Db2 for z/OS and REST Services

Tori Felt
Db2 for z/OS Specialist – WSC

Keziah Knopp
Db2 for z/OS Specialist – WSC

Mobile Trends & the API Economy

It’s not just a fad

5 mobile trends with significant implications for the enterprise

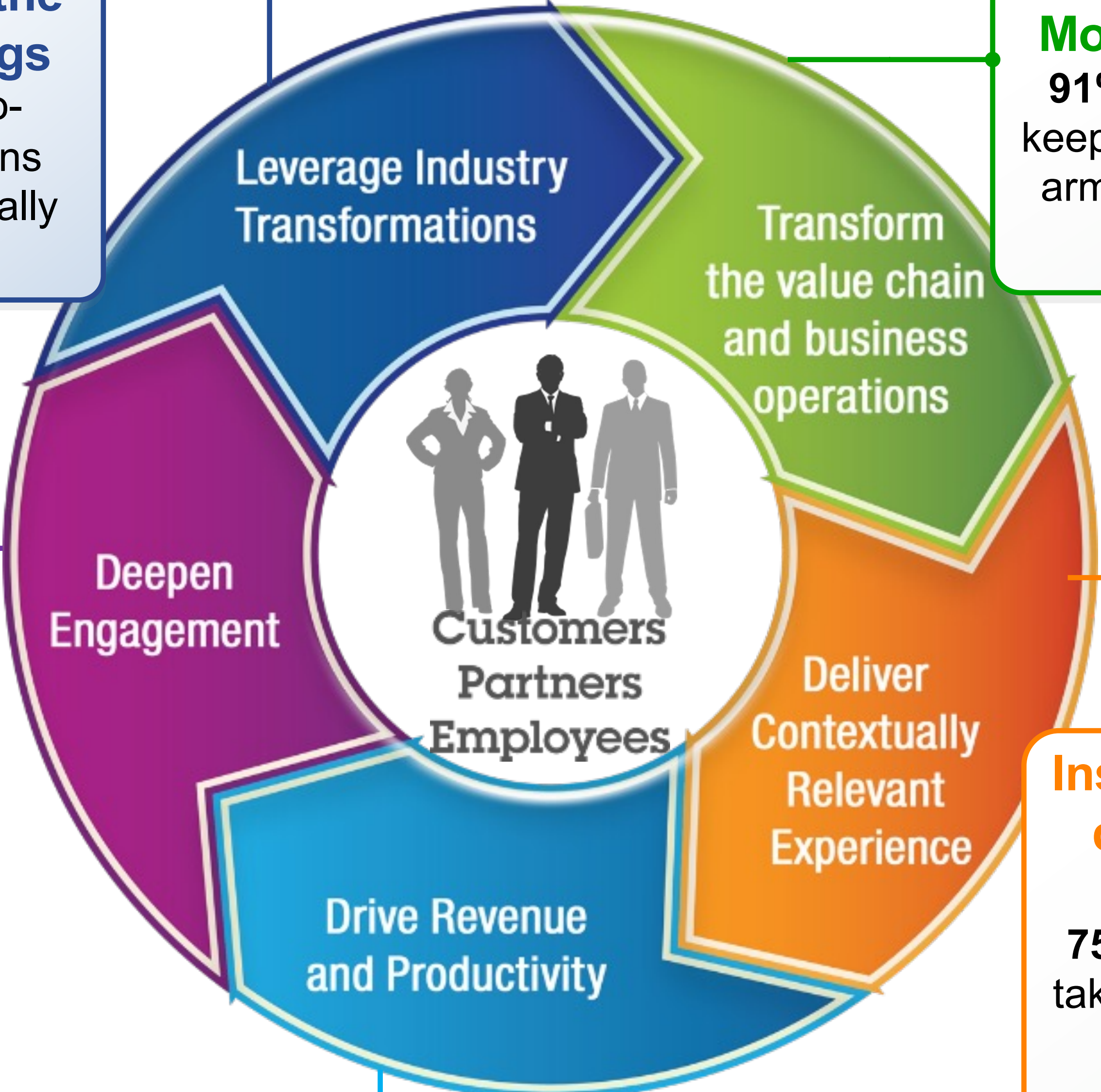
Mobile enables the Internet of Things
Global machine-to-machine connections increasing dramatically

Mobile is primary
91% of mobile users keep their device within arm's reach 100% of the time

Mobile must create a continuous brand experience
90% of users use multiple screens as channels come together to create integrated experiences

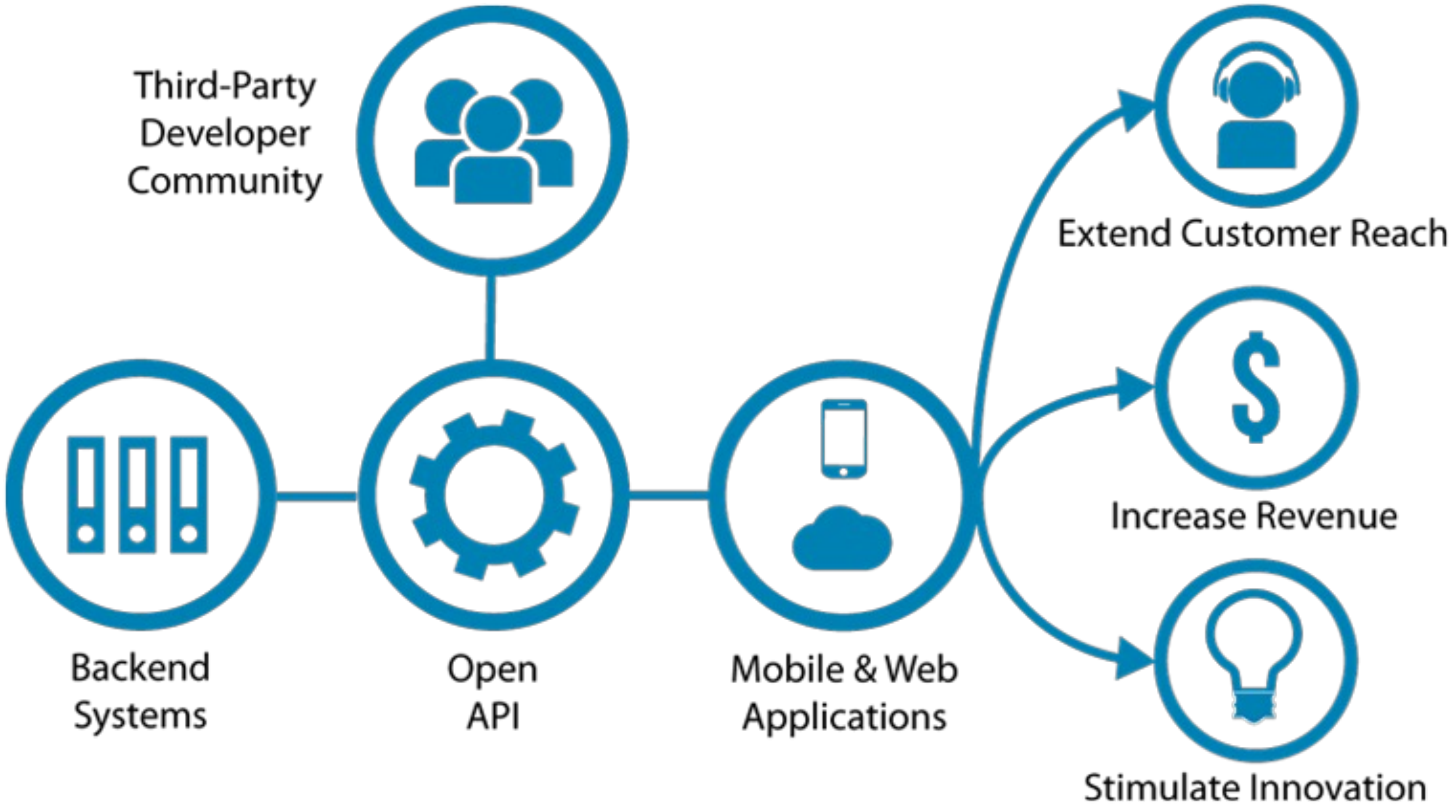
Insights from mobile data provide new opportunities
75% of mobile shoppers take action after receiving a location-based messages

Mobile is about transacting
Evidence: "Cyber Monday"


Leverage Industry Transformations
Transform the value chain and business operations
Deliver Contextually Relevant Experience
Drive Revenue and Productivity
Deepen Engagement
Customers
Partners
Employees

What is the API economy

The use of “business APIs” to positively affect the company



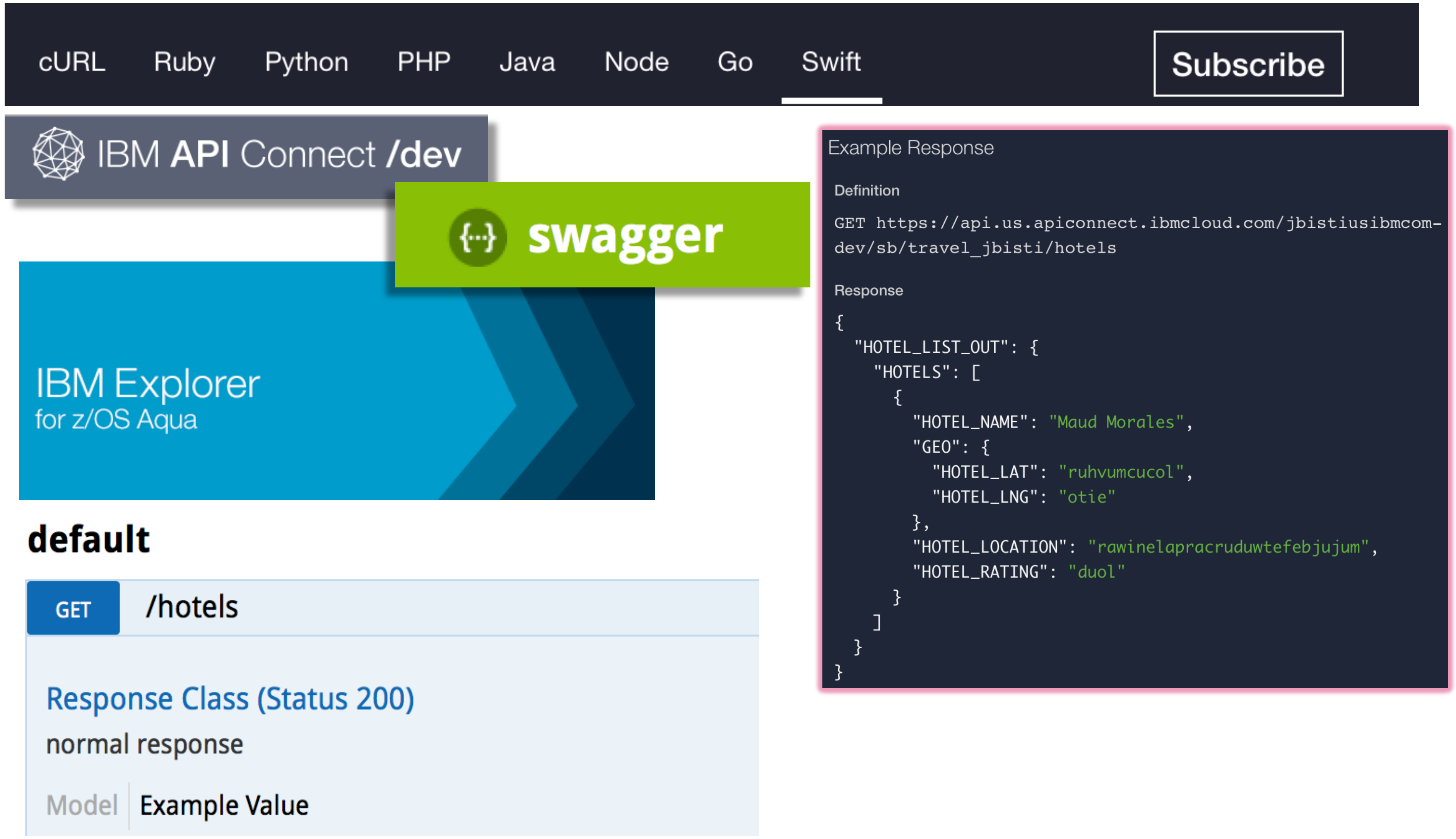
Expectations of Cloud

Misconceptions of z/OS

- I can connect to whatever I want
- I can try things out without penalty
- I won't be tied to any one single solution
- Everything will be familiar and standard

- Doesn't support modern technologies
- Changes take too long
- Security policies seem foreign
- zSystems doesn't integrate

The Reality



default

GET /hotels

Response Class (Status 200)

normal response

Model | Example Value

Example Response

Definition

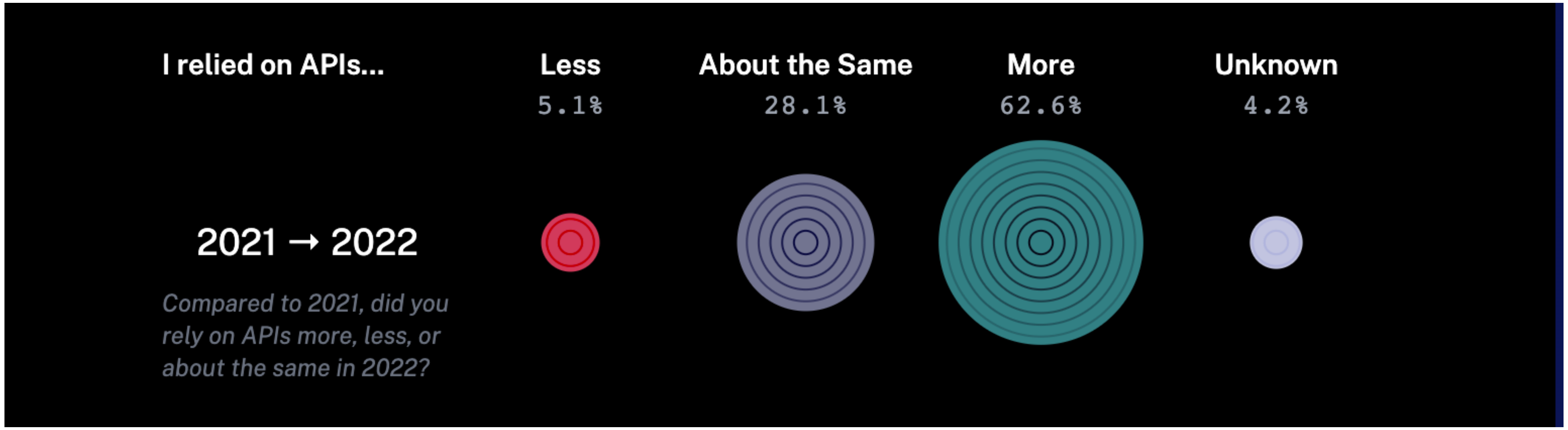
```
GET https://api.us.apiconnect.ibmcloud.com/jbistiusibmcom-
dev/sb/travel_jbisti/hotels
```

Response

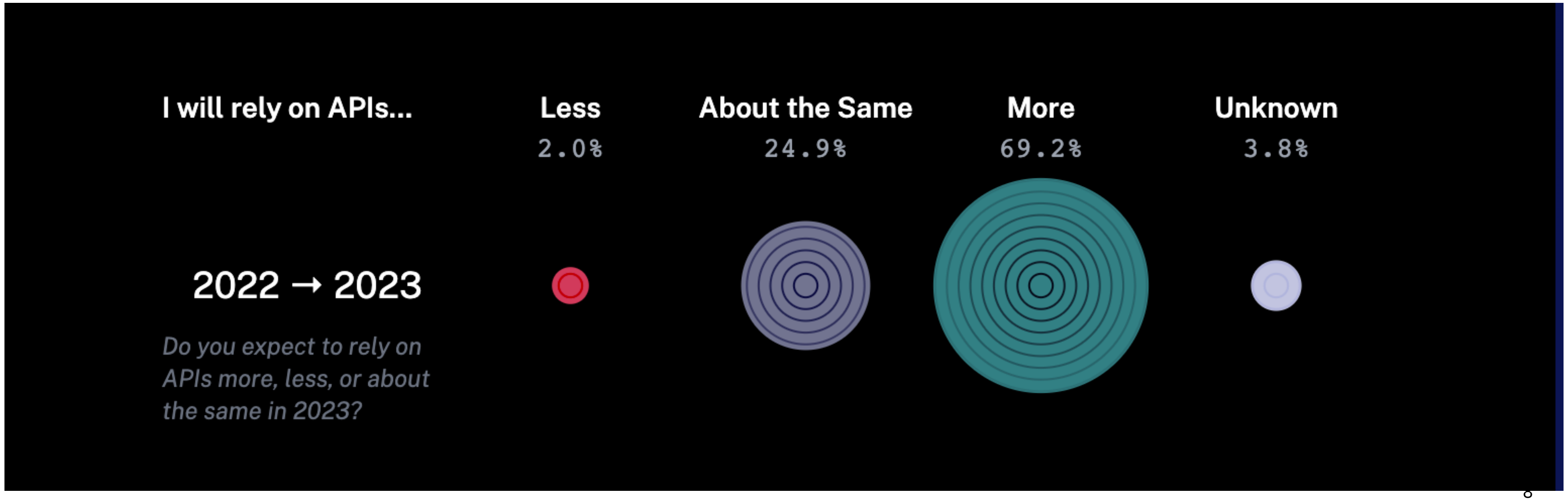
```
{
  "HOTEL_LIST_OUT": {
    "HOTELS": [
      {
        "HOTEL_NAME": "Maud Morales",
        "GEO": {
          "HOTEL_LAT": "ruhvumcucol",
          "HOTEL_LNG": "otie"
        },
        "HOTEL_LOCATION": "rawinelapracruduwtefebjujum",
        "HOTEL_RATING": "duol"
      }
    ]
  }
}
```

Relying on APIs

Over 62.6% of developers reported relying on APIs more in 2022 than they did in 2021. Additionally, 69.2% expect to rely on APIs even more in 2023.



Compared to 2021, did you rely on APIs more, less, or about the same in 2022?



Do you expect to rely on APIs more, less, or about the same in 2023?

RESTful APIs

Technical overview

What is a RESTful API?

REST stands for Representational State Transfer architecture. (It is sometimes spelled "**ReST**".)

- stateless,
- client-server,
- cacheable communications protocol
  - ✓ HTTP protocol is used

A RESTful API is an application programming interface (API) that uses HTTP requests to GET, PUT, POST, and DELETE data.

NOTE

Db2 native REST only supports the POST method for applications.

GET can be used for some system related functions only, not applications. The z/OS Connect API Editor allows you to reassign POST to a different method.

This is why Db2 native REST is REST, while z/OS Connect is RESTful

Key principles of REST

Method **URI** URI=Uniform Resource Identifier

Use HTTP verbs for Create, Read, Update, Delete (CRUD) operations

```
GET
POST
PUT
DELETE
```

```
http://<host>:<port>/path/parameter?name=value&name=value
```

URIs represent things (or lists of things)

Query Parameters are used for refinement of the request

Request/Response Body is used to represent the data object

```
GET http://www.acme.com/customers/12345?personalDetails=true


RESPONSE: HTTP 200 OK
BODY { “id” : 12345
       “name” : “Joe Bloggs”,
       “address” : “10 Old Street”,
       “tel” : “01234 123456”,
       “dateOfBirth” : “01/01/1980”,
       “maritalStatus” : “married”,
       “partner” : “http://www.acme.com/customers/12346” }
```

REST and JSON

Typically, REST and JSON are the interface and data payload format:

Representational **S**tate **T**ransfer **(REST)**

```
http://www.myhost.com:port/account/update
```

← The application understands what to do based on the URI.

↑ Using HTTP verbs: GET, PUT, POST, etc.

Java**S**cript **O**bject **N**otation **(JSON)**

```
{
  "account":  "12345",
  "lastName": "Smith",
  "action":   "Deposit",
  "amount":   "$1000.00",
}
```

← Data is represented as a series of name/value pairs.

← This is serialized and passed in with the URI or returned with a response.

HTTP Request Method Examples

Using the URL: **https://myhost.com/customer/235**

[GET]	= Record for customer #235.
	(in SQL terms - SELECT)
[PUT] + Info	= Updated record for customer #235.
	(in SQL terms - UPDATE)
[POST] + Info	= New record for customer #235.
	(in SQL terms - INSERT)
[DELETE]	= Customer #235 Deleted.
	(in SQL terms - DELETE)

NOTE

Db2 native REST can only use POST for applications, however this can be paired with SELECT, INSERT, UPDATE, DELETE, TRUNCATE, and WITH. For example, you can use the POST method with the SQL issuing a DELETE.

Why is REST so popular?

Increasingly Common

Relatively Lightweight

Ubiquitous Foundation

Stateless

Relatively Easy Development

Db2 for z/OS REST Services

Technical overview

Db2 for z/OS REST objectives

Using REST and JSON to invoke one SQL statement or Stored Procedure

Enabling new business value for your enterprise data

Modernizing using the power of SQL

Unleashing Db2 data for the API Economy

Db2 REST service properties

Db2 REST service invocation

- Direct Db2 DDF REST access

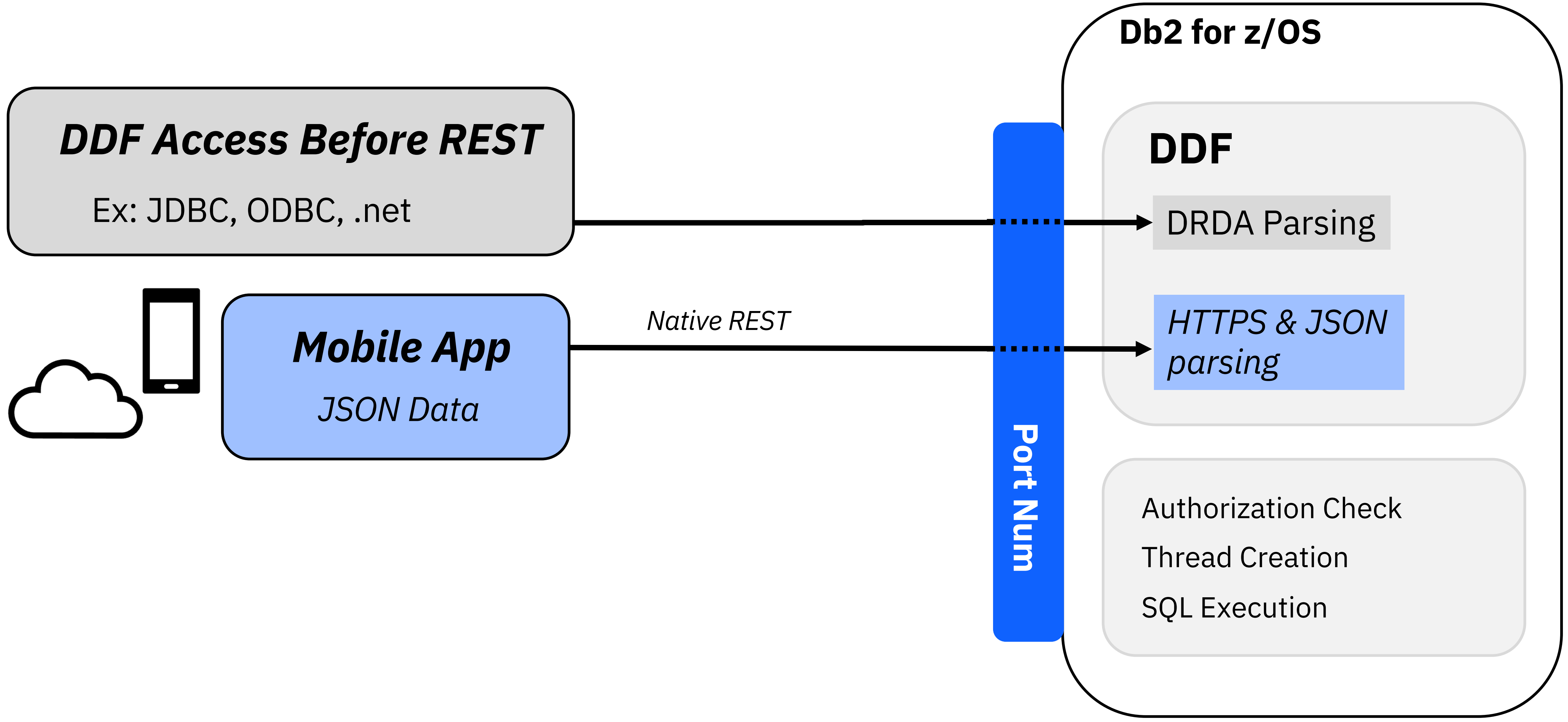
Service details

- One SQL statement or Stored Procedure call is permitted per service
  - Service is a statically bound package in Db2
- CALL, DELETE, INSERT, SELECT, TRUNCATE, UPDATE and WITH
  - MERGE can be in a service comprised of a Stored Procedure, not in a service comprised of a single SQL statement

All the various Db2 base SQL data types

- Including BLOB, CLOB, DBCLOB and XML

Architecture Diagram



When a developer goes to retrieve data

Max

Mobile App Developer

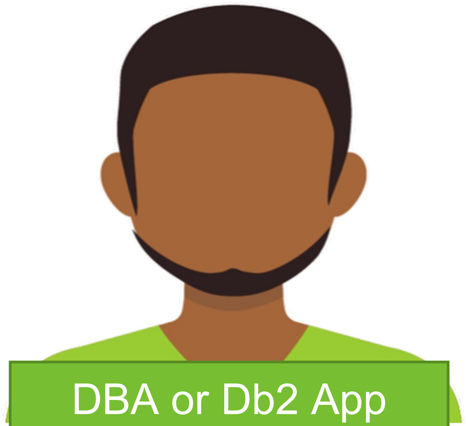
Invokes a Db2 REST service

- Service consists of stored procedure or SQL statement

Output returns in JSON format

Doesn't need to know SQL, nor that the data came from Db2

Devon



DBA or Db2 App Developer

Creates a Db2 REST service

- Service consists of a stored procedure or SQL statement

Creates or reuses the stored procedure or SQL statement used in the REST call

Doesn't need to know JSON

Managing Db2 REST services

REST client in browser

- Typically a plug-in

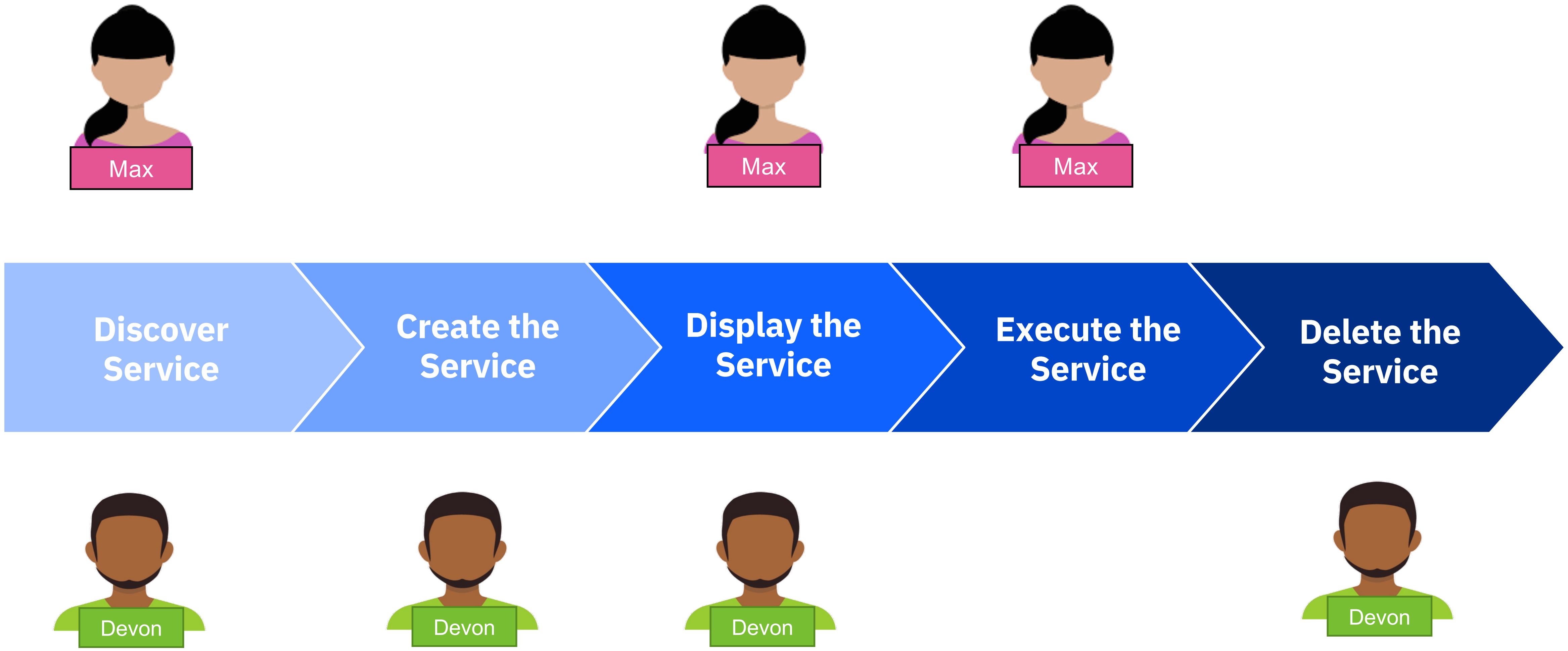
REST app

- Browser look and feel

BIND subcommand

- Standard Db2 interaction

Db2 REST Service Progression



Db2 REST Service Progression

Max

Devon

Discover Service → Create the Service → Display the Service → Execute the Service → Delete the Service

Before you begin

You must have one of the following privileges or authorities to discover all Db2 REST services:

- Execute privilege on the package for the service
- Ownership of the service
- SYSADM or SYSCTRL authority
- System DBADM

Procedure

To discover all services, issue an HTTP or HTTPS GET or POST request through a REST client with the following URI:

- POST https://<host>:<port>/services/Db2ServiceDiscover
    - *Note: Set the HTTP header Accept and Content-Type fields to application/json for the POST request.*
- GET https://<host>:<port>/services

To discover all services using a browser use the following URL:

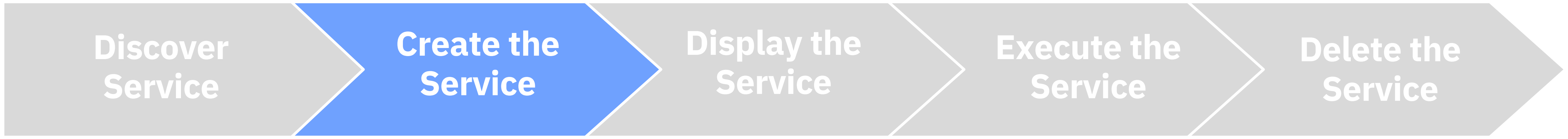
- https://<host>:<port>/services

Successful completion:

REST Status Code = 201.

This is an HTTP code - not Db2!

Db2 REST Service Progression



Before you begin

When you create a service, Db2 identifies you or the authorization ID that you use as the default owner of the service.

Therefore, you must have the required privileges to create a service and bind the associated package into a collection.

For example, you must be authorized to execute the SQL statement that is embedded in the service.

Procedure

To create a service, issue an HTTP or HTTPS POST request through a REST client with the following URI:

- POST https://<host>:<port>/services/Db2ServiceManager
    - *Note: Set the HTTP header Accept and Content-Type fields to application/json for the POST request.*

Specify the following HTTP body for the request – note the JSON name pair format:

```
{ "requestType": "createService",
 "sqlStmt": "<sqlStatement>",
 "collectionID": "<serviceCollectionID>",
 "serviceName": "<serviceName>",
 "description": "<serviceDescription>",
 "bindOption": "<bindOption>"}
```

Successful completion:

REST Status Code = 201.

This is an HTTP code - not Db2!

Creating a Db2 REST Service using JCL

Discover Service → **Create the Service** → Display the Service → Execute the Service → Delete the Service

Before you begin

The BIND SERVICE (DSN) subcommand builds an application package that represents a Db2 REST service.

The package owner must have the required authorization, such as SYSADM authority, to execute the SQL statement embedded in a package and to build the package.

Procedure

To create a service, submit a batch JCL job through a TSO interface with the following format:

Example:

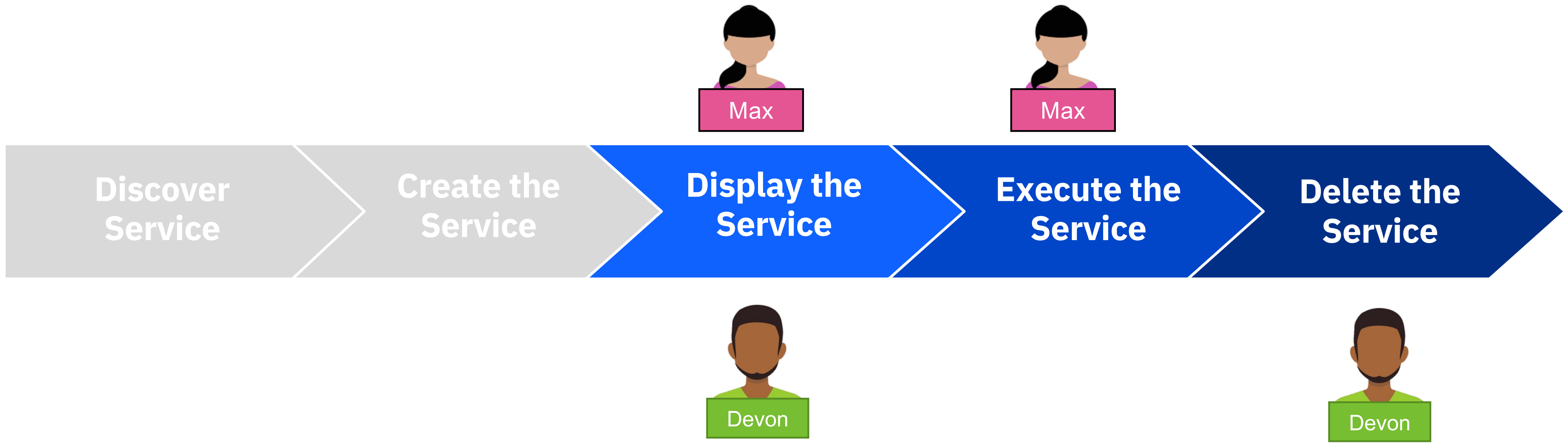
```
//RESTSP   JOB MSGCLASS=H,CLASS=A,NOTIFY=&SYSUID,REGION=0M
//BIND EXEC PGM=IKJEFT01,DYNAMNBR=20
//STEPLIB  DD DSN=DSN1210.DB2.SDSNEXIT,DISP=SHR
//         DD DSN=DSN1210.DB2.SDSNLOAD,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DSNSTMT  DD *
CALL EMPL_DEPTS_NAT(:whichQuery,:department1,:department2)
//SYSTSIN  DD *
DSN SYSTEM(DSN2)
BIND SERVICE("SYSIBMService") -
NAME("selectByDeptSP") -
SQLENCODING(1047) -
DESCRIPTION('Select employees by departments or department range')
```

Successful completion:

0000

This is not an HTTP code!

Db2 REST Service Progression



For more information about these steps, please visit the following pages in IBM Docs:

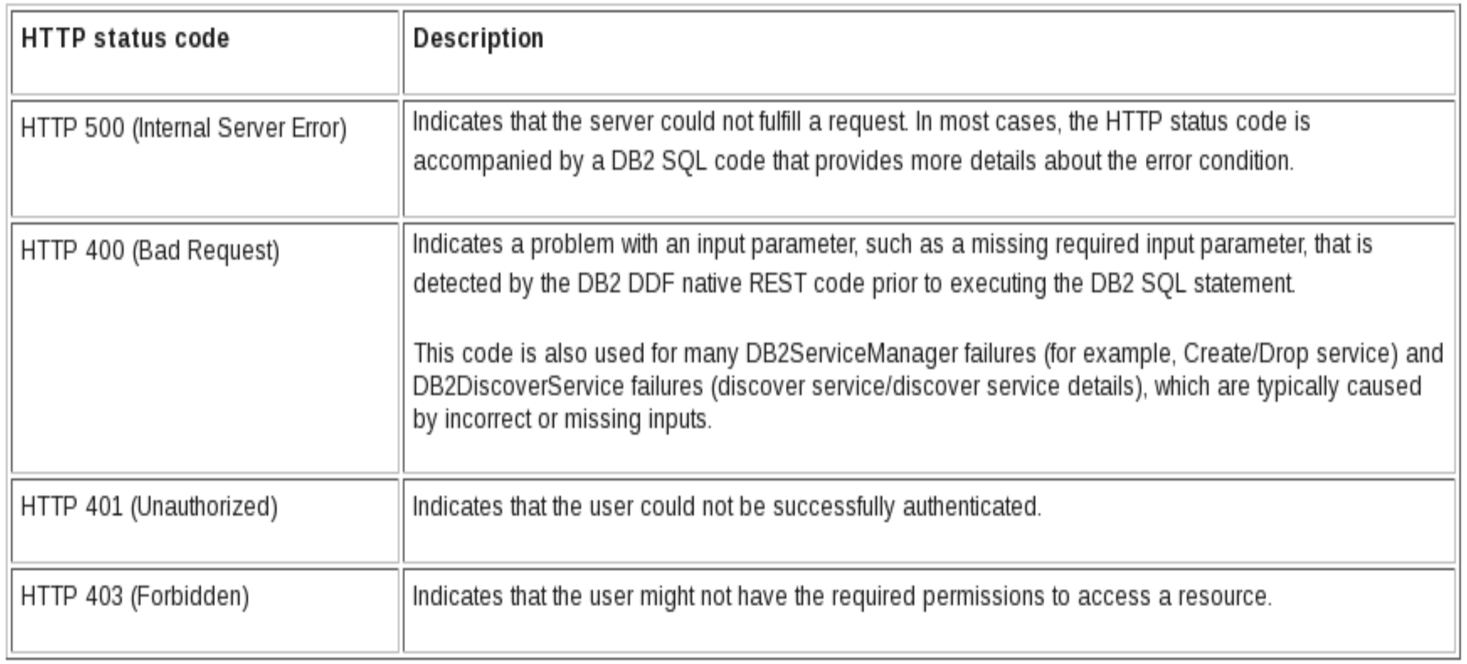
https://www.ibm.com/docs/en/db2-for-zos/12?topic=db2-rest-services

Troubleshooting REST service requests

CATEGORY	DESCRIPTION
1xx: Informational	Communicates transfer protocol-level information.
2xx: Success	Indicates that the client's request was accepted successfully.
3xx: Redirection	Indicates that the client must take some additional action in order to complete their request.
4xx: Client Error	This category of error status codes points the finger at clients.
5xx: Server Error	The server takes responsibility for these error status codes.

Common HTTP status codes for REST service error conditions

For more information on HTTP status codes, please visit:
https://restfulapi.net/http-status-codes/



HTTP status code	**Description**
HTTP 500 (Internal Server Error)	Indicates that the server could not fulfill a request. In most cases, the HTTP status code is accompanied by a DB2 SQL code that provides more details about the error condition.
HTTP 400 (Bad Request)	Indicates a problem with an input parameter, such as a missing required input parameter, that is detected by the DB2 DDF native REST code prior to executing the DB2 SQL statement. This code is also used for many DB2ServiceManager failures (for example, Create/Drop service) and DB2DiscoverService failures (discover service/discover service details), which are typically caused by incorrect or missing inputs.
HTTP 401 (Unauthorized)	Indicates that the user could not be successfully authenticated.
HTTP 403 (Forbidden)	Indicates that the user might not have the required permissions to access a resource.

Versioning Db2 REST Services

APAR PI98649

Versions of REST Services

Allow for development and deployment of new versions of REST Services while existing versions are still being used

Built on existing package versioning support

Use same authorizations

Specify *“version ID”* or accept default “V1”

Select default version

URI Format

Original	/services[/<collection id>]/<service name>
	Example: /services/SYSIBMServices/displayEmployee
Versioning	/services/<collection id>/<service name>[/<version>]
	Example: /services/SYSIBMServices/selectByEmpNum/V1

/services/IBMServices/displayEmployee/

SELECT FNAME, LNAME FROM EMPLOYEE

Versioning ENABLED

/services/IBMServices/selectByEmpNum/V1

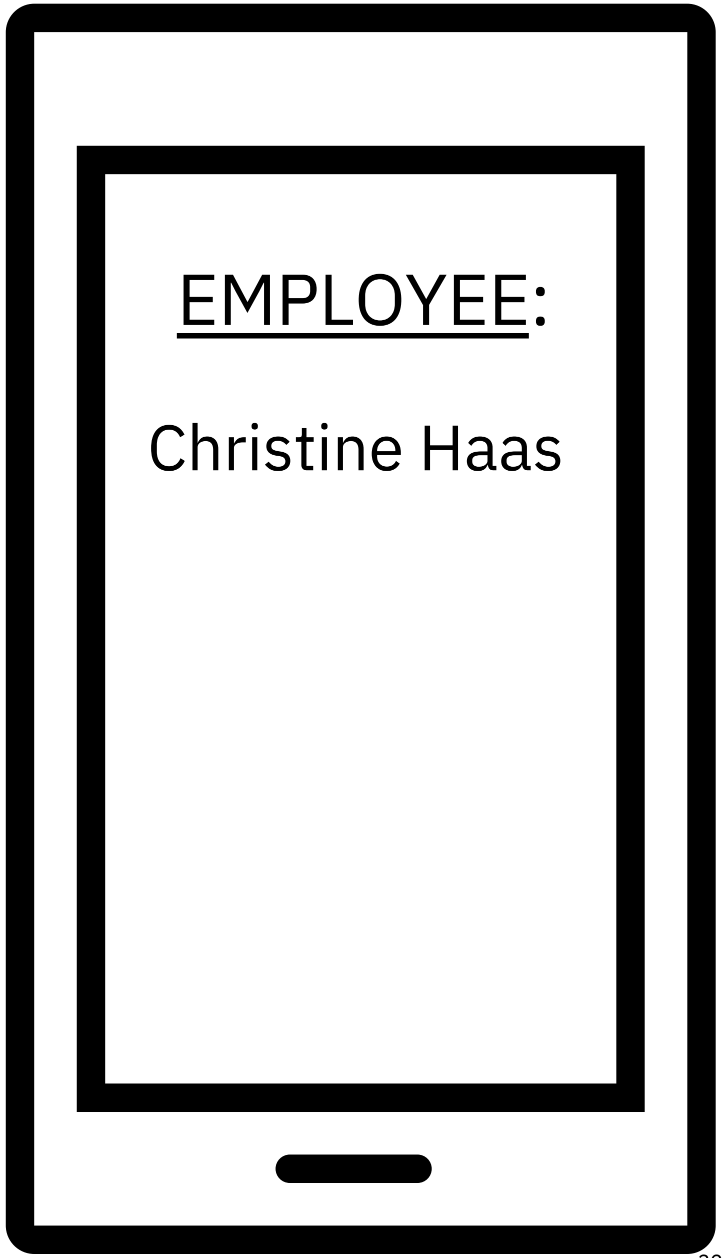
SELECT FIRSTNME, LASTNAME, PHONENO, WORKDEPT FROM DSN81210.EMP WHERE EMPNO = :EMPNUM

/services/IBMServices/selectByEmpNum/V2

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM and E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO

/services/IBMServices/selectByEmpNum/V3

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER, M.PHONENO AS MGRPHONE FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM AND E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO



/services/IBMServices/displayEmployee/

SELECT FNAME, LNAME FROM EMPLOYEE

Versioning ENABLED

/services/IBMServices/selectByEmpNum/V1

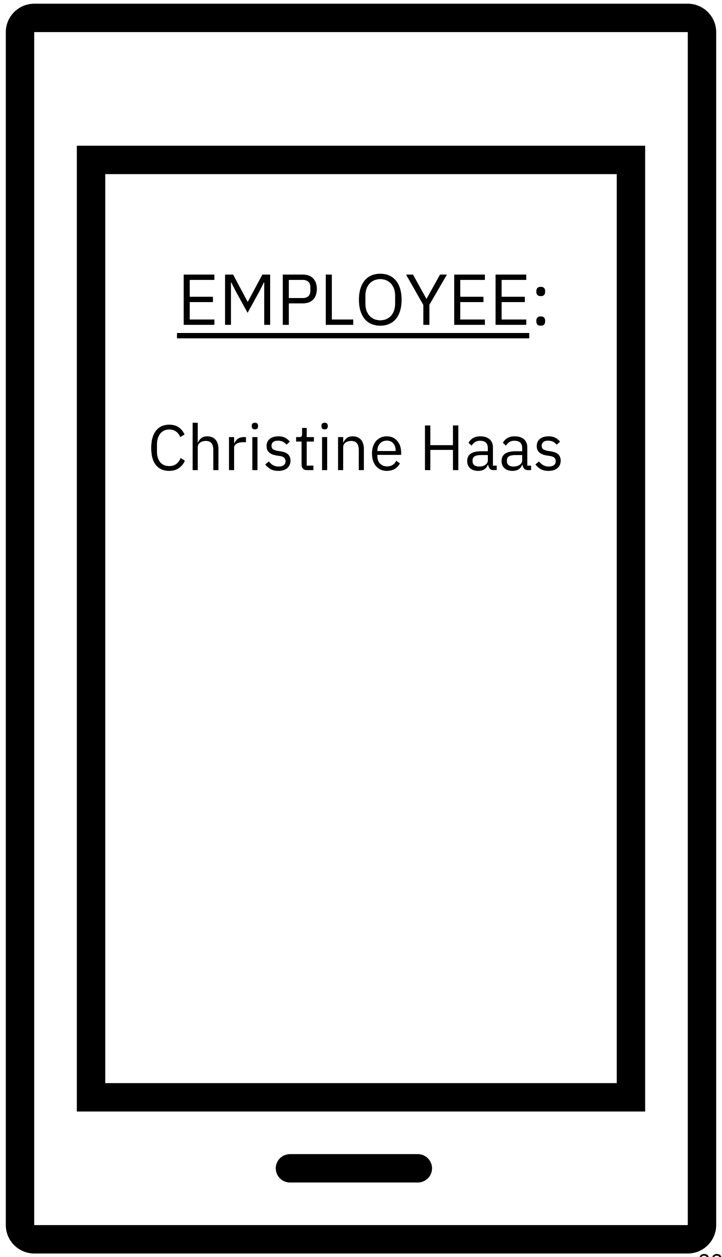
SELECT FIRSTNME, LASTNAME, PHONENO, WORKDEPT FROM DSN81210.EMP WHERE EMPNO = :EMPNUM

/services/IBMServices/selectByEmpNum/V2

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM and E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO

/services/IBMServices/selectByEmpNum/V3

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER, M.PHONENO AS MGRPHONE FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM AND E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO



/services/IBMServices/displayEmployee/

SELECT FNAME, LNAME FROM EMPLOYEE

Versioning ENABLED

/services/IBMServices/selectByEmpNum/V1

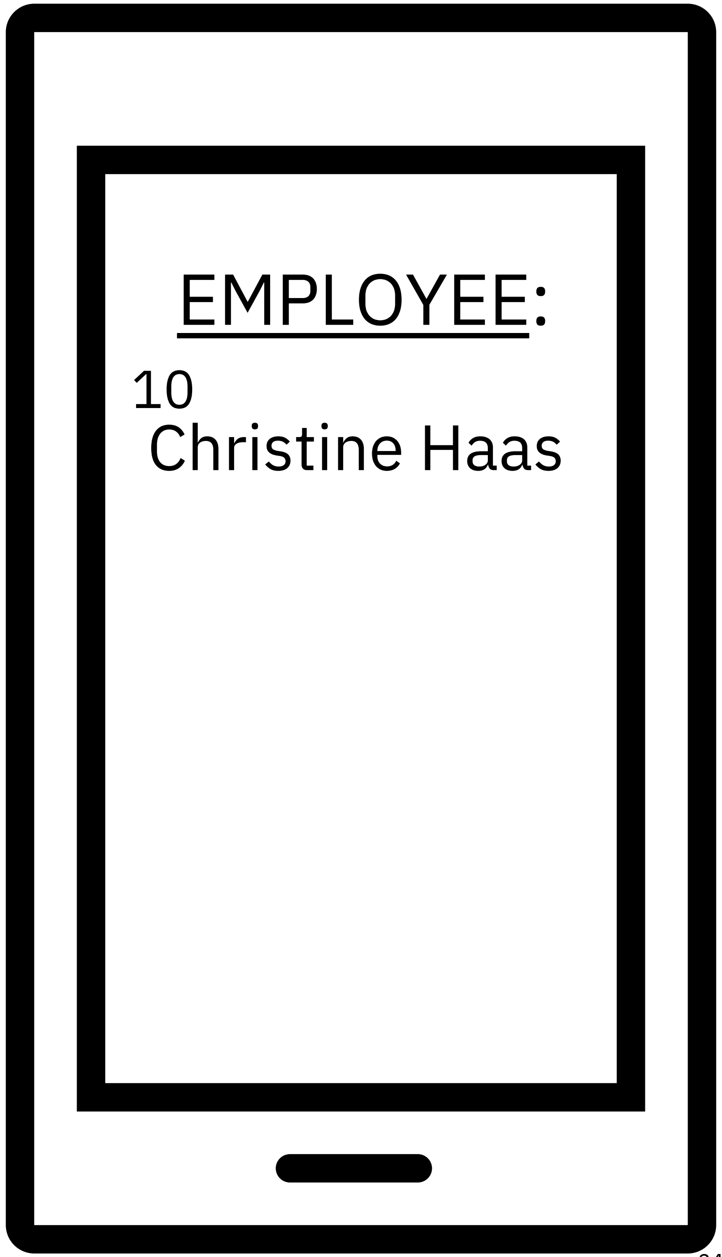
SELECT FIRSTNME, LASTNAME, PHONENO, WORKDEPT FROM DSN81210.EMP WHERE EMPNO = :EMPNUM

/services/IBMServices/selectByEmpNum/V2

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM and E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO

/services/IBMServices/selectByEmpNum/V3

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER, M.PHONENO AS MGRPHONE FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM AND E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO



EMPLOYEE:

10
Christine Haas

/services/IBMServices/displayEmployee/

SELECT FNAME, LNAME FROM EMPLOYEE

Versioning ENABLED

/services/IBMServices/selectByEmpNum/V1

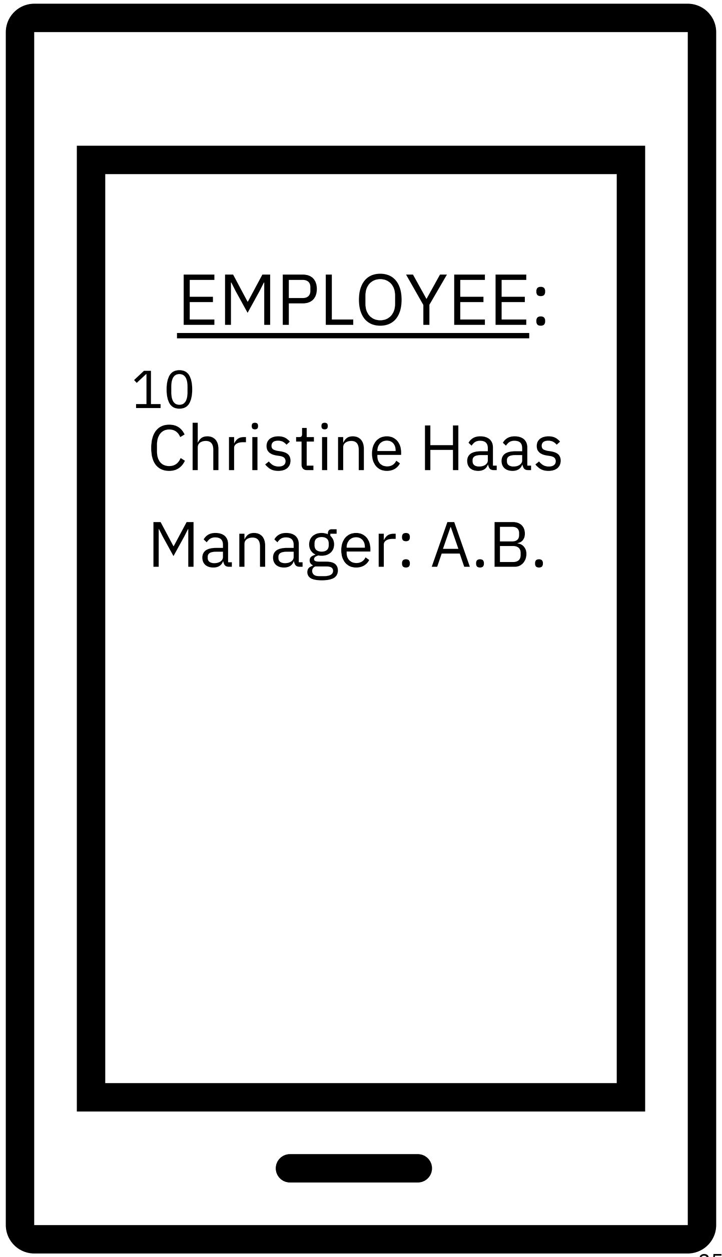
SELECT FIRSTNME, LASTNAME, PHONENO, WORKDEPT FROM DSN81210.EMP WHERE EMPNO = :EMPNUM

/services/IBMServices/selectByEmpNum/V2

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM and E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO

/services/IBMServices/selectByEmpNum/V3

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER, M.PHONENO AS MGRPHONE FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM AND E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO



/services/IBMServices/displayEmployee/

SELECT FNAME, LNAME FROM EMPLOYEE

Versioning ENABLED

/services/IBMServices/selectByEmpNum/V1

SELECT FIRSTNME, LASTNAME, PHONENO, WORKDEPT FROM DSN81210.EMP WHERE EMPNO = :EMPNUM

/services/IBMServices/selectByEmpNum/V2

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM and E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO

/services/IBMServices/selectByEmpNum/V3

SELECT E.FIRSTNME, E.LASTNAME, E.PHONENO, E.WORKDEPT, M.LASTNAME AS MANAGER, M.PHONENO AS MGRPHONE FROM DSN81210.EMP E, DSN81210.EMP M, DSN81210.DEPT D WHERE E.EMPNO = :EMPNUM AND E.WORKDEPT = D.DEPTNO and D.MGRNO = M.EMPNO

EMPLOYEE:
10
Christine Haas
Manager: A.B.
Manager Phone:
123-456-7890

Db2 REST Service Versioning Enablement

Apply Db2 APAR PI98649

Enable versioning by running sample job DSNTIJR2

NOTE

If APAR PI98649 is **removed**, the entire Db2 REST service functionality will be UNAVAILABLE.

Versioning Features

No impact to pre-existing, version-less REST services

Empty string value “” version ID for version-less services

Services created after enablement are always versioned

Simplify modification of services; improve time to market

What if I want to use all the REST methods?

Db2 REST & z/OS Connect

Perfectly paired

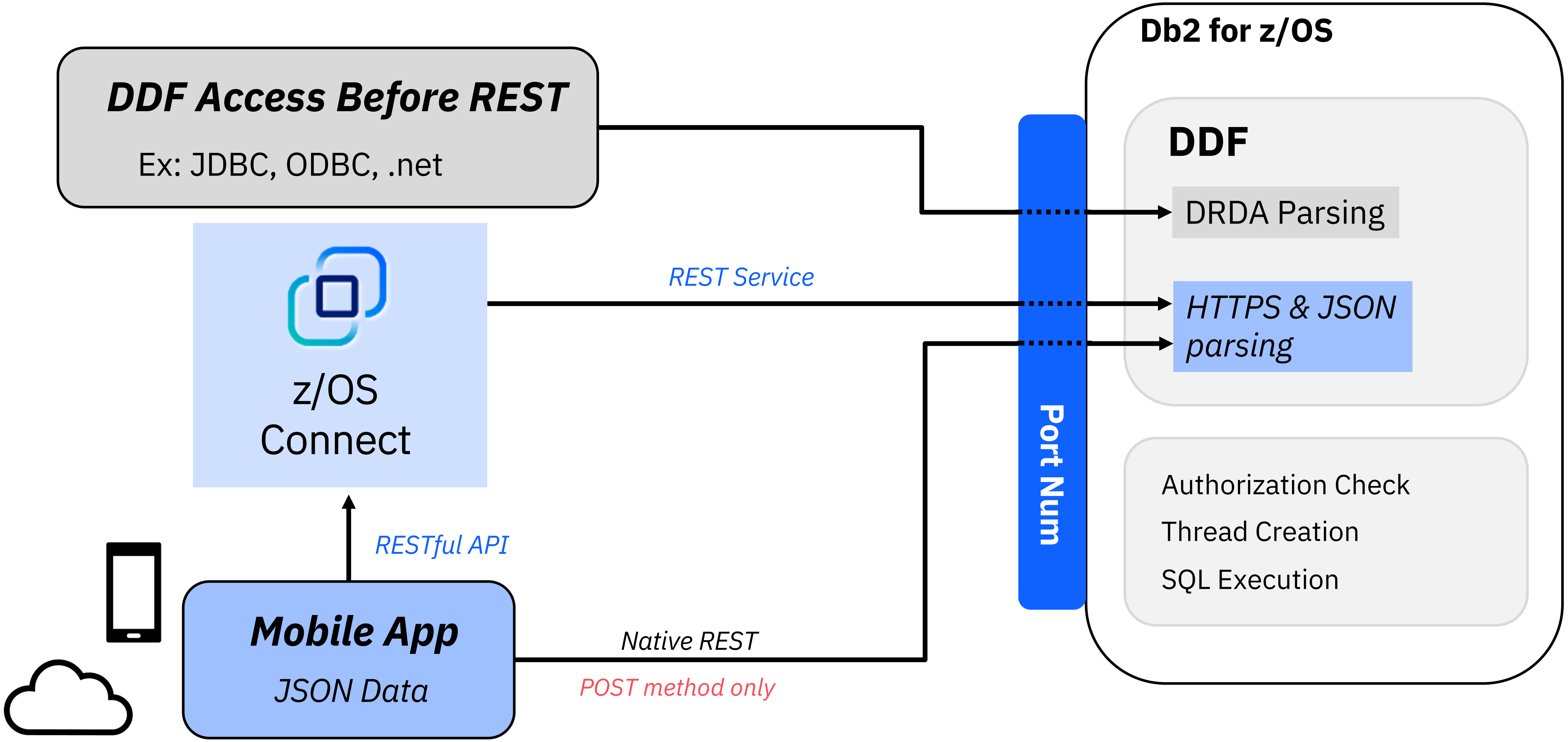
Db2 REST Services with z/OS Connect

Db2 user created native REST services are invoked by the *POST method only*

Mobile and cloud programmers following the RESTful API design model use the *HTTP Methods (Verbs): POST, GET, PUT and DELETE*

z/OS Connect Enterprise Edition's tool "API Editor" can map a Db2 POST method SQL statement to the appropriate RESTful method for a given behavior

Architecture Diagram

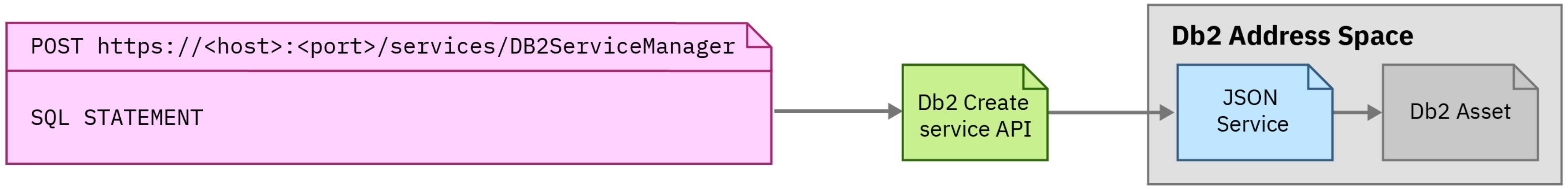


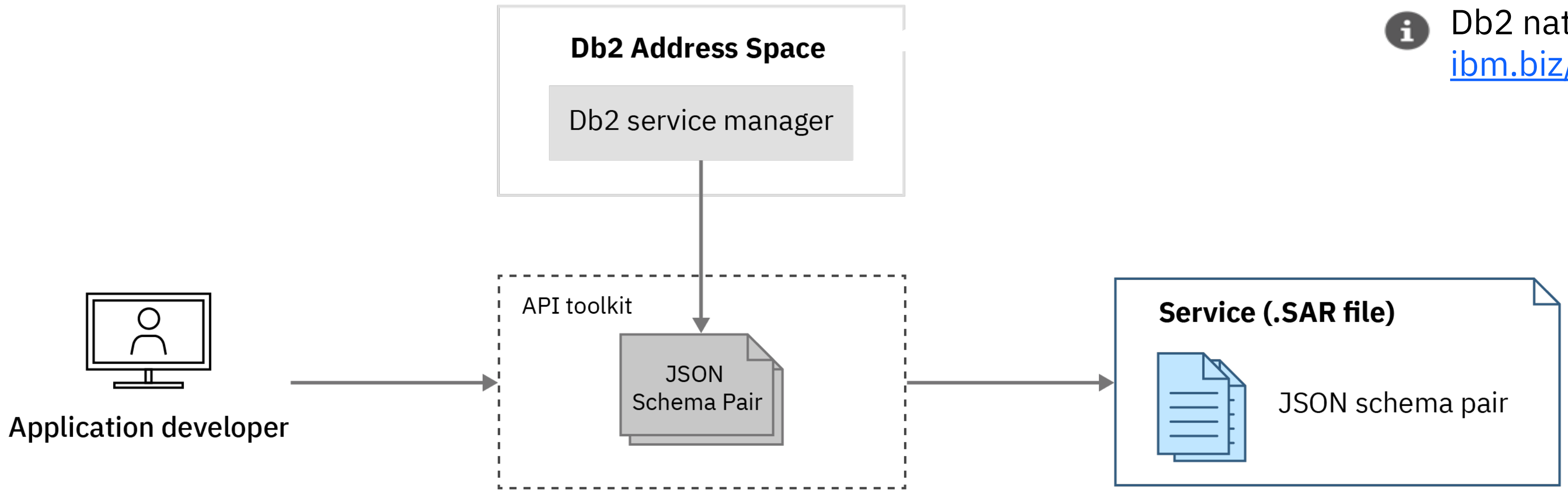
Db2 REST service using a stored procedure with select SQL statements (read only)

Db2 REST service	**METHOD |** POST **URI | http://wg31.washington.ibm.com:1446/services/SYSIBMSERVICE/selectByDeptSP** **HEADERS |** Content-Type = application/json and Accept: application/json **BODY |** {"WHICHQUERY":1,"DEPT1":"A00"}
Db2 REST API with z/OS Connect	**METHOD |** GET **URI | https://wg31.washington.ibm.com:9446/employee/deptNo/A00**

- GET method includes input properties within the URI
- API Editor-created constants reduce the number of input parameters
- Both REST statements produce the same output
- z/OS Connect example follows RESTful standard

Connecting to Db2 with z/OS Connect:
Create the service definition





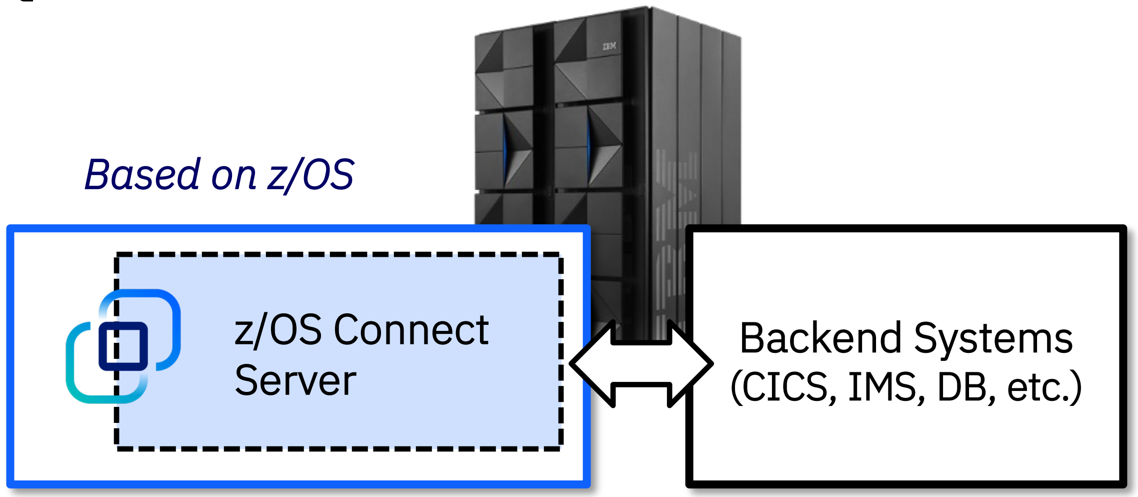
Db2 native REST Service creation
ibm.biz/zosconnect-db2-rest-services

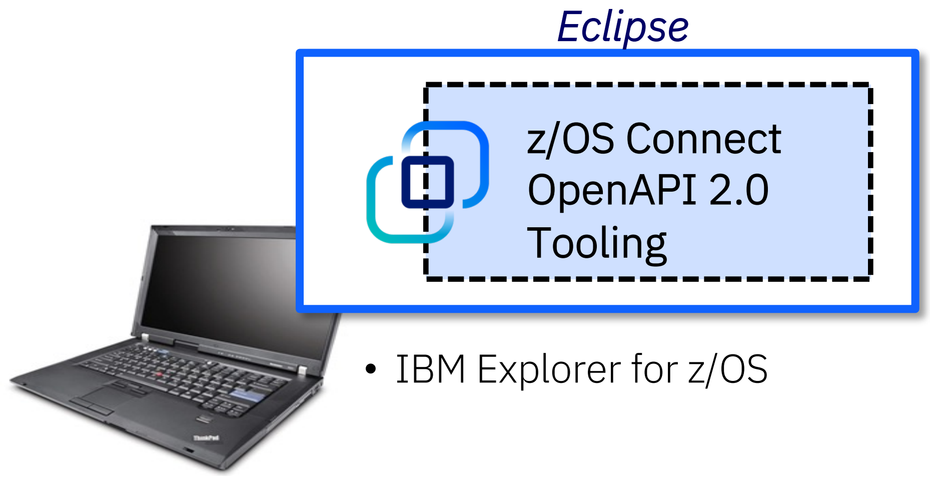
z/OS Connect overview
https://www.ibm.com/products/zos-connect?mhsrc=ibmsearch_a&mhq=z%26sol%3BOS%20Connect

High Level Overview of z/OS Connect

1 Runtime Server

- Hosts APIs you define to run
- Connects with backend system
- Allows for multiple instances



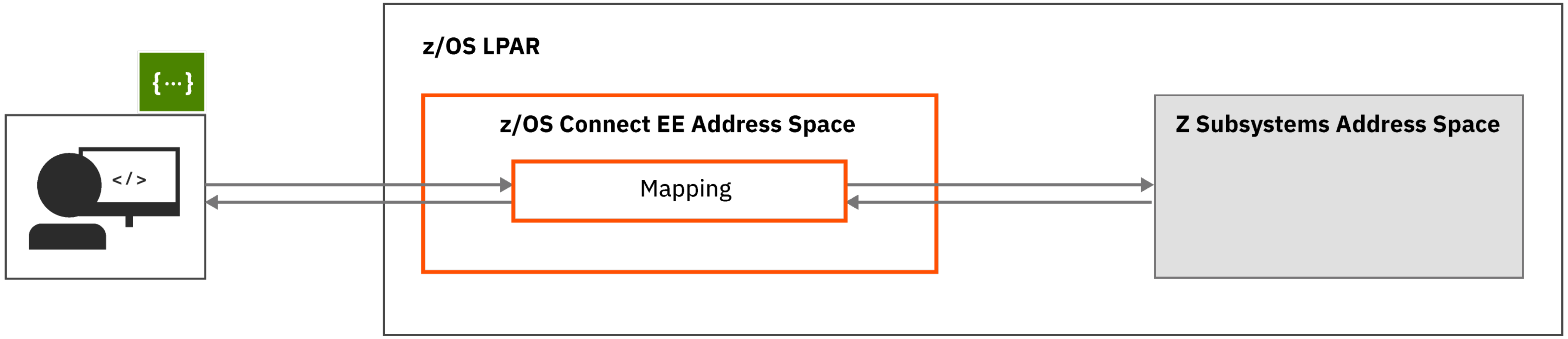


- IBM Explorer for z/OS

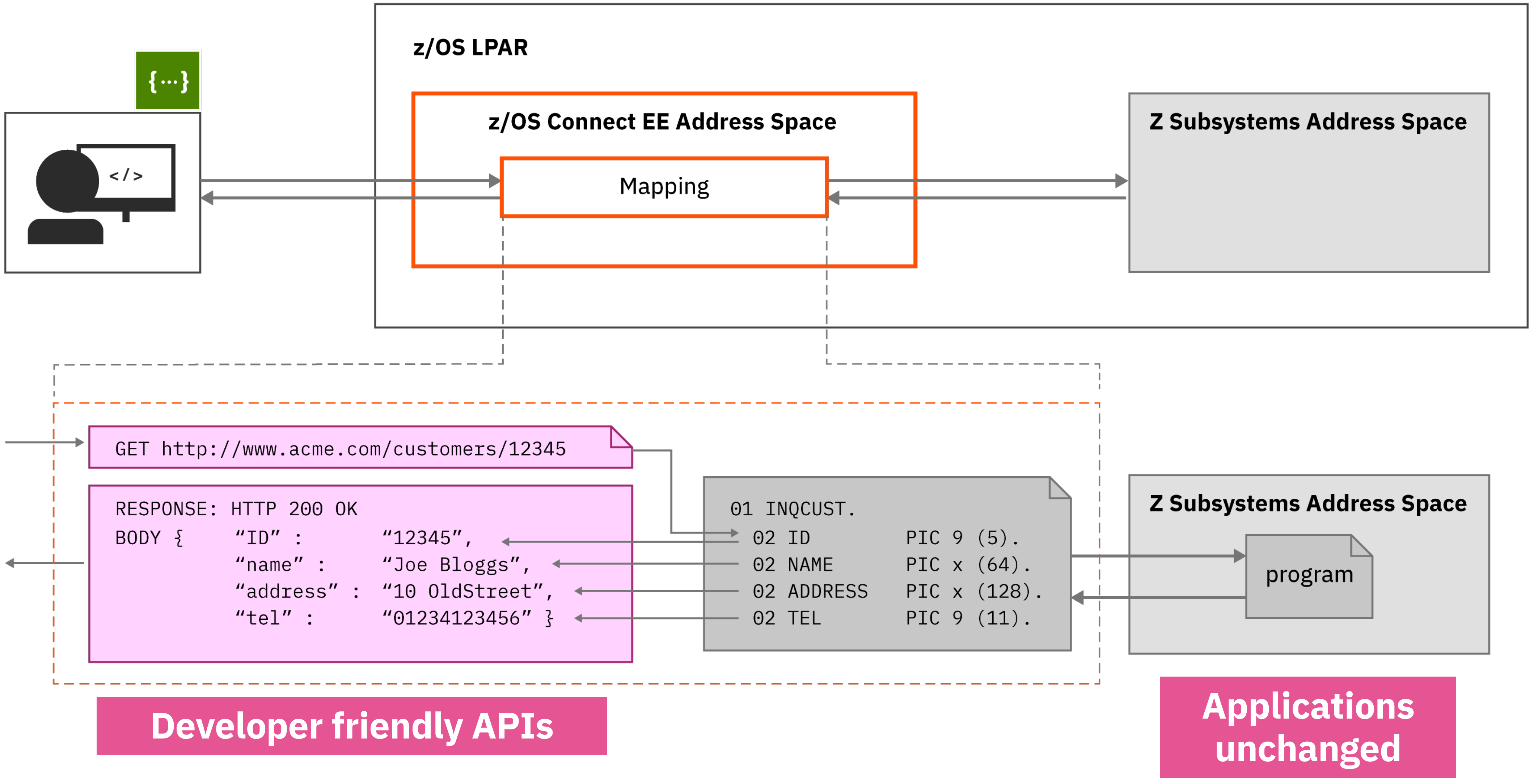
2 Tooling Platform – Eclipse based

- Define APIs
- Define data mapping
- Deploy APIs to runtime server
- Create an Open API document
- Export API archive for other tools to deploy

Data Mapping: Overview



Data Mapping: A Closer Look


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

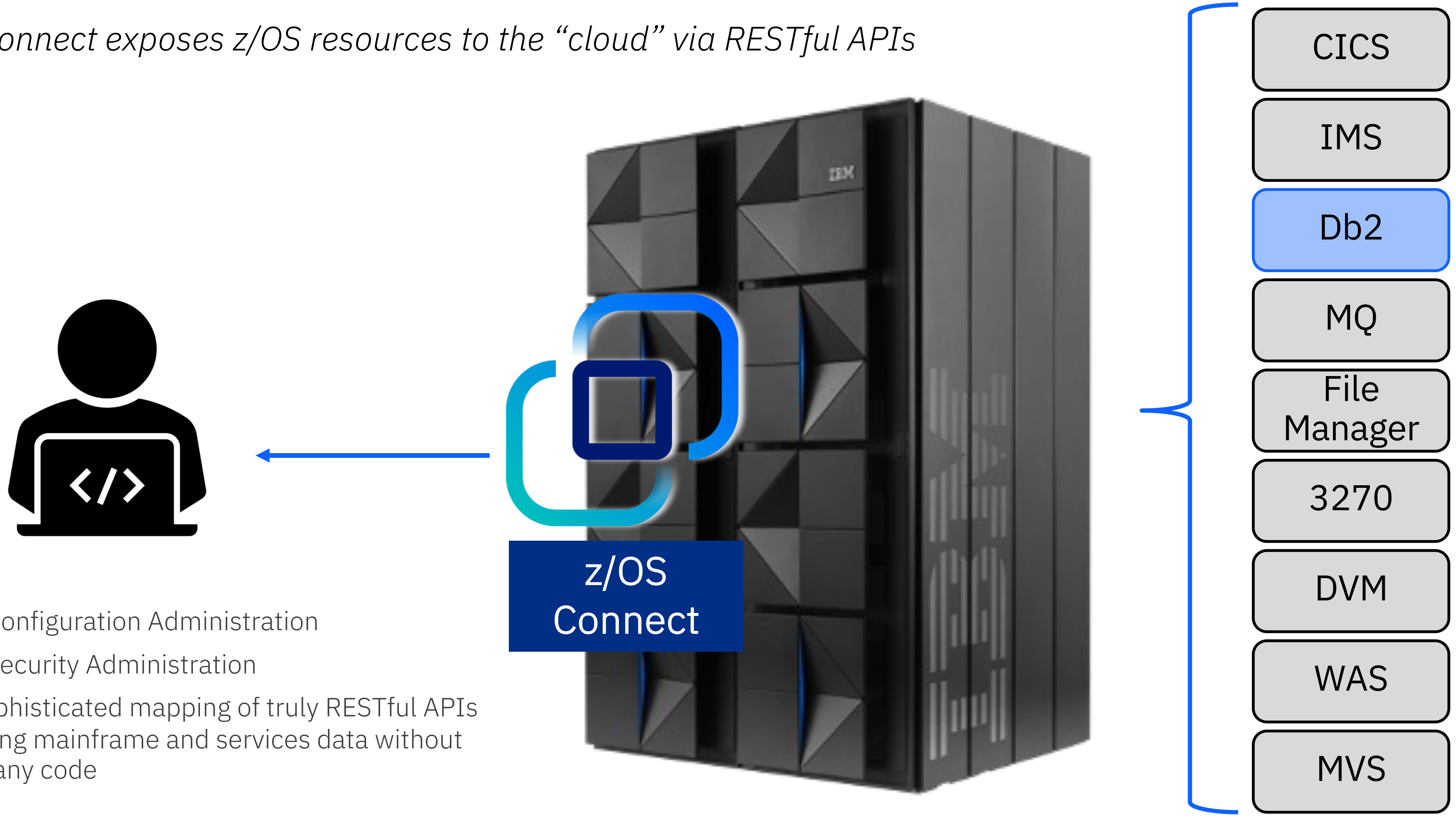
```
GET http://www.acme.com/customers/12345
```

```
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).
```

Single Point of Entry

z/OS Connect exposes z/OS resources to the "cloud" via RESTful APIs



Single Configuration Administration

Single Security Administration

With sophisticated mapping of truly RESTful APIs to existing mainframe and services data without writing any code

Running on REST:
New access with native REST services

The Customer:

A large US manufacturer

Business Challenge:

The company needed a simple portal to navigate Db2 for z/OS

Their Need:

The manufacturer needed an easy access point to navigate around Db2 for z/OS.

Our Solution:

Db2 native REST services were used to create a web-browser UI and tool for interacting with Db2 for z/OS.

Customer Benefit:

All DBAs, new and experienced, could smoothly navigate Db2 for z/OS using a familiar interface.

APIs deliver peace of mind to customers during a global crisis

The Customer:

A large automotive company

Business Challenge:

The automotive company wanted to rapidly automate their manual process for loan extensions .

Their Need:

The company needed to rapidly automate their manual loan extension request process to handle at speed the huge increase of finance extension requests (19,000 per day) driven by the COVID-19 outbreak.

Our Solution:

IBM z/OS Connect generated REST APIs that could be easily called from applications on any platform.

Kubernetes®, microservices and z/OS Connect API enablement enabled unprecedented acceleration to create new user experiences owned by multiple groups within the enterprise.

Customer Benefit:

Leveraging on z/OS Connect’s APIs to rapidly innovate processes in days/weeks. Digitization of business processes enables customer needs to be met and the business the capacity to focus recourses on processes requiring manual customization.