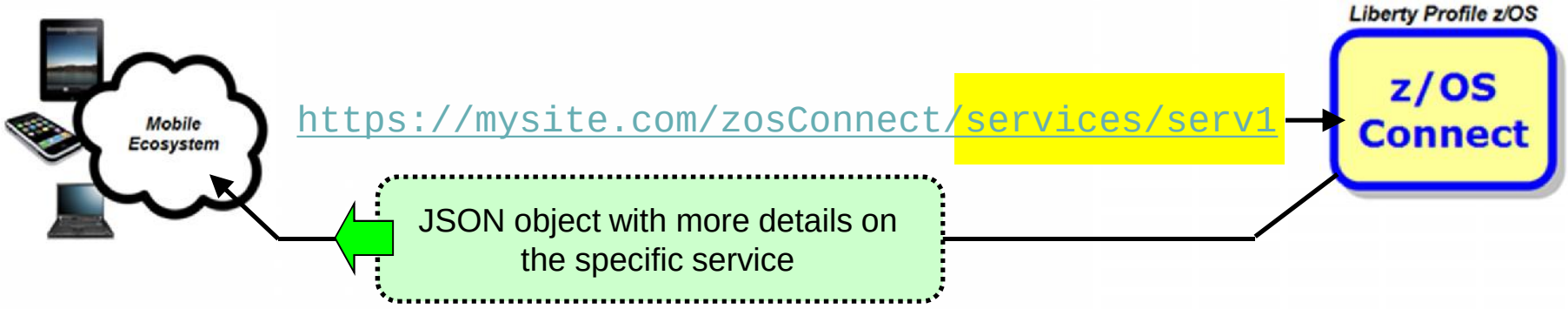
- For z/OS Connect to understand what URIs to handle and how to handle them, you need to configure that information into `server.xml`
  More information on server.xml configuration coming up
- The sample above is a simplified representation of the configuration for multiple services
- z/OS Connect understands 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

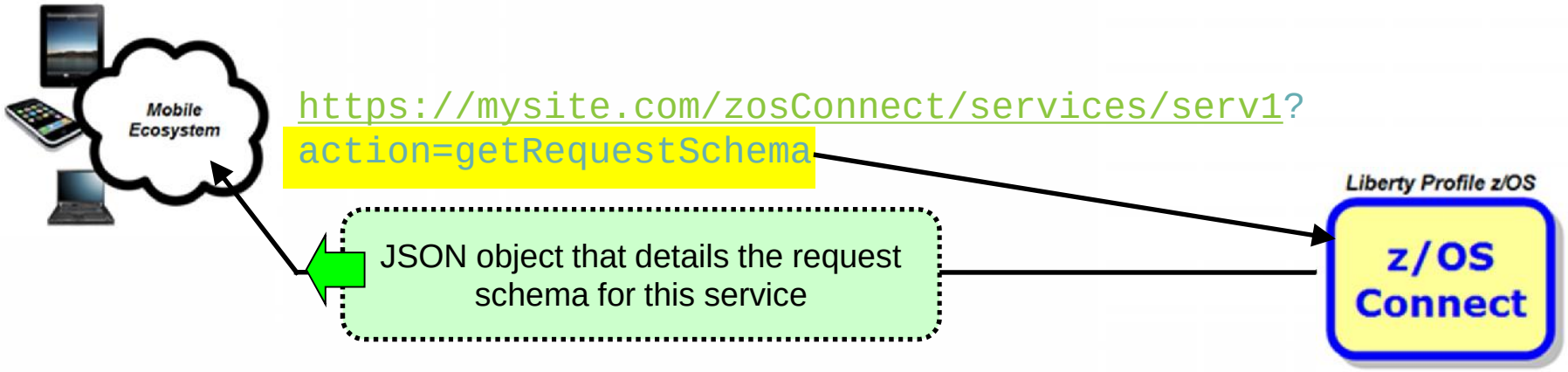
Query for details on a given configured service



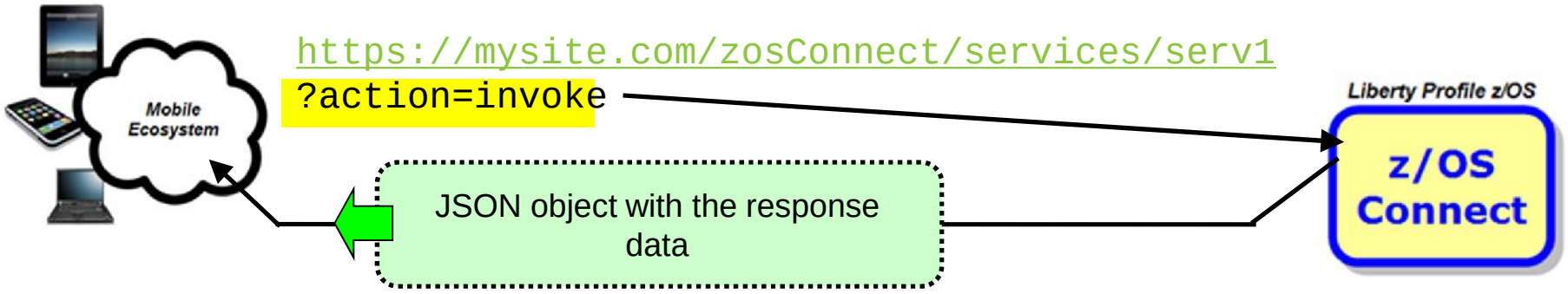
z/OS Connect for Service Invocation

Once you have discovered a service that you would like to invoke, you can use z/OS Connect to get the request schema and then invoke the service:

Get the request schema

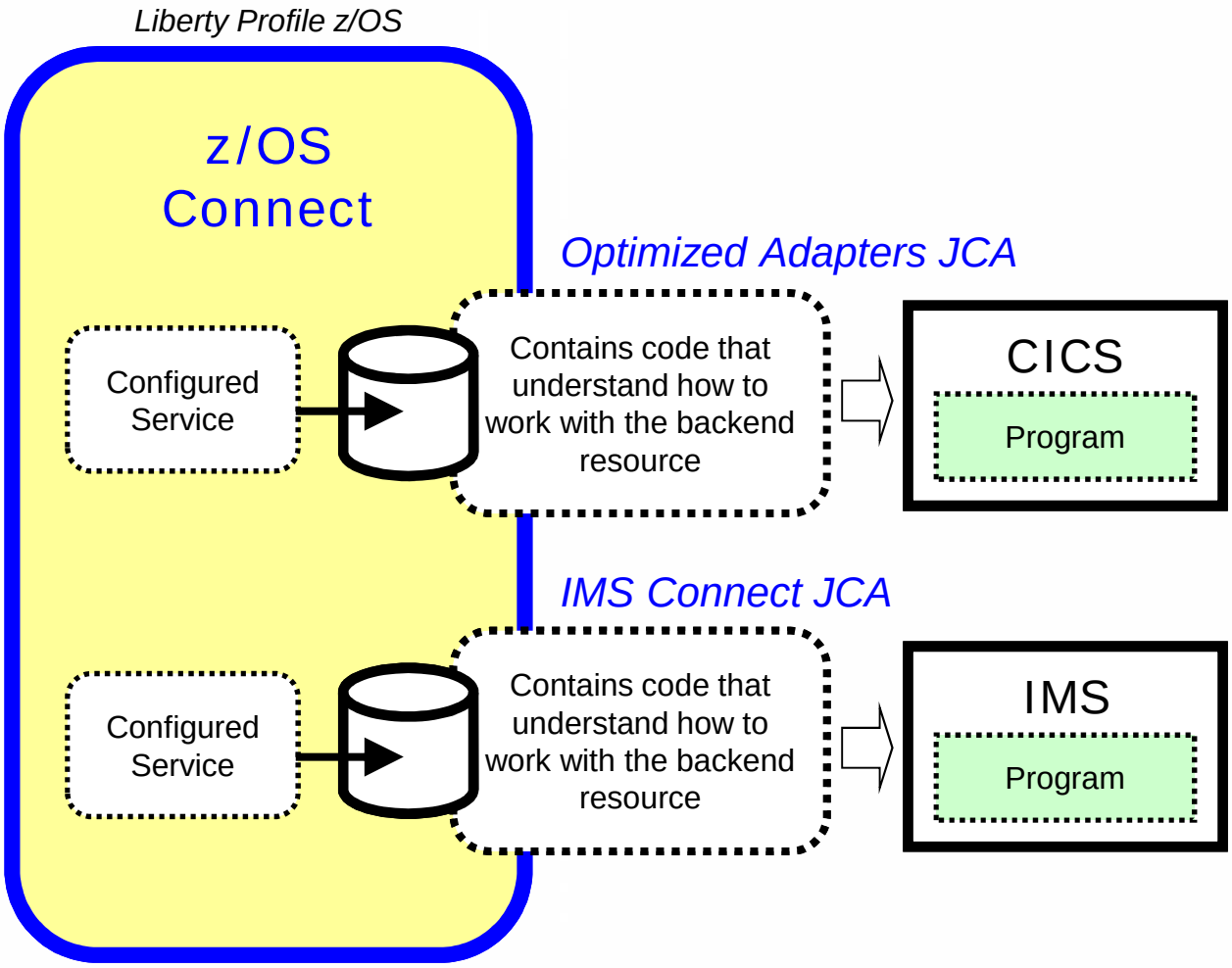


Invoke the service



z/OS Connect 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

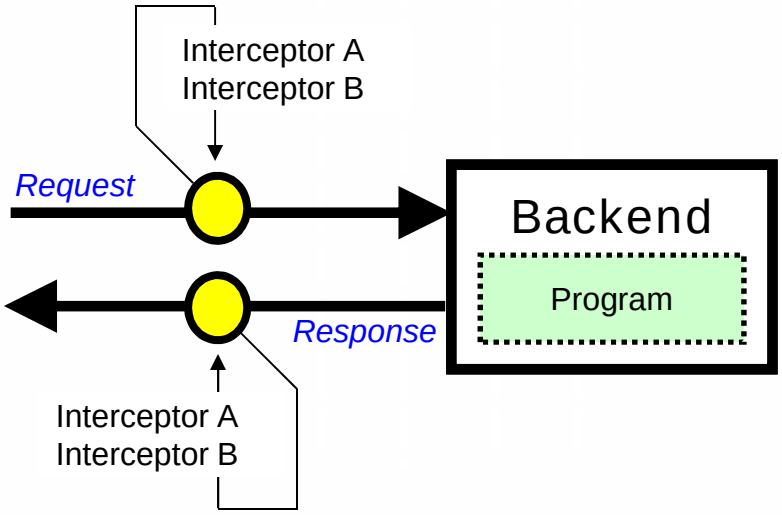
A ‘service provider’ definition provides that information to z/OS Connect

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

Multiple service providers are supported, as illustrated here

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:

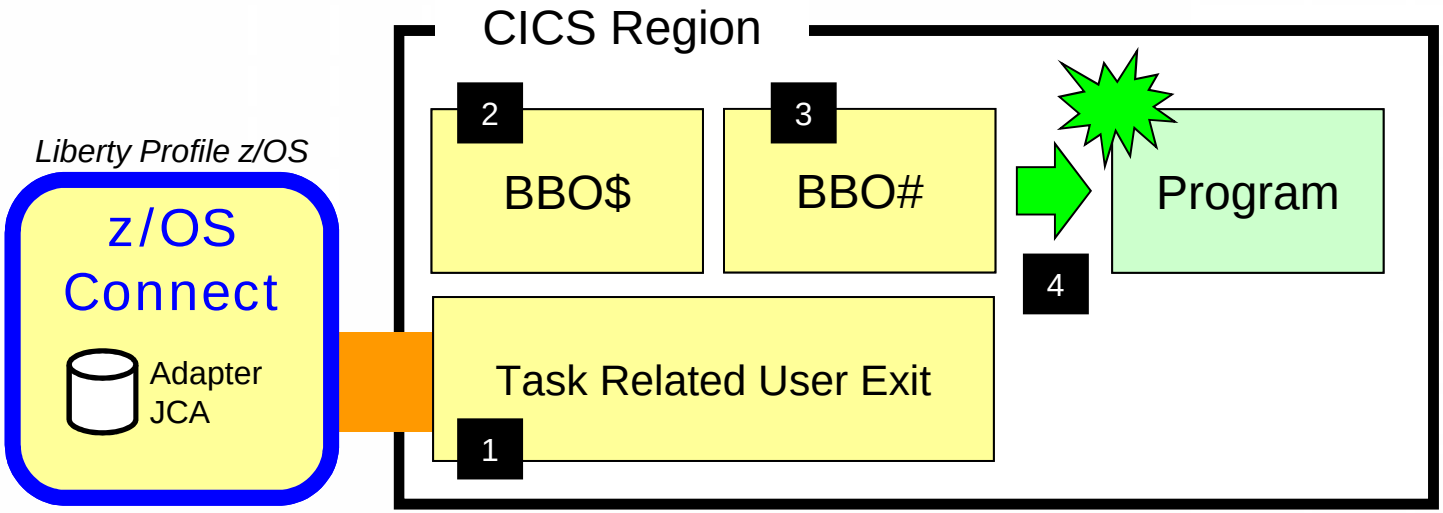
- Define 'global interceptors,' which apply to all configured services
- Define interceptors specific to a given configured service
- Have services 'opt out' of global interceptors

z/OS Connect comes 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

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.

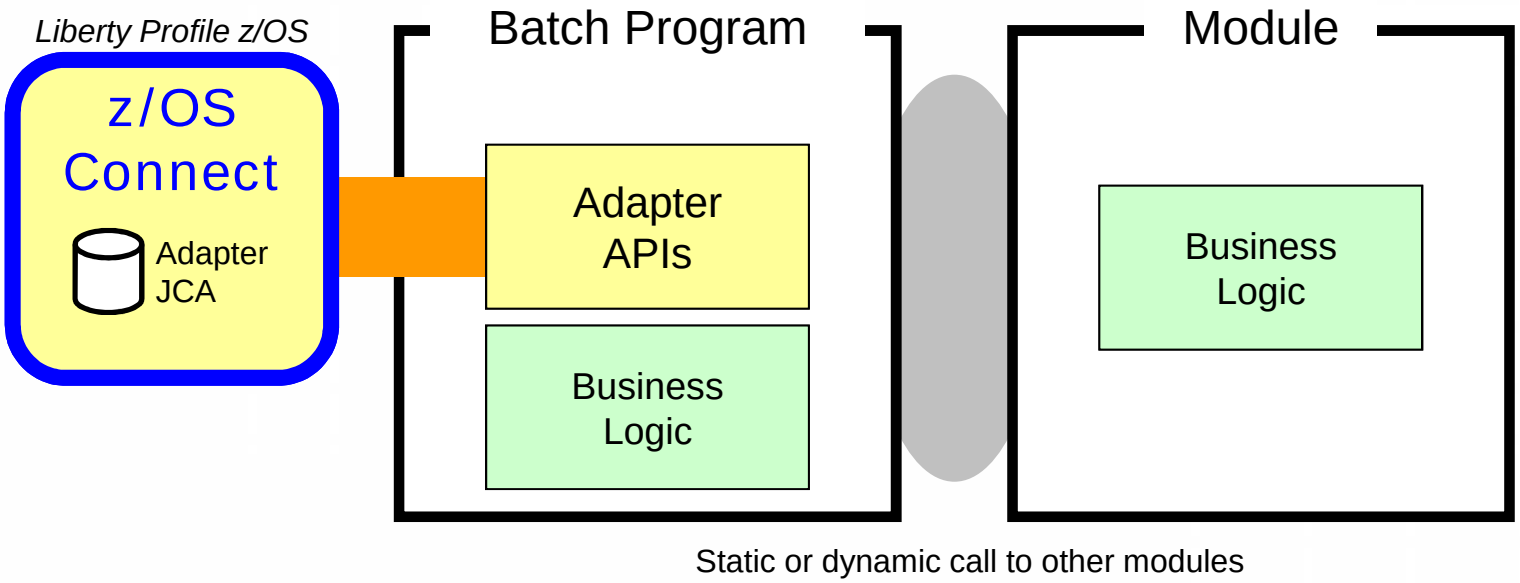
2. The BBO$ long running task handles the calls coming from z/OS Connect. The BBO$/BBO# combination shields the target program from any awareness of Optimized Adapters.

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:



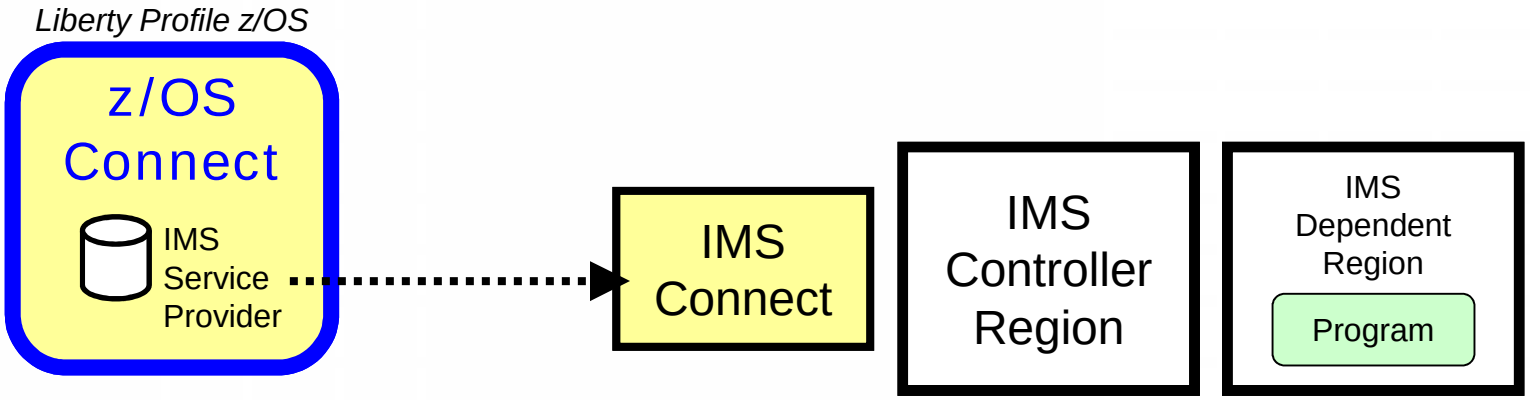
When the target of the z/OS Connect call is a Batch Program, that program will need to use WOLA APIs to “Host a Service”

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:



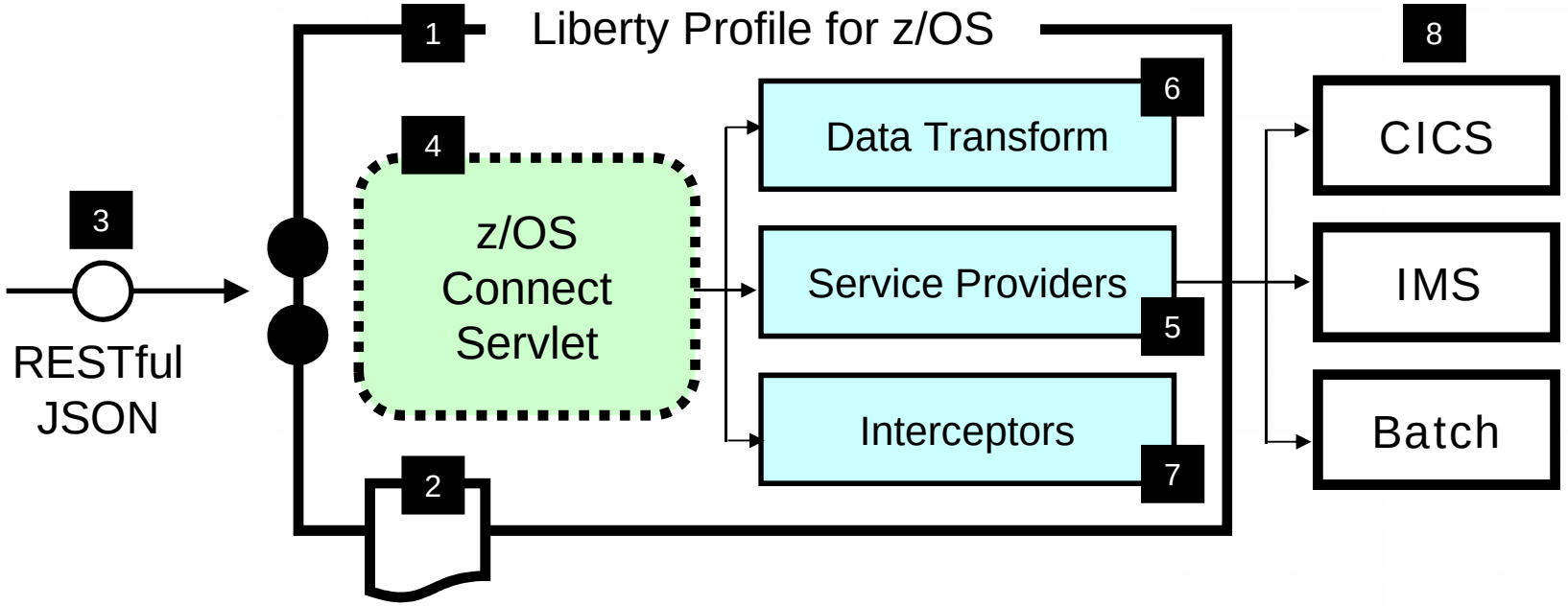
From a z/OS Connect perspective the concepts are the same as what we've shown so far. The syntax for the service provider would be different (IMS Connect vs. Optimized Adapters). And the pointer to the JCA resource adapter would specify the IMS JCA.

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
```

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

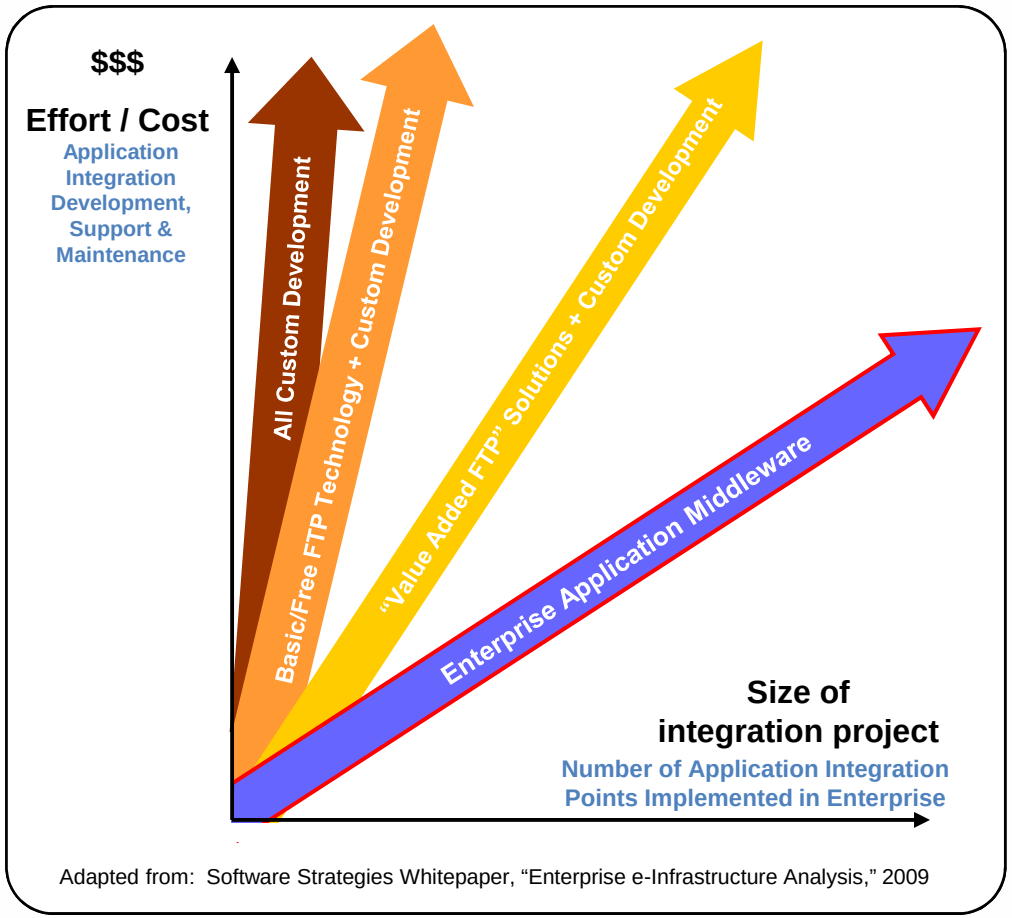
6 z/OS Connect provides the ability to transform JSON to the layout required by backend

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

Messaging and z

IBM messaging middleware **saves you 2-4 times** compared with the cost of rolling-your-own or using FTP



Adapted from: Software Strategies Whitepaper, "Enterprise e-Infrastructure Analysis," 2009

"Our staggering and somewhat shocking research finding is that custom-built, in-house, hard-coded integration solutions (the majority using free FTP software) are much the most widely-used approach.

These often take **2 to 4 times the time and effort** to build as Enterprise Integration middleware-supported integration projects, require a similar multiple of ongoing maintenance and support effort, and are **insecure, fragile and vulnerable** to several serious risks.

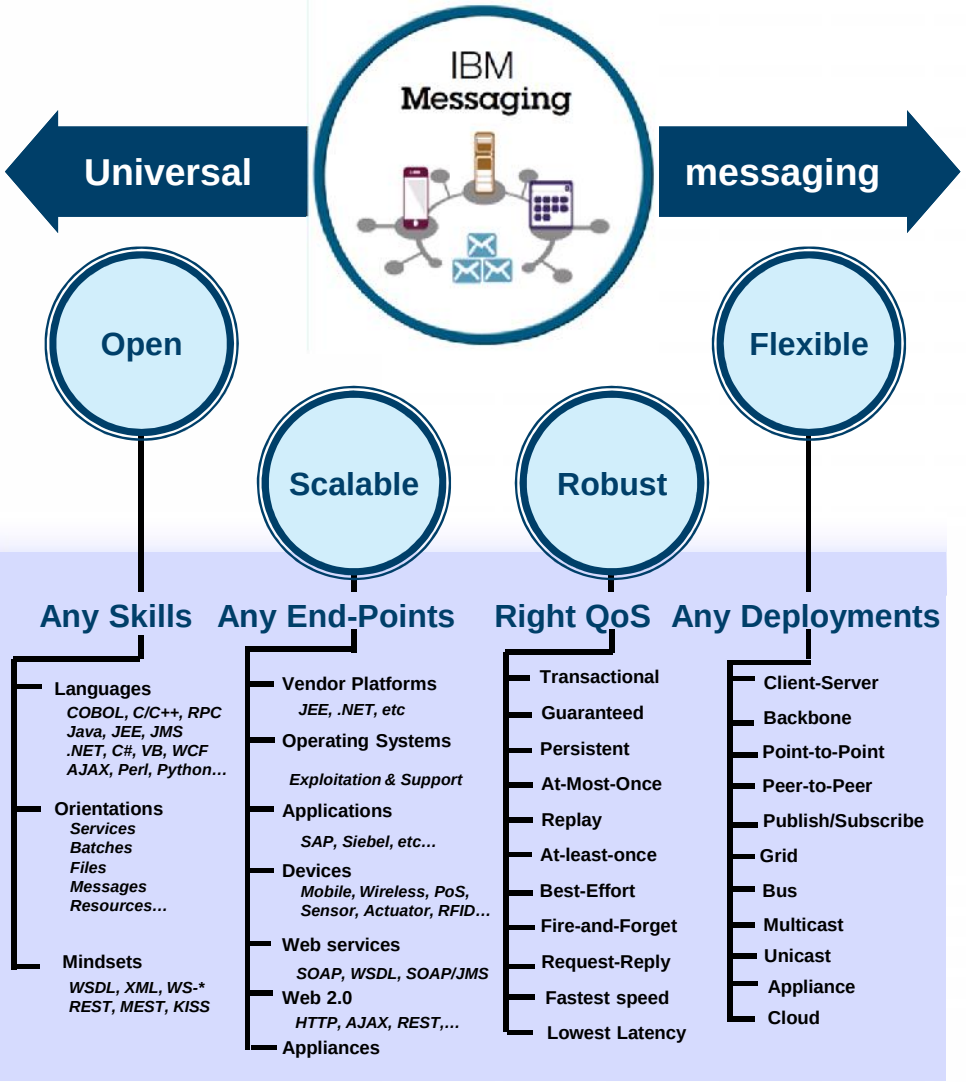
… IBM application integration costs 2-4 times less than custom-built integration approaches . . . **the more applications you integrate the more you save**."

Software Strategies
"Enterprise e-Infrastructure Analysis" 2009

About IBM MQ

IBM MQ:

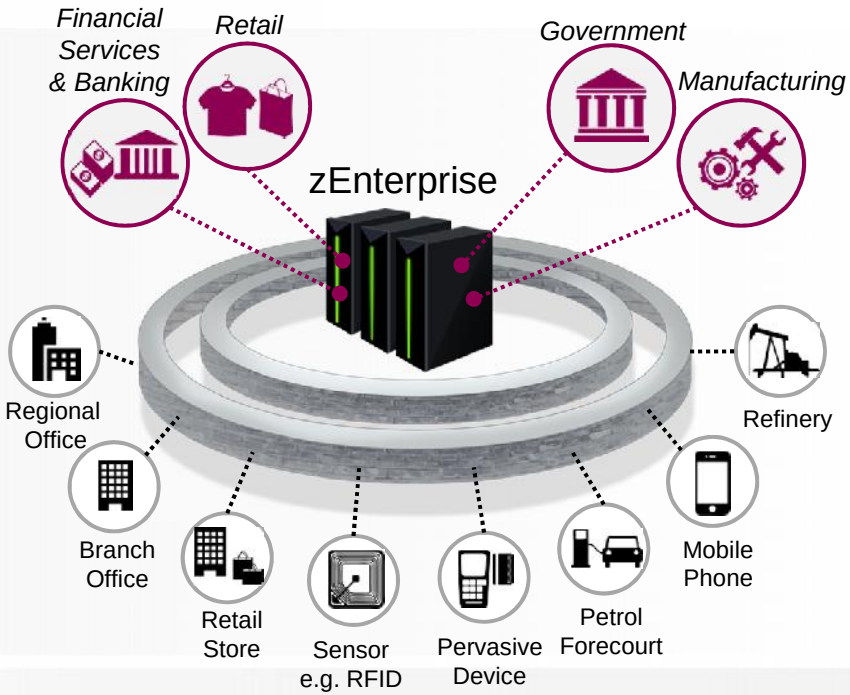
- Provides **versatile messaging integration**, from mainframe to mobile, in a single robust messaging backbone.
- Connects **virtually any commercial IT system**, with support for more than 80 platforms.
- Shields application developers from networking complexities, enabling them to develop and **deploy new applications faster**.
- Includes **administrative features** that simplify messaging management and reduce time spent using complex tools.
- Offers a range of **Qualities of Service** (QoS).

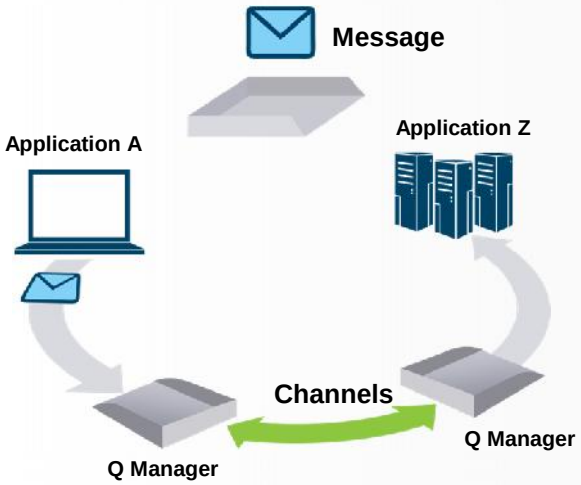


What does IBM MQ do?

Provides messaging services to applications and Web services that need to exchange data and events with:

- Inherent reliable delivery and transaction control
- Native, high-speed handling of any type of message or file
- Native lightweight capabilities for supporting remote devices & sensors
- End-to-end advanced security
- Single point of control, visibility, and management for all data movement





Universally supported by multiple platforms 20 years leading in transactional message delivery

- Applications become more flexible and data movement becomes more reliable
- Capabilities like the Coupling Facility in z Systems provide unique strengths
- Extensive support through years of development, skills and partner ecosystem extensions
- Comprehensive single solution reduces complexity of deployment and operation

Scalable, secure, comprehensive connectivity using reliable messaging everywhere you need it

- **Simplify your applications, increased resilience for your business**
- **Rapid, reliable, simple, secure exchange of data across applications, systems and services**
- **Connect your applications, share information held in files, streamline your processes**
- **Scale to meet peaks in transaction workloads and provide the QoS your business demands**

What's new in IBM MQ V8.0

- ✓ ***Grow your deployment faster, with larger workloads, easier administration***
- ✓ ***Enhanced security with simpler, more powerful operational support***
- ✓ ***New capabilities, additional platform coverage, support for new standards***
  - ✓ *IBM MQ AMS available on IBM i for the first time*

What's new in IBM MQ for z/OS v8.0

- ✓ ***Exploit z Systems capabilities to improve manageability &performance***
- ✓ ***Improved throughput, scalability and operational tasks***
- ✓ ***Enhanced security and connectivity, with support for new standards***
  - ✓ *No Charge for connecting MQ clients from other platforms to MQ for z/OS*

Looking at new enterprise workload?

- Do you need to:
  - Support a new business opportunity with a new application?
  - Build a new application to streamline existing business operations?
  - Better connect existing applications across your enterprise
- Are you:
  - Looking to deploy on z for mission critical reliability and enterprise class scaling?
  - Looking to keep operational costs/expense low?

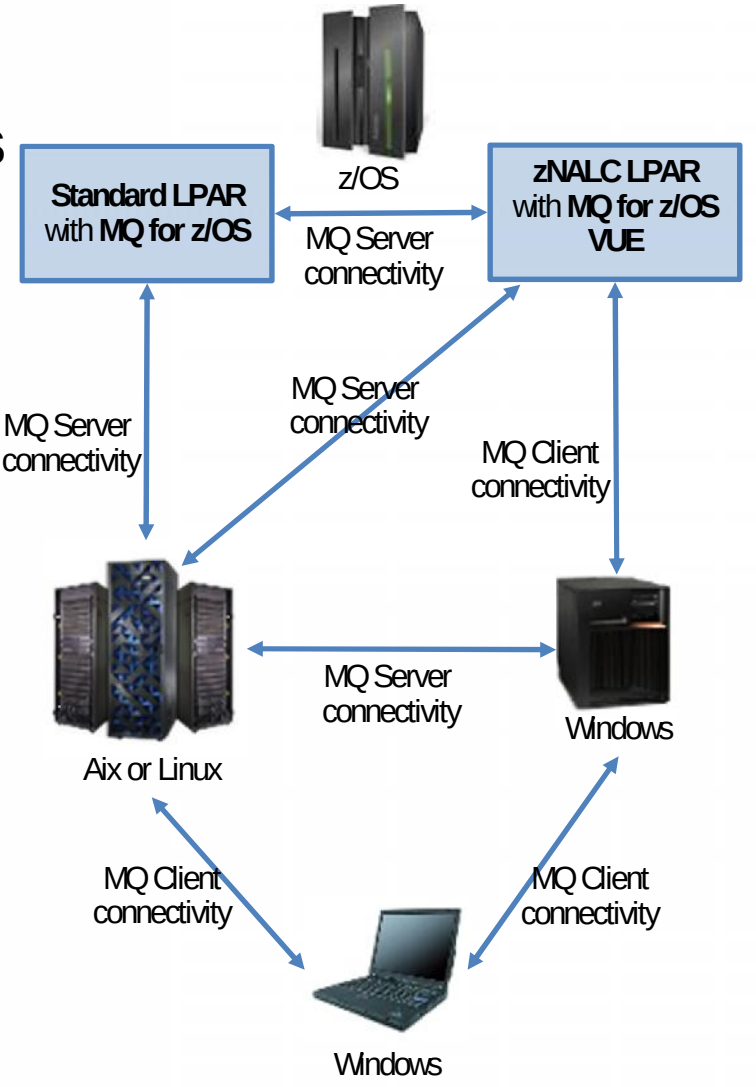
IBM MQ for z/OS Value Unit Edition

- All the power of the **premier, unmatched messaging product on z/OS** sold under an OTC license. The **same product and capabilities** as MQ for z/OS
  - Easy to connect distributed workloads and data directly to applications on z/OS using MQ
- Connectivity for your new workload running on your zNALC partition
  - Connect to your applications running on the same zNALC LPAR,
  - Connect to another z/OS LPAR using MQ
  - Connect to MQ on other platforms
- Can federate with other instances of MQ for z/OS
- Can be managed by the same System Management tool as other instances of MQ for z/OS
- The affordable way to connect your zNALC workloads
  - Has been sold to non-zNALC LPARs where no existing MQ for z/OS (exception needed)
- Can be extended with IBM MQ Advanced for z/OS
  - Adds IBM MQ Managed File Transfer and IBM MQ Advanced Message Security

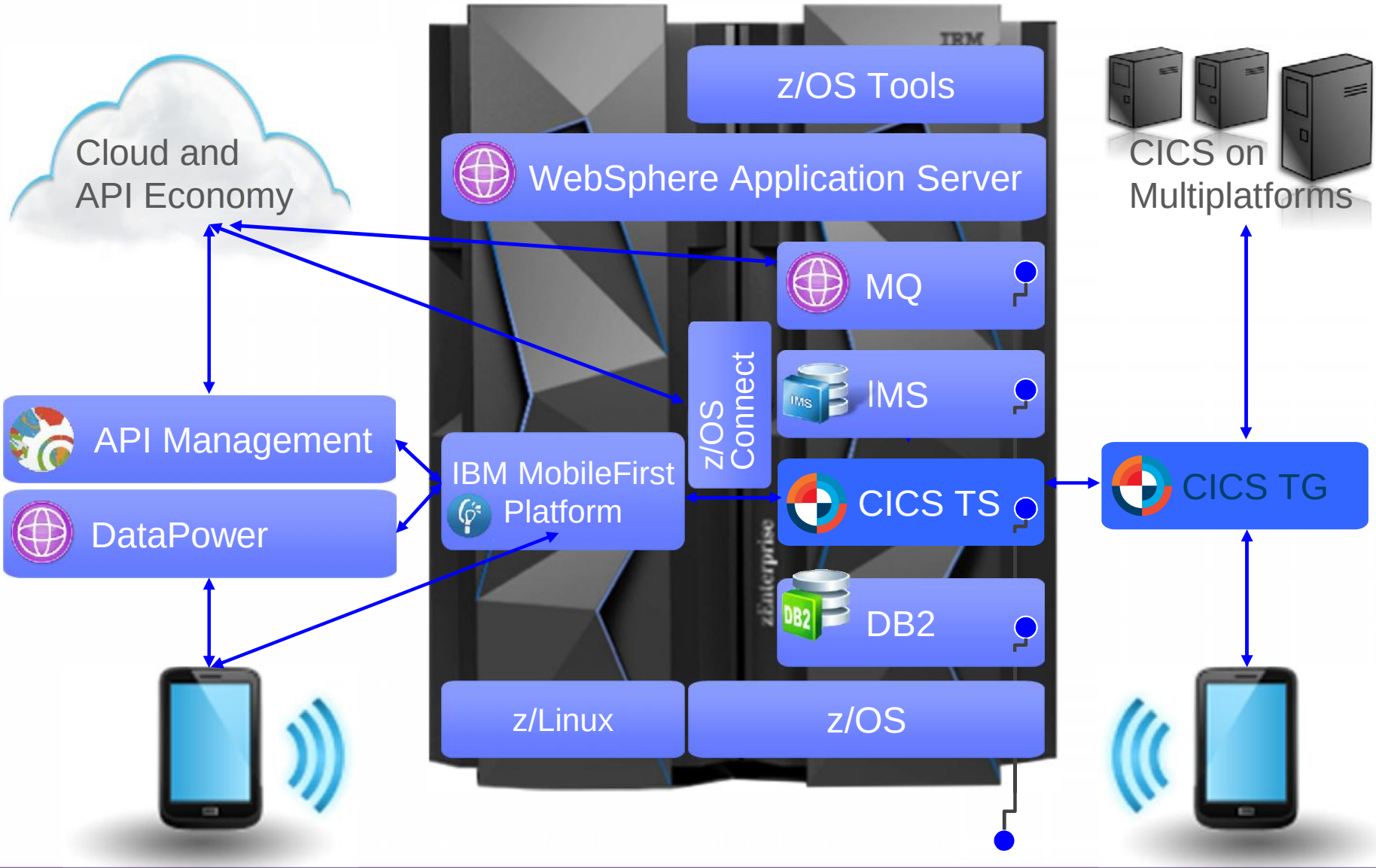
SCENARIO

Customer has MQ on distributed and MQ for z/OS

- Already have a zNALC LPAR?
  - Could add new MQ for z/OS VUE to the zNALC LPAR without increasing Operational Expense but gain easy connectivity to the rest of the enterprise.
- Don't currently have a zNALC LPAR…
  - If one is added then adding MQ VUE there will ease connectivity to the rest of the enterprise.
- In either situation benefits can be seen from not just MQ VUE but if MQ Advanced for z/OS is added it will provide MQ MFT and MQ AMS.
- If customer doesn't have MQ for z/OS then MQ VUE is still an option as an exception for LPARs where no MQ on z/OS today



Systems of Engagement meet Systems of Record

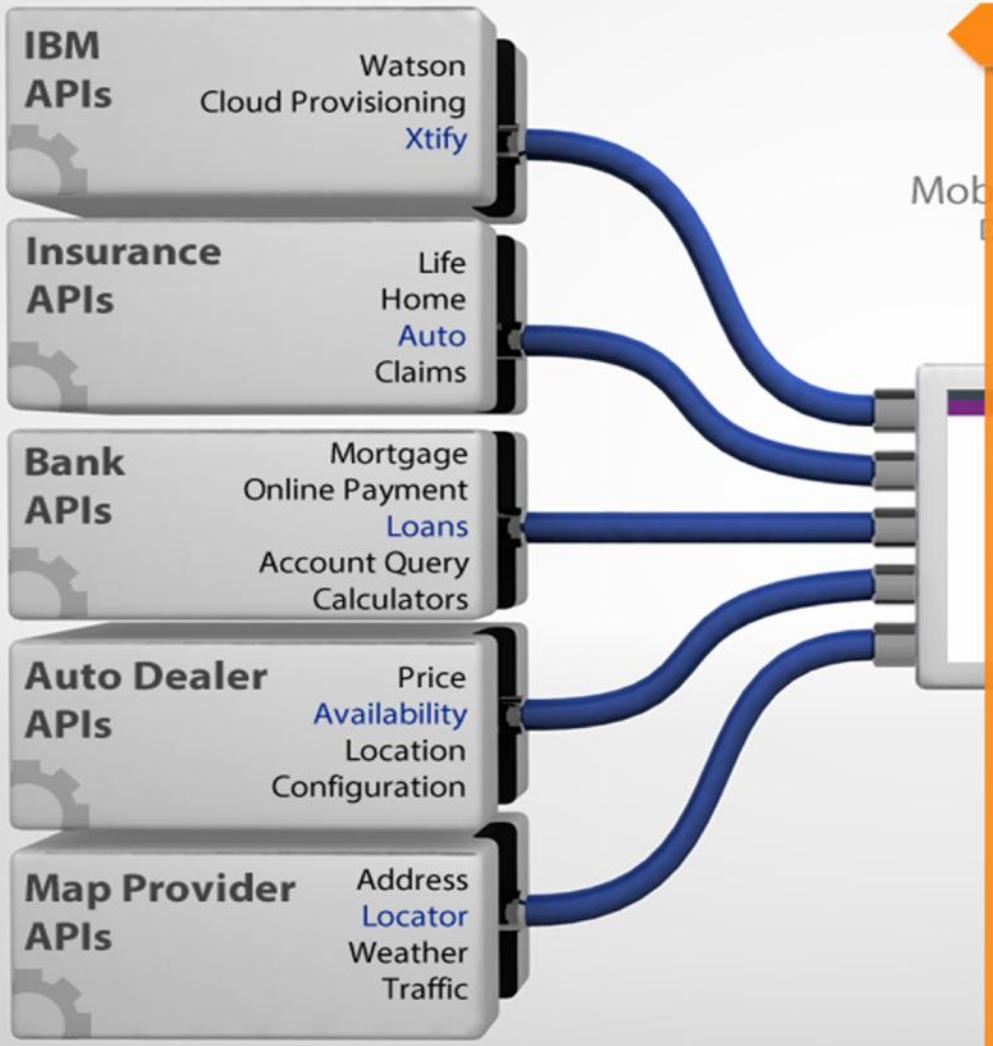


Thank You

Backup Details

API Economy

Mobile App Assembly



Providers

Consumers

Benefits for Application Assemblers

- Quick time to Market by using existing assets and services to decrease costs and speed up product and service development
- Driving innovation by capitalizing on the merging of capabilities provided by different APIs
- Attracting clients to your products and services by joining other (established) API ecosystems
- Potentially having freedom of choice in terms of available APIs (avoiding vendor lock-in)
- Focus on core competence and value add rather than reinventing common capabilities
- Create new business opportunity from APIs
- Innovate around business models and reach new customers from the provider's ecosystem

API Economy

Provider Perspective

Mobile App Assemblers

Developers & Partners

IBM APIs
- Watson
- Cloud Provisioning
- Xtify

Insurance APIs
- Life
- Home
- Auto
- Claims

Bank APIs
- Mortgage
- Online Payment
- Loans
- Account Query

Auto Dealer APIs
- Price
- Availability
- Location
- Configuration

Map Provider APIs
- Address
- Locator
- Weather
- Traffic

New Customers

Providers | **Consumers**

API Economy

Provider Perspective

Providers

Benefits

- Expanding into new customer bases and niches that you normally wouldn't be able to reach on your own
- Focusing on your core value
- Expanding your brand and brand loyalty
- Easier possibilities in establishing new partnerships and capitalizing on them
- Benefitting from open innovations of crowdsourcing and expert sourcing, which are implicitly unlocked through the opening of APIs
- Realize profit from new business models
- Decreased development costs and time through software (e.g., mobile apps) produced by third parties
- Keeping up with application demands (e.g. new apps, new features)
- Wider and quicker coverage of different platforms and devices

Mobile App Assemblers

Developers & Partners

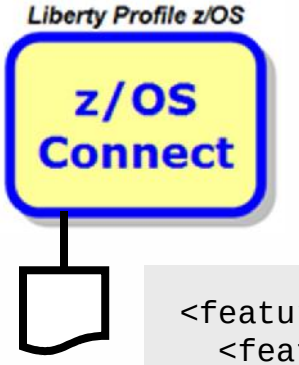
New Customers

Consumers

45

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>
```

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.

```
<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 the service provider – WOLA in this case – that provides the connectivity to the backend CICS region

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
```

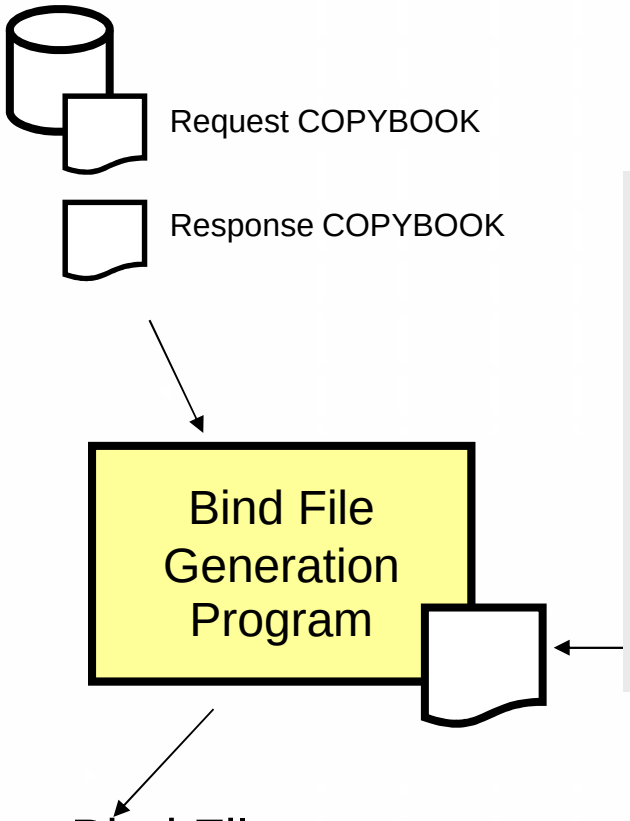
```
{
    "firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": “1234 Main Street",
        "city": “Anytown",
        "state": "NY",
        "postalCode": "10021-1234"
    },
}
```

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

JSON can be passed back to the client as well.

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

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
```

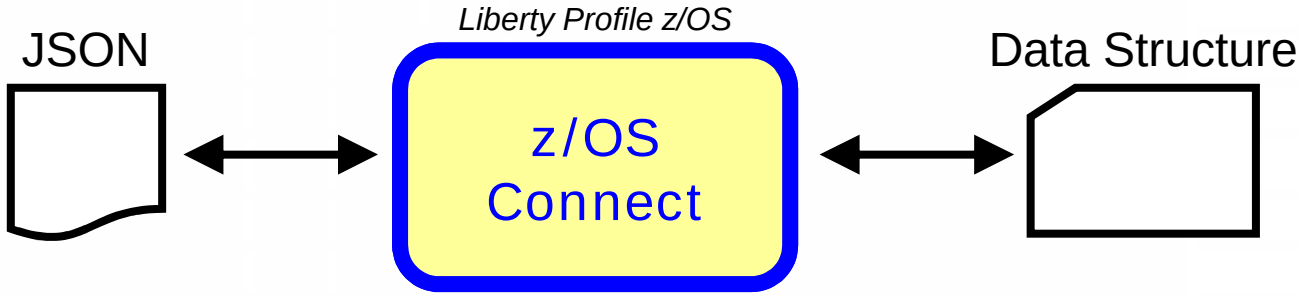
```
<zosConnectService id="zcs3" serviceName="ContactCreate"
     serviceDescription=“Create"
     serviceRef="wolaKIXPHONE"
     dataXformRef="xformJSON2Byte">
</zosConnectService>

<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

Data Transformation

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

Audit (SMF) Interceptor

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

Liberty Profile z/OS

z/OS Connect

- 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