SOAP/ REST and IBM i & Watson
Tim Rowe - timmr@us.ibm.com
Business Architect Application Development
What is an API - Agenda

• What is an API
• What is a Web Service
• SOAP vs REST
  – What is SOAP
  – What is REST
  – Benefits
  – Drawbacks
Connections

There

Applications

Devices

Here

© 2017 International Business Machines Corporation
You have no upcoming trips.

Add a booking

© 2017 International Business Machines Corporation
API Definition
Application Programming Interface
API Definition

Application programming interface

From Wikipedia, the free encyclopedia

"API" redirects here. For other uses, see API (disambiguation).

In computer programming, an Application Programming Interface (API) is a set of subroutine definitions, protocols, and tools for building application software. In general terms, it is a set of clearly defined methods of communication between various software components. A good API makes it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer. An API may be for a web-based system, operating system, database system, computer hardware or software library. An API specification can take many forms, but often includes specifications for routines, data structures, object classes, variables or remote calls. POSIX, Microsoft Windows API, the C++ Standard Template Library and Java APIs are examples of different forms of APIs. Documentation for the API is usually provided to facilitate usage.
APIs - Simple

Simple way to connect endpoints. Send a request and receive a response.
Example

Kitchen
The API Economy

Not just a buzz-word, but rather the evolution of services-oriented IT. Allows users, businesses & partners the ability to interact in new and different ways resulting in the growth (in some cases the revolution) of business.
What is the API economy?

- Cloud, mobile and social - business as-a-service economies
- Data has considerable value and can be monetized given an easy-to-consume API
- APIs standard building block for doing development
  - Mobile
  - Web
- Easy integration with other apps and services
- APIs provide consumability for 3rd Parties – broaden your reach
What is the API Economy

Creating value by offering APIs that others want

Using APIs to help your developers innovate freely

Supporting your mobile development team with APIs

© 2017 International Business Machines Corporation
What is the API Economy

Making APIs the common language in a hybrid world

Linking devices to data on the Internet of Things (IoT)
What is a Web Service?

... a service?

A repeatable business task – e.g., check customer credit; open new account

Available on the Web
SOAP vs REST
Simple Object Access Protocol

© 2017 International Business Machines Corporation
What is SOAP

• Exposes operations that implement logic
• Designed for distributed computing
• Standardized
• Aligns with Enterprise Application needs
  • Support multi transport
  • Enterprise security – WS.Security
  • Governance with strong typing
  • Broad Development tooling support
• XML Based message protocol
• Uses WSDL as a contract between consumer and provider
REpresentational State Transfer
What is REST

• Architectural Style as described by Roy Fielding
• Resource focused
• Every request is via hyperlink ie http request
• Easily consumed by any client, especially web clients
• Light weight
  • Uses JSON vs XML
  • No required header for each message
• Resources are driven by HTTP Specification
  • GET, PUT, DELETE, POST
Why one vs the other? Philosophical Difference

SOAP
- Enterprise Driven
- Contract based
- Robust Infrastructure
- More Security Options

Rest
- Simplicity
- Small packet size
- HTTP focused
- Easy to call from JavaScript
SOAP vs. REST example

Is the light bulb currently on?

**SOAP**

Service: LightbulbManagement([http://w3/services/lightbulbMgmt](http://w3/services/lightbulbMgmt))

Operation: getState
   - Input: lightbulbID [string]
   - Output: state [On | Off]

Operation: setState
   - Input: lightbulbID [string], state [On | Off]
   - Output: void

**REST**

Resource: lightbulbs/sam

Operation: GET
   - Output: state [On | Off]

Operation: POST
   - Input: state [On | Off]
   - Output: nothing or resource representing the new state

```python
obj.getState(sam)
```

```
http://w3/lightbulbs/sam
```

LightBulbManagement
SOAP vs. REST example data flows

**SOAP request**

POST /services/LightBulbManager HTTP/1.1
Host: example.com
Content-Type: text/xml; charset=UTF-8
SOAPAction: "LightBulbManager#getState"

```xml
<?xml version='1.0' ?>
<env:Envelope xmlns:env="...">
  <env:Body>
    <ns1:getState xmlns:ns1="...">
      <in0 xsi:type="xsd:string">SAM</in0>
    </ns1:getState>
  </env:Body>
</env:Envelope>
```

**REST request**

GET http://w3/lightbulbs/SAM HTTP/1.1
Host: example.com
Accept: application/xml
Creating Web Services on IBM i

Tim Rowe - Business Architect for Application Development

timmr@us.ibm.com
Using a REST API with Watson

Search flights

From ➔ To:

DALLAS to BOSTON

09/07/2017

For example: Flight from houston to CHICAGO

https://ibm-i-watson-test.mybluemix.net/
What are the characteristics of a web service?

**Web Service**
- Encapsulated
  - Access through interface
- Reusable
  - Write once – use everywhere
- Stateless
  - Information not retained
- Event driven
  - No required order
- Loosely coupled
  - Callable from anywhere

**Traditional subroutine**
- Global data
  - Access directly
- Reuse by copy
  - Maintain everywhere
- Stateful
  - Information retained in job
- Application driven
  - Fixed order
- Tightly coupled
  - Tied to application
How do you connect your IBM i to the outside world?
URL – http://getSpecialReport

- Normal IBM i Job

- Process Request
- Parse the Web Stuff
- Call back end PGM

- Http Server
  - Web Server
  - Listening for Requests

- Application Server

- RPG PGM
  - Super Special Report

- Db2 Data
Rational Developer for i

RDi & Modernization Tools: Web Services Wizard

More Information:
Host Access Transformation Services

IBM Rational HATS Toolkit

1. Create a macro that steps through your application.
2. Create an Integration Object.
3. Create web service support files.
4. Create the web service and WSDL.

More Information:
http://www-01.ibm.com/software/awdtools/hats/
IBM i Integrated Web Services Environment

IBM i: Integrated Web Services Server  SOAP & REST

Included with IBM i
About integrated web services

• Released December of 2007 on IBM i 5.4, 6.1, and 7.1
  – Installed as part of base operating system option 3
  – Always load latest HTTP Group PTF for latest fixes and enhancements
• Consists of two separate entities
  – Integrated web services client for ILE
  – Integrated web services server
• Latest information, including product prerequisites, can be found at http://www.ibm.com/systems/i/software/iws/

• Continues to be re-invented and enhanced on 7.2 & 7.3
IWS server API requests – mapping & transformation

SoE: Consumer view
/myAPI/resource/{id}

HTTP Headers + JSON

POST
GET
PUT
DELETE

IWS Server

API Mapping Model

SoR

RPG A
RPG B
COBOL

API Package

Discover APIs w/Swagger

Export API

PCML

IBM apiconnect

IBM Bluemix

© 2017 International Business Machines Corporation
Web service client/server flow does not change

• When a web services server is created, an associated HTTP server is also created
  – You can go straight to the web services server, but if you need SSL or basic authentication, you need to do it via the HTTP server
• The web service provider implementation code (i.e. RPG or COBOL programs or service programs) are run in separate jobs
What are all the parts...

- HTTP Apache Server
  - Connector to the IAS server
- IAS Server (Liberty)
  - JAX-RS (REST)
  - JAX-WS (SOAP)
- Java program
  - Handles Inputs
  - Calls the backend ILE Program
  - Converts Output back to Web format
About integrated web services server REST support

- Supported in IBM i 7.1 and 7.2 and 7.3
  - On version 2.6 of integrated web services server
  - Server will handle both SOAP and REST services
- Uses JAX-RS
  - Java API for RESTful Web Services
- Two ways to deploy a REST service
  - IBM Web Administration GUI updated
    - Deploying a REST service will require more user input than when deploying a SOAP service
  - QShell script installWebService.sh updated to support REST
How to Get Started - Rest

- Identify the RPG / Cobol / Java pgm
- Figure out the HTTP methods
  - GET  - read activities
  - POST - create entries
  - PUT   - update an entry
  - DELETE – remove
- Determine the URI - Identifiers
  - Use Nouns vs Verbs
  - Keep it simple
Getting Started

Examples

• GET /defects: list all bugs.
• GET /defects/123: Retrieve bug 123.
• POST /defects/123: Create a new defect 123 with the POST request body.
• PUT /defects/123: Update defect 123 with the PUT request body.
• DELETE /defects/123: delete defect 123
Getting Started

• Specify media types (e.g. XML, JSON, etc.) the procedure will accept

• Specify media types the procedure will return

• Optionally specify what values to inject in procedure input parameters
  • Path segment (e.g. /accounts/{id})
  • Matrix parameters (e.g. /cars;color=blue)
  • Query parameters (e.g. /cars?color=blue)
  • Form data
  • HTTP headers
  • HTTP Cookies

• Optionally designate response code and HTTP header output parameters
Create web services server


Click on the Create New Web Services Server link
Web Integration Permissions

In the past, any user wanting to use Web Admin they were required to have *ALLOBJ and *IOSYSCFG special authority!

System Security policy just does not allow this!

‘Permissions’ Support

- Now a *USER granted ‘permission’ can use the GUI
- Group profiles are now supported
Web Integration Permissions

• Developers can use Web Admin
  – No longer need *ALLOBJ special authority
  – Administrators can grant users ‘Permission’
  – Empowering the User
  – Group Profile support

• Two Permissions Available
  – Operator – Start & Stop servers
  – Developer – All functions

Integrated GUI interface now available to Developers and Operators without compromising your system security
How do you test things?

Free Download  https://www.soapui.org
Create web services server (cont.)

Step 1: Specify server name.

Welcome to the Create Web Services Server wizard. A Web services server provides a convenient way to externalize existing programs running on IBM i, such as RPG and COBOL programs, as Web services. Web service clients can then interact with these IBM i program-based services from the Internet or intranet via Web service-based industry standard communication protocols such as SOAP. The clients can be implemented using a variety of platforms and programming languages such as C, C++, Java and .NET. This wizard creates everything needed to run Web services.

For more information, please visit: http://www-03.ibm.com/systems/softwares/ws/

Specify a unique name for this server 🟣

Server name: WSSERVICE3

Server description: Web services server created by the Create Web Se
Create web services server (cont.)

Step 2: User Profile for web container.

Create Web Services Server
Specify User ID for Server - Step 2 of 3

The server requires an IBM i user ID to run the server’s jobs. It is recommended that a special user ID is specified to run the server’s jobs since this user ID is given authority to all of the server’s objects, such as files and directories.

Specify user ID for this server.

- Use default user ID
  
  Note: The default server user ID is QWSERVICE.

- Specify an existing user ID

- Create a new user ID
Create web services server (cont.)

**Step 3: Create the server**

<table>
<thead>
<tr>
<th>IBM Web Administration for</th>
<th>Setup</th>
<th>Manage</th>
<th>Advanced</th>
<th>Related Links</th>
</tr>
</thead>
</table>

**Create Web Services Server**

**Summary - Step 3 of 3**

**Servers**

**Web Services Server Information**
- **Server name**: WSERVICE3
- **Server description**: Web services server created by the Create Web Services Server wizard.
- **Internal port range**: 10076 - 10095
- **Server root**: Wwww/WSERVICE3
- **Server URL**: http://ip280124.nchland.ibm.com:10086
- **User ID for server**: GWSERVICE
- **Context root**: Wweb

**HTTP Server Information**
- **HTTP server name**: WSERVICE3
- **HTTP server description**: Web services server created by the Create Web Services Server wizard.
- **Port**: 10086
- **Document root**: Wwww/WSERVICE3/wt/docs
- **Server root**: Wwww/WSERVICE3
- **Server association**: WSERVICE3
Create web services server (cont.)

Once created, the server is started and deployed sample service started
Install web service

Select “Deploy New Service” to install a new web service

The web service server provides a convenient way to externalize existing programs running on IBM i, such as RPG and COBOL programs, as Web services. Web service clients can then interact with these IBM i program-based services from the Internet or intranet using Web service-based industry standard communication protocols such as SOAP. The clients can be implemented using a variety of platforms and programming languages such as C, C++, Java and .NET. An easy-to-use wizard is provided to configure the Web services server and the services for IBM i program objects. Other management functions such as starting, stopping and deleting services are also provided.

For more information, please visit: http://www-03.ibm.com/systems/i/software/ws/
Step 1: Select Next to install a new web service

**Deploy New Service**

---

Welcome to the Deploy New Service wizard. This wizard helps you externalize an IBM i program as a Web service which allows interfacing with clients from the Internet or intranet using Web service based industry standard communication protocols such as SOAP. The wizard guides you through the steps to specify the IBM i program object and allow you to select the program's procedures to be externalized. Once the criteria are specified, a new Web service is deployed to this server, therefore allowing access to the selected functionally provided by the program object.

The program object to externalize as a Web service must be an existing Integrated Language Environment (ILE) program (*PGM) or service program (*SRVPGM) object. Currently, only program objects created using the COBOL or RPG programming languages are supported.
Install web service (cont.)

Step 2: What program or service program contains the web service?

Deploy New Service

Deploy New Service. Specify Location of IBM i Program Object - Step 2 of 3

The IBM i object to be externalized as a Web service must be an existing ILE program (*PGM) or service program (*SRVPGM) located on the system. Currently, only program objects written using the COBOL or RPG programming languages are supported.

Specify the library and program object for the Web service.

- Specify IBM i library and ILE program object name (Recommended)
  - You can specify the program object location by entering the name of the library that contains the program object, as well as the name of the program object. This is the fastest and recommended way to locate the program object.
  - Library name: [WSRRI]
  - ILE Object name: [NFS400_THA]
  - ILE Object type: *SRVPGM *PGM

- Browse the integrated file system for the IBM i program object
Install web service (cont.)

Step 3: What should we call this new web service?

**Deploy New Service**

Deploy New Service: Specify Name for Service - Step 3 of 9

Specify a unique name for this service. 

- Service name: FindCustomers
- Service description: NFS400_THR
Install web service (cont.)

**Step 4: What procedures should be externalized as web service operations?**

<table>
<thead>
<tr>
<th>Select</th>
<th>Procedure name/Parameter name</th>
<th>Usage</th>
<th>Data type</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GETCITYNAME</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FINDLOCITIES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FINDFROMCITIES</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GETFLIGHTINFO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FINDFLIGHTS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FINDFLIGHTSDESC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GETCUSTNAME</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GETCUSTNUMBER</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>FINDCUSTOMERS</td>
<td>POSITION</td>
<td>input</td>
<td>char</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LISTTYPE</td>
<td>input</td>
<td>char</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COUNTRY</td>
<td>input</td>
<td>int</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COUNTRID</td>
<td>input</td>
<td>int</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CUSTLIST</td>
<td>output</td>
<td>select</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COUNTRID</td>
<td>output</td>
<td>select</td>
</tr>
</tbody>
</table>

© 2017 International Business Machines Corporation
Install web service (cont.)

Step 5: Specify user profile for the web service

Deploy New Service

Deploy New Service: Specify User ID for this Service - Step 5 of 9

The service requires an IBM i user ID to run the program object that contains the Web service business logic.

Specify User ID for this Service:

- Use server's user ID
- Specify an existing user ID

The user ID must have the necessary authority to this program object and any other additional program objects.

User ID: [MyUser]

Update the server's user ID to have *USE authority to this user ID.
Install web service (cont.)

Step 6: Specify library list for the web service

The functionality of the IBM i program you want to externalize as a Web service may depend upon other IBM i programs in the system. Specify all libraries in which programs exist that the Web service programs depend on. If no library is specified, a default library list is used.

Specify library list position for this Web service:
- Insert libraries in front of user library portion of the library list
- Insert libraries at the end of user library portion of the library list

Library list entries:

<table>
<thead>
<tr>
<th>Library name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSR11</td>
</tr>
<tr>
<td>flight400</td>
</tr>
</tbody>
</table>

Add | Remove | Remove All | Move up | Move down | Continue

© 2017 International Business Machines Corporation
Install web service (cont.)

Step 7: Specify what request information should be passed to web service

**Deploy New Service**

Deploy New Service: Specify Transport Information to Be Passed - Step 7 of 9

Specify transport information to be passed to the web service implementation code.

**Information to be passed to web service implementation code**

Specify Transport Metadata:

- **Transport Metadata**
  - □ REMOTE_ADDR

Specify HTTP Headers:

- **HTTP Headers**
  - There are no entries for this table.
  - Add | Remove All

© 2017 International Business Machines Corporation
Install web service (cont.)

Step 8: Specify WSDL options for web service

Deploy New Service

Specify options that control what is generated in the Web Services Description Language (WSDL) file.

Specify WSDL Options

- Generate web service bindings for SOAP 1.1 protocol: Enable
- Generate web service bindings for SOAP 1.2 protocol: Disable
- Generate nullable elements for all fields: Disable
- Generate optional elements for all fields: Disable
- Expose service metadata: Enable
### Step 9: Ready to deploy the new web service – Services tab

<table>
<thead>
<tr>
<th>IBM Web Administration for</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup</td>
<td>Manage</td>
</tr>
<tr>
<td>All Servers</td>
<td>HTTP Servers</td>
</tr>
<tr>
<td>Running</td>
<td>Server</td>
</tr>
</tbody>
</table>

#### Deploy New Service

**Summary - Step 9 of 9**

When you click Finish the web service is deployed.

<table>
<thead>
<tr>
<th>Services</th>
<th>Operations</th>
<th>Request Information</th>
<th>WSDL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>FindCustomers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td>NFS400_THR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service Install path:</td>
<td>www/wwservlet3/wr/web/services/services/FindCustomers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User ID for service:</td>
<td>MyUser</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Program:</td>
<td>/GYS/USB/MVSRI.LBD/NFS400_THR.SRVPGM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Library list for service:</td>
<td>MVSRI/LIB/FLS400</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

After the service is installed, use this URL to access the Web service definition: http://ip23:824:10035/web/services/FindCustomers?wsdl.
Install web service (cont.)

After a few seconds, service is installed and started

Manage Deployed Services

Data current as of Apr 3, 2012 3:55:04 PM.

Deployed services:

<table>
<thead>
<tr>
<th>Service name</th>
<th>Status</th>
<th>Startup type</th>
<th>WSDL - service definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FindCustomers</td>
<td>Running</td>
<td>Automatic</td>
<td>View definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Manage the web service

You can view WSDL as long as server is active

<table>
<thead>
<tr>
<th>Service name</th>
<th>Status</th>
<th>Startup type</th>
<th>WSDL - service definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvertTemp</td>
<td>Running</td>
<td>Automatic</td>
<td>View definition</td>
</tr>
<tr>
<td>FindCustomers</td>
<td>Running</td>
<td>Automatic</td>
<td>View definition</td>
</tr>
</tbody>
</table>

© 2017 International Business Machines Corporation
Manage the web service (cont.)

View the WSDL file (partial listing below)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:tns="http://findcustomers.webservices" targetNamespace="http://findcustomers.webservices">
  <wsdl:types>
    <xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
        targetNamespace="http://findcustomers.webservices">
      <xsd:element name="findcustomers/XMLSchema">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="arg0" type="xsd:FINDCUSTOMERInput"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:complexType name="FINDCUSTOMERInput">
        <xsd:sequence>
          <xsd:element name="COUNTREC" type="xsd:int"/>
          <xsd:element name="LISTTYPE" type="xsd:string"/>
          <xsd:element name="POSITION" type="xsd:string"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:element name="findcustomers/XMLSchemaResponse">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="return" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="findcustomers"/>
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="arg0" type="xsd:FINDCUSTOMERInput"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:schema>
  </wsdl:types>
</wsdl:definitions>
```

© 2017 International Business Machines Corporation
Manage the web service (cont.)

You can test the web service (but not over SSL and only for SOAP 1.1)
Manage the web service (cont.)
Manage the web service (cont.)

You can view and modify web service properties

Manage Deployed Services

Data current as of Apr 3, 2012 4:05:14 PM

Deployed services:

<table>
<thead>
<tr>
<th>Service name</th>
<th>Status</th>
<th>Startup type</th>
<th>WSDL - service definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvertTemp</td>
<td>Running</td>
<td>Automatic</td>
<td>View definition</td>
</tr>
<tr>
<td>FindCustomers</td>
<td>Running</td>
<td>Automatic</td>
<td>View definition</td>
</tr>
</tbody>
</table>

© 2017 International Business Machines Corporation
Manage the web service (cont.)

Web service properties – General tab

Service Properties

Name: FindCustomers
Description: NFS400_THR
Startup type: Automatic
Service install path: /nfs/systemp/aonr/nfsws/ws/wservice3/services/FindingCustomers
Program: /systemp/aonr/nfsws/ws/wservice3/services/FindingCustomers/program
Web service definition URL: http://ip:/wssrv:10086/services/FindingCustomers?wsdl
WSDL target namespace URI: http://findingcustomers.webonly.iseries
User ID for this service: MyUSER

Select "Update the server's user ID to have *USE authority to this user ID."
About integrated web services server REST support

- Supported in IBM i 7.1, 7.2, & 7.3
  - On version 2.6 of integrated web services server
  - Server will handle both SOAP and REST services
- Uses JAX-RS
  - Java API for RESTful Web Services
- Two ways to deploy a REST service
  - IBM Web Administration GUI updated
    - Deploying a REST service will require more user input than when deploying a SOAP service
  - QShell script installWebService.sh updated to support REST
Best practices for REST services

- Use HTTP methods as CRUD (create/read/update/delete) operations: POST (create), GET (read), PUT (update), DELETE (delete)
- URI design matters
  - Use nouns, not verbs (/accounts/{id} not /getaccount?id=nn)
  - Predictable
  - Learn from popular APIs (Google, Facebook, Twitter, etc.)
- Keep them stateless (independent)
- Don’t send data that is not needed
- Think about cacheability
  - To improve network efficiency, scalability and user-perceived performance of your API
- Think about pagination, querying, sorting
New things to set when deploying a REST web service

- Specify the URI path to the resource (e.g. /accounts)
- For each procedure (resource method)
  a) Specify the HTTP method the procedure will handle
  b) Optionally specify URI segment path for the procedure
  c) Specify media types (e.g. XML, JSON, etc.) the procedure will accept
  d) Specify media types the procedure will return
  e) Optionally specify what values to inject in procedure input parameters
     Path segment (e.g. /accounts/{id})
     Matrix parameters (e.g. /cars;color=blue)
     Query parameters (e.g. /cars?color=blue)
     Form data
     HTTP headers
     HTTP Cookies
  f) Optionally designate response code and HTTP header output parameters
Procedure and program parameter rules

- No injection to input parameters will be allowed if:
  - There is more than one input parameter that is a structure
  - There is an input parameter that is an array
  - The data type of an input parameter is something other than byte, integer, char, float, packed, or zoned

- If you want to accept JSON or XML as an input parameter, then specify an input parameter that is a structure
  - A resource request method (i.e. procedure) can accept JSON, XML, or both, assuming you indicate what media types the procedure accepts
  - A resource request method can return both types of media types, based on what the client sends on the Accept request header. For example, following example indicates that client only accepts XML responses:

    Accept: application/xml
HTTP response code and headers

• A procedure output parameter with type integer can be designated as the HTTP response code parameter
  – Allows you to control what response code to return (e.g. 405 – not allowed)

• A procedure output parameter that is an array of type char can be designated as the HTTP header parameter
  – Mainly for specifying HTTP caching headers
Demo of REST based Methodology
Deploy new Service – SOAP or REST

Deploy New Service

Specify Web service type - Step 1 of 9

Welcome to the Deploy New Service wizard. This wizard helps you externalize an IBM program object as a Web service.

Specify Web service type:

- **SOAP**
  - A SOAP-based Web service is a self-contained software component with a well-defined interface that describes a set of operations that are accessible over the Internet and exchange XML messages that are based on the SOAP protocol.

- **REST**
  - A REST-based Web service exposes resources, where client requests are handled by resource methods and the format of messages that are exchanged is defined by the resource itself.
REST Service – Specify *PGM or *SVRPGM

**Deploy New Service**

*Specify Location of IBM i Program Object - Step 2 of 9*

The IBM i object to be externalized as a Web service must be an existing ILE program (*PGM) or service program (*SRVPGM) located on the system.

Specify the program object for the Web service.

- Specify IBM i library and ILE program object name (Recommended)
- Browse the integrated file system for the IBM i program object

Alternatively, you can search for the program object in the integrated file system, which could take a while if a directory is specified that contains a lot of objects, such as /QSYS.LIB.

Path of program object: /QSYS.LIB/WSRP.Lib/NFS400_THR.SVRPGM [Browse] /QSYS.LIB/MLIB

Note: Specify a *PGM or *SRVPGM object.
REST – Specify Service Name

Deploy New Service

Specify Name for Service - Step 3 of 9

The Web service to be externalized is a resource. The URI path template identifies matching patterns for incoming HTTP requests. The path is relative to the context root and can be a simple string or one or more template parameters that can contain regular expressions to further restrict what is allowed.

- Resource name: FindCustomer
- Service description: Find the Customers
- URI path template: / e.g. /temperature, /temperature(tempId=)
REST – Select the Export Procedures

- GETCITYNAME
- FINDTOKITIES
- FINDFROMCITIES
- GETFLIGHTINFO
- FINDFLIGHTS
- FINDFLIGHTSDOW
- GETCUSTOMNAME
- GETCUSTOMNUMBER
- FINDCUSTOMERS

- POSITION
- LISTTYPE
- COUTREQ
- COUTRET

© 2017 International Business Machines Corporation
REST – Define the Parameters

Deploy New Service
Specify Resource Method Information - Step 5 of 9

Procedures are mapped to resource methods. Each resource method needs to be defined to handle client requests by mapping an HTTP request method to a resource method.

Specify resource method information.

Procedure name: FINDCUSTOMERS
URI path template for resource: /
HTTP request method: GET
URI path template for method: *NONE
Allowed input media types: *ALL
Returned output media types: *XML_AND_JSON
HTTP response code output parameter: *NONE
HTTP header array output parameter: *NONE
Whether to wrap input parameters:
□ Wrap input parameters
□ Do not wrap input parameters

Input parameter mappings:

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Data type</th>
<th>Input source</th>
<th>Identifier</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>POSITION</td>
<td>char</td>
<td>*QUERY_PARAM</td>
<td>position</td>
<td>*NONE</td>
</tr>
<tr>
<td>LISTS_TYPE</td>
<td>char</td>
<td>*QUERY_PARAM</td>
<td>listtype</td>
<td>*NONE</td>
</tr>
<tr>
<td>COUNTRIES</td>
<td>int</td>
<td>*QUERY_PARAM</td>
<td>country</td>
<td>*NONE</td>
</tr>
</tbody>
</table>
REST – Specify User Profile for the Service

Deploy New Service
Specify User ID for this Service - Step 6 of 9

The service requires an IBM user ID to run the program object that contains the Web service business logic.

Specify User ID for this Service:

- Use server's user ID
  The server's user ID must have the necessary authority to this program object and any other additional program objects.

- Specify an existing user ID
REST – Update the Library List

Deploy New Service

Specify Library List - Step 7 of 9

The functionality of the IBM i program you want to externalize as a Web service may depend upon that the Web service programs depend on. If no library is specified, a default library list is used.

Specify library list position for this Web service:
- Insert libraries in front of user library portion of the library list
- Insert libraries at the end of user library portion of the library list

Library list entries:

Library name
- IWSRT
- [insert 400]

Add | Remove | Remove All | Move up | Move down | Continue
Deploy New Service

Specify Transport Information to Be Passed - Step 8 of 9

Specify transport information to be passed to the web service implementation code.

Information to be passed to web service implementation code

Specify Transport Metadata:
- Transport Metadata
  - REMOTE_ADDR

Specify HTTP Headers:
- HTTP Headers
  - There are no entries for this table.
REST - Finish

Deploy New Service
Summary - Step 9 of 9

When you click Finish, the web service is deployed.

Service  Method  Request Information

Resource name: FindCustomer
Resource description: Find the Customers
Service install path: /www/timmittst/services/services/FindCustomer
URI path template: / 
User ID for service: SERVER (QWSERVICE)
Program: /QSYS.LIB/IWSRLIB/NFS400_THR.SRVPGM
Library list for service: IWSRLIB/FLGHT400

Back  Finish  Cancel
What have we done lately....

- 3 node support
  - HTTP on one node
  - Application Server on a node
  - Backend RPG on a node

- Use Authenticated User

- Services re-deploy
- Connection pool pre-initialization
- Variable length fields

- Many other updates as requested by the community
Calling Web Services from IBM i

Tim Rowe - Business Architect for Application Development

timmr@us.ibm.com
RPG - Where to we start....
RPG - A Modern Business Language

- Interoperability
  - Java
  - XML
  - SQL

- Readability
  - Free form
  - Blank lines
  - Comments

- Functionality
  - Procedures
  - Data areas
  - Data structures
  - More data types
  - Extended file support

- Modern Tools
  - RD i, RTC, ARCAD Power Pack

```plaintext
read file;          // Get next record
dow not %eof(file);  // Keep looping with record
  if %error;
    dsply 'The read failed';
    leave;
  else;
    chain(n) name database data;
    time = hours * num_employees + overtime_saved;
    pos = %scan (',' : name);
    name = %xlate(upper : lower : name);
    exsr handle_record;
    read file;
  endif;
enddo;

begsr handle_record;
  eval(h) time = time + total_hours_array (empno);
  temp_hours = total_hours - excess_hours;
  record_transaction();
endsr;
```
Rational Developer for i

- Modern
- Integrated
- Analysis
- Debugger
- Visual
- Supports RPG, COBOL, CL, C, C++, SQL, DDM
How do you connect RPG to Watson

- Many different ways….
  - SQL
  - Integrated Web Services Client
  - Call another language that speaks ‘REST’
    - Java
    - Node.js
    - Python
Integrated Web Services Client

• Overview
  – Based on Apache AXIS C++
  – Consists
    o Tool to convert WSDL to RPG/C/C++ stubs (SOAP only support)
    o SOAP Client
    o REST Client
  – Availability
    o IBM i SS1 Option 3
    o Latest HTTP group PTF
• Supports
  – C, C++, RPG, COBOL
Integrated Web Services Client

Call the Axis Client Interfaces directly from RPG

Client RPG Application

Axis Client

Server

Request

Response

SOAP/REST Request

SOAP/REST Response

Call/Return

© 2017 International Business Machines Corporation
RPG Calling Axis Client

str1 = 'https://watson-api-explorer.mybluemix.net/' +
    'language-translator/api/v2/translate?model_id=' +
fromLang +'-' + toLang + '&text=' + str1 + 'X'00';

// Create HTTP transport handle.
tHandle = axiscTransportCreate(str1:AXISC_PROTOCOL_HTTP11);

// Set SSL information - turn off SSLv2, SSLv3, TLSv1 and tolerate
// certificate not being in key store
NONE = 'NONE' + 'X'00';
propBuf1 = '/QIBM/USERDATA/ICSS/CERT/SERVER/DEFAULT.KDB' + 'X'00';
propBuf2 = 'true' + 'X'00';

axiscTransportSetProperty(tHandle: AXISC_PROPERTY_HTTP_SSL:
    %addr(propBuf1):
    %addr(NULLSTR): %addr(NULLSTR):
    %addr(NONE) : %addr(NONE):
    %addr(NONE) : %addr(NULLSTR): %addr(NULLSTR):
    %addr(propBuf2));
RPG Calling Axis Client

// Indicate that the payload in response should stay in UTF-8
propBuf2 = 'false' + X'00';
axiscTransportSetProperty(tHandle: AXISC_PROPERTY_CONVERT_PAYLOAD:
    %addr(propBuf2));

// Flush transport so request is sent and receive response.
rc = axiscTransportFlush(tHandle);
if (rc = 0);
    rc = axiscTransportReceive(tHandle:
        %ADDR(response): %SIZE(response): 0);

    dow rc > 0 AND bytesRead < %SIZE(response);
    bytesRead = bytesRead + rc;
        rc = axiscTransportReceive(tHandle:
            %ADDR(response)+bytesRead:
                %SIZE(response)-bytesRead: 0);
    enddo;
endif;
Some Additional Light Reading

Developer Works – 3 Part Series on Rest for IBM i

Special notices

This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Consult your local IBM business contact for information on the IBM offerings available in your area.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquiries, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

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

The information contained in this document has not been submitted to any formal IBM test and is provided "AS IS" with no warranties or guarantees either expressed or implied. All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved. Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client's credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies.

All prices shown are IBM's United States suggested list prices and are subject to change without notice; reseller prices may vary.

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

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.

Revised September 26, 2006