CICS, REST, JSON, etc.

IBM Advanced Technology Group

Steve Fowlkes – fowlkes@us.ibm.com

Leigh Compton – lcompton@us.ibm.com

Abstract

- This presentation provides an overview of how CICS applications can integrate into your environment using REST and JSON technologies.

Agenda

- REST
  - What is it used for
  - Its origin
  - Details
    - HTTP, the flows
- JSON
  - Origin
  - Layout and specification
- Native CICS JSON Support
- z/OS Connect

Who are we communicating with?

- Mobile devices
  - REST (JSON or XML)(XML not covered in this presentation)
  - Web service (SOAP/XML) (not covered in this presentation)
  - ATOM feeds (XML) (ATOM not covered in this presentation)
- Web browser
  - REST (JSON or XML) (XML not covered in this presentation)
  - ATOM feeds (XML) (ATOM not covered in this presentation)
- Application to Application
  - Web service (SOAP/XML) (WS not covered in this presentation)
  - Sometimes- REST (missing out on all those WS-specifications !)

REST Services

REST

- It is an architectural style
- Invented 1994, Roy Fielding, documented in his year 2000 doctoral thesis
- Data sent/received is whatever you like
  - JSON and XML most often used
- Standards
  - Leverages the HTTP protocol
- Use
  - Program to JavaScript in browser
  - Program to mobile device
  - Application-to-Application
    - aka “RESTful Services”
- Supplies data to:
  - Designed for a Web browser
  - Today: program-to-program, where web services is deemed too heavy

REST

- No official standard for REST web APIs:
  - REST is an architectural style (unlike SOAP which is a protocol)
  - Uses Web standards like HTTP, URI, XML, JSON, etc
- Attributes (generally agree on architectural constraints)
  - Client-Server
  - Stateless
  - Cacheable
  - Layered system
  - Uniform Interface
- Components
  - URI – e.g. http://example.com/resources
  - Internet media type – JSON, XML, Atom, images
  - HTTP methods: GET, PUT, POST, DELETE

REST Simple Sample

- Request

```
GET /mortgage/231677 HTTP/1.1
 Host: www.example.com
 Accept-Language: en
 Charset: UTF-8
```

- Response

or

```
HTTP/1.1 200 OK
 Language: en_us
 Charset: UTF-8
 Content-Type: application/json
{“principal”:”238000”,”rate”:”3.5”, “type”:”5/1 ARM”}
```

```
HTTP/1.1 200 OK
 Language: en_us
 Charset: UTF-8
 Content-Type: text/xml
<mortgage><principal>238000</principal><rate>3.5</rate><type>5/1 ARM</type></mortgage>
```

The ways to do REST to/from CICS - summary

- CICS provides support for JSON web services (inbound only)
- Use the EXEC CICS WEB API commands (from CICS TS V2.2+) (in and out)
- From a Servlet or JAX-RS in the Liberty Profile (in and out)
- ATOM support in CICS (XML Only – inbound only)
- CICS TG V9.1+ (inbound only)
- Node.js (inbound only)
- z/OS Connect (in and out)
  - Has discovery capabilities
  - WAS Liberty outside CICS ➔ IPIC ➔ CICS program

JSON

JSON (JavaScript Object Notation)

- JavaScript Object Notation
  - Data in attribute-value pairs
  - Used primarily to transmit data between a server and web browser
  - Alternative to XML
- Originally based on JavaScript –
  - Most commonly used in web browsers
  - It is a dynamic computer programming language
- JSON MIME type
  - Official: “application/json”
  - Unofficial: “text/json” or “text/javascript”

JSON (JavaScript Object Notation)…

- Attribute-value pairs (example of JSON)

```
{
   "id": 1,
   "name": "Foo",
   "price": 123,
   "tags": [ "Bar", "Eek" ],
   "stock": {
        "warehouse": 300,
        "retail": 20
      }
 }
```

- Info about JSON schema: `http://json-schema.org/`
- Info about JSON schema core definitions and terminology: `http://tools.ietf.org/html/draft-zyp-json-schema-04`
- Info about JSON schema interactive and non-interactive validation: `http://tools.ieft.org/html/draft-fge-json-schema-validation-00`

CICS native JSON support

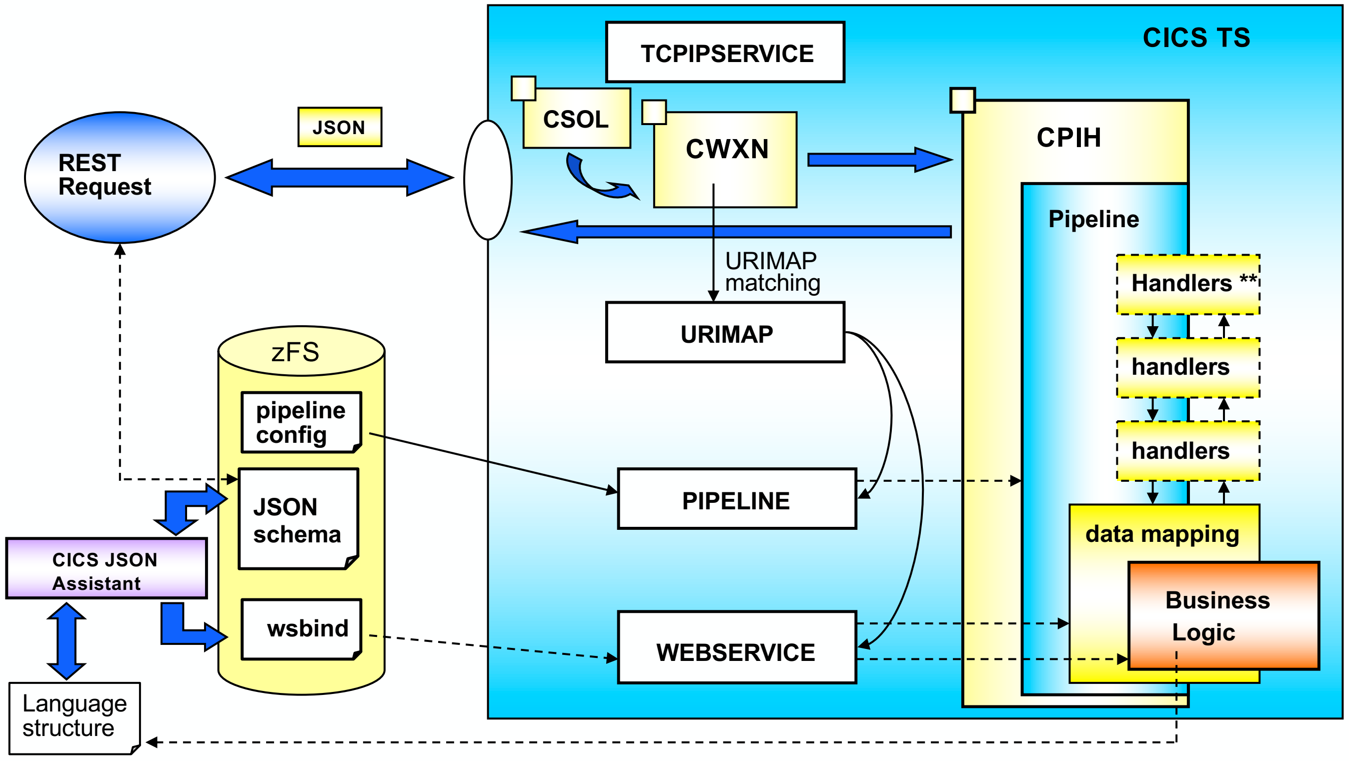
CICS TS native JSON support

- Can put a JSON-RPC interface on an existing CICS program
  - Call or request-response interaction
  - Bottom-up, copybook as input, only honors the POST request
- Can have full REST semantics
  - Top-down, JSON schema as input
  - Wrapper program that uses channel/containers, and understands GET,PUT,POST,DELETE
  - Instead of defining the schema yourself:
    - Use the bottom-up approach to create schema from copybook
    - Modify the schema if needed
    - Use the top-down approach to create copybooks to be used in wrapper program
- Can programmatically invoke JSON transformation
  - LINK to JSON transformation routine
  - EXEC CICS TRANSFORM command

Details of native JSON support

- CICS implements support for standard JSON schema specification
  - Specification at http://json-schema.org
  - Draft 4 of the emerging specification
  - CICS supplies a partial implementation
- CICS provides utilities to generate binding files for transformation
  - DFHLS2JS can be used to generate a schema
  - DFHJS2LS to consume one and generate copybook(s)
- Can use latest levels of IDz (formerly RDz) to develop JSON binding files
- MQ transport not supported with JSON pipeline
- Full list of CICS restrictions is at:
  - https://www.ibm.com/docs/en/cics-ts/5.6?topic=services-json-web-service-restrictions

CICS JSON Pipeline



** Handler are not normally used

CICS JSON Transformer

- Transform data in language structure to and from JSON
- Run DFHLS2JS or DFHJS2LS
  - Output is Bundle containing bindings file and JSONTRANSFRM resource
- DFHJSON – the Linkable interface
  - LINK to DFHJSON passing data in Containers
- EXEC CICS TRANSFORM
  - JSONTODATA
  - DATATOJSON
  - Data passed in Containers

Redbook: Implementing IBM CICS JSON Web Services for Mobile Applications

Table of Contents:

IBM

Implementing IBM CICS JSON Web Services for Mobile Applications

Redbooks

http://www.redbooks.ibm.com/abstracts/sg248161.html

JSON support in CICS TG

REST in Java

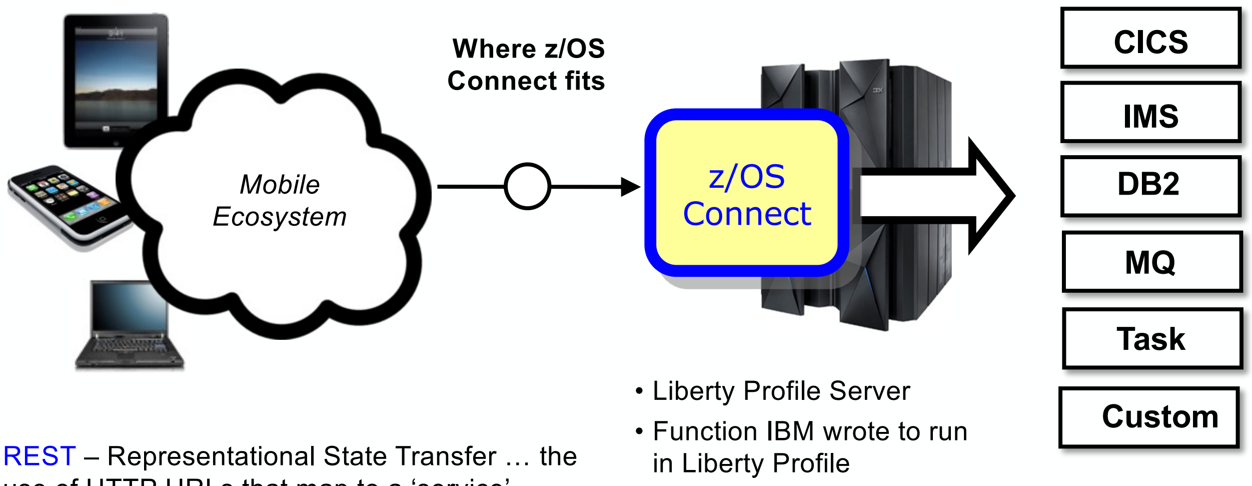
REST/JSON in CICS using Java Liberty Server

- Liberty Profile Server executes within CICS
  - Dynamic profile of WAS
    - Provision only those Java features needed for application
  - Initially delivered web application server capabilities to CICS
- JAX-RS
  - Java API for RESTful Web Services
  - Simplifies development and deployment of service clients and endpoints
- JSON4J
  - Set of classes for handling JSON data
  - Simple model for transforming JSON data to other representations
- JAX-RS and JSON4J are well accepted in the Java community

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:



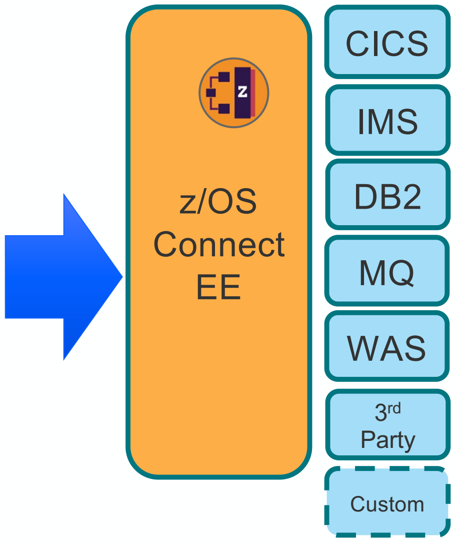
- Liberty Profile Server
- Function IBM wrote to run in Liberty Profile

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

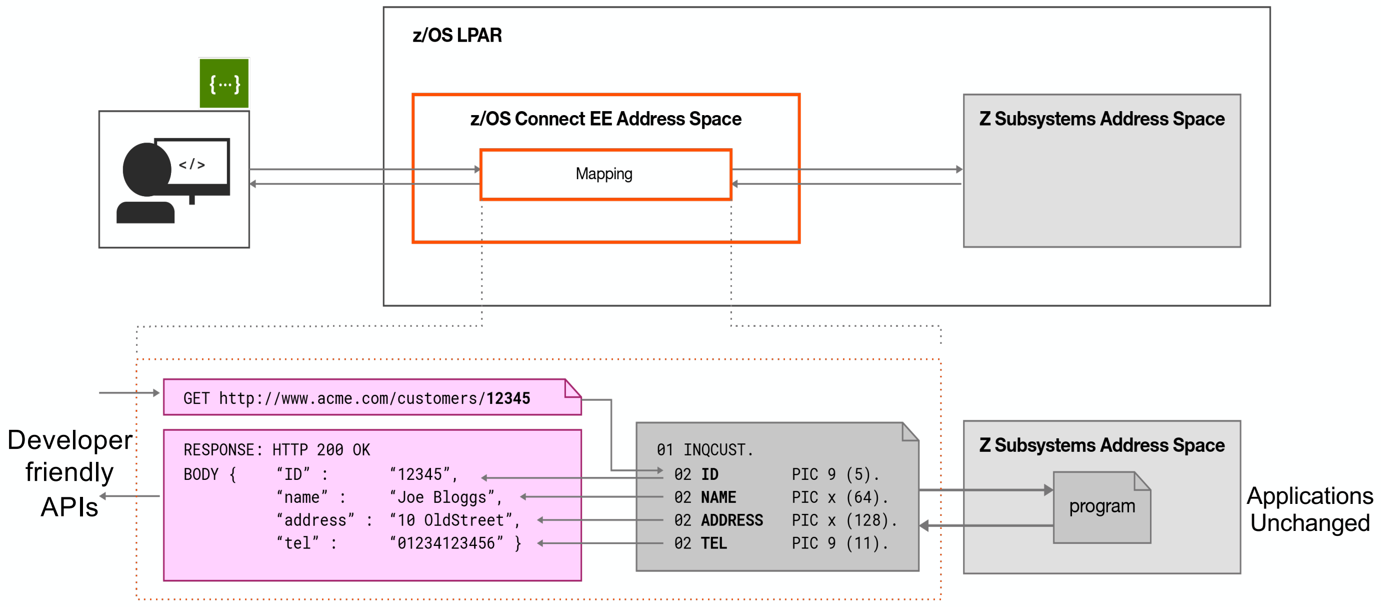
JSON – JavaScript Object Notation … a standard of representing data as a set of name/value pairs. This is passed back and forth along with REST request/responses

z/OS Connect EE High Level Points

- Provides a common and consistent entry point for mobile access to one or many backend systems
- "Fully REST" enable major z/OS Subsystems
  - **z/OS applications and data appear as any other REST Provider**
  - **Support of all REST verbs**
- No Backend Application Changes
- No Coding Required (Tool Driven)
- Support of Open Standard Open API Doc (aka Swagger 2.0) for Integration with other products that support the standard
- Provide agile (dynamic) API creation, simple testing and easy deployment
- Enable "division of duties" between z/OS team and enterprise API team for fast time to deploy



Data mapping - A closer look



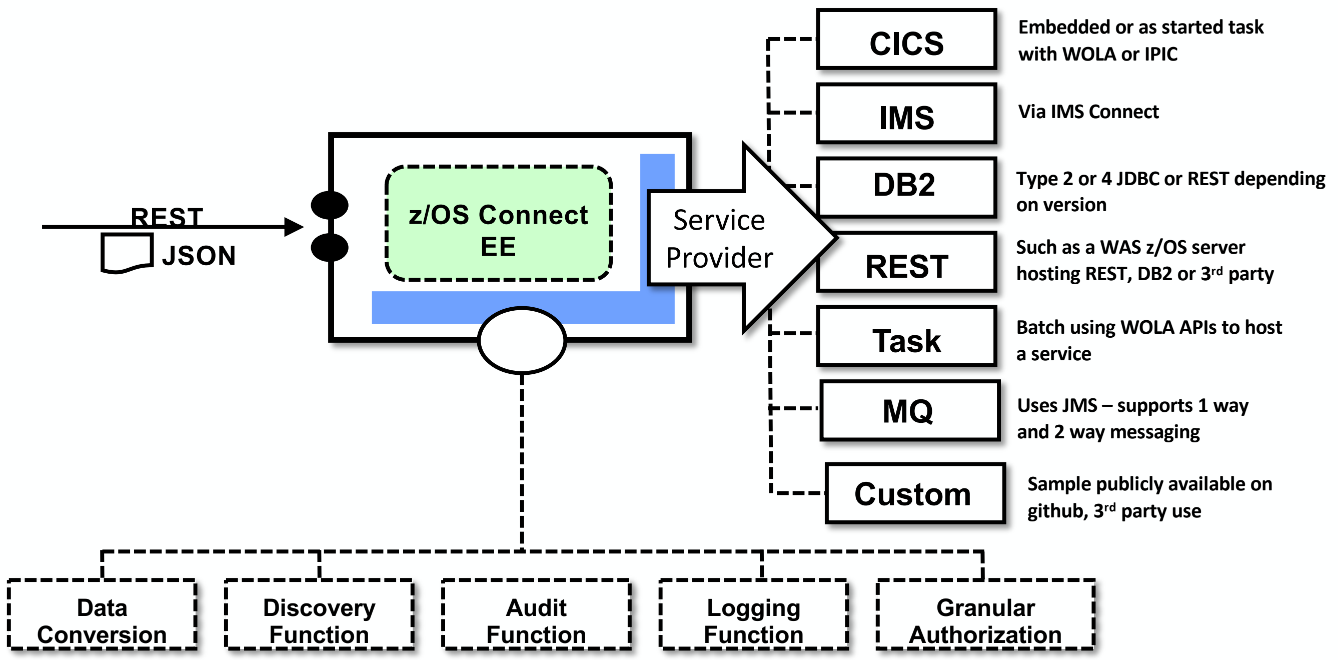
z/OS LPAR
{…}
</>
z/OS Connect EE Address Space
Mapping
Z Subsystems Address Space
Developer friendly APIs
Z Subsystems Address Space
program
Applications Unchanged

```
GET http://www.acme.com/customers/12345
```

```
RESPONSE: HTTP 200 OK
BODY {     "ID" :        "12345",
           "name" :      "Joe Bloggs",
           "address" :   "10 OldStreet",
           "tel" :       "01234123456" }
```

```
01 INQCUST.
  02 ID         PIC 9 (5).
  02 NAME       PIC x (64).
  02 ADDRESS    PIC x (128).
  02 TEL        PIC 9 (11).
```

z/OS Connect EE Structure



Embedded or as started task with WOLA or IPIC

Via IMS Connect

Type 2 or 4 JDBC or REST depending on version

Such as a WAS z/OS server hosting REST, DB2 or 3rd party

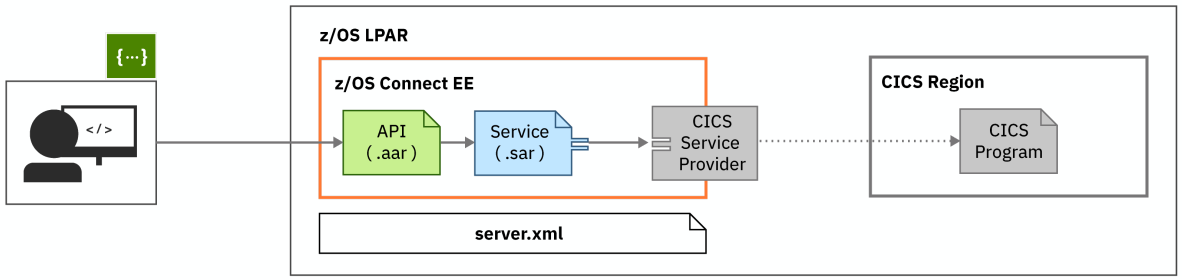
Batch using WOLA APIs to host a service

Uses JMS – supports 1 way and 2 way messaging

Sample publicly available on github, 3rd party use

Connect to CICS

- **API provider:** Liberty z/OS Connect on z/OS
- **Data transform:** WSBind files
- **CICS integration: IPIC**



Connection to CICS is configured in `server.xml.`

An IPIC connection must be configured in CICS.

Transformation of data from JSON to copybook layout and mapping of the REST semantics to the program linkage takes place within the z/OS Connect server and the target CICS programs are invoked via a LINK/DPL, receiving data in a Commarea or Containers.

ibm.biz/zosconnect-scenarios

Which way to go?

REST which way to go ?

- z/OS Connect
  - Fast on-ramp to z/OS applications (IMS, CICS, etc)
    - Common interface
  - Tools to allow developer to define secure REST service without the need to develop extensive code for System z
- JSON programs in CICS
  - REST service in CICS using the CICS API or PIPELINE
  - No automatic client support with JSON, although you can use the CICS WEB API and the DFHJSON program (or EXEC CICS TRANSFORM)
- Java program
  - Lots of Java programmers
  - Java standards
- Skill set
  - Many people use what they know

Summary

- REST
  - What is it used for
  - Its origin
  - Details
    - HTTP, the flows
- JSON
  - Origin
  - Layout and specification
- CICS native JSON support
- z/OS Connect

References for JSON and REST

- Articles and tutorials:
  - https://www.youtube.com/watch?v=Sqtlm1jxpBI&feature=c4-overview-vl&list=PLJxIWNrnCsg-oAben8EgVEF4a9FlRdNBa
  - https://www.youtube.com/watch?v=6TkQ9PzeevQ&list=PLJxIWNrnCsg-oAben8EgVEF4a9FlRdNBa&index=4
  - https://www.youtube.com/watch?v=SqjlXxw2FiY&list=PLJxIWNrnCsg-oAben8EgVEF4a9FlRdNBa&index=1
  - https://www.youtube.com/watch?v=5JyJ0XXR_3c&index=2&list=PLJxIWNrnCsg-oAben8EgVEF4a9FlRdNBa

References for JSON and REST

- Roy Fielding's Doctoral Thesis – for the definition of REST –
  http://en.wikipedia.org/wiki/Roy_Fielding
  http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

- SG24-8161 – Implementing CICS JSON Web Services for Mobile Applications
  http://www.redbooks.ibm.com/abstracts/sg248161.html

- JSON RFC 4627 –
  http://www.ietf.org/rfc/rfc4627.txt

- JSON – the basics –
  http://www.ibm.com/developerworks/webservices/library/ws-restful/