Web Services from the World of REST

Tim Rowe - timmr@us.ibm.com

Agenda

• Introduction to web services
• The API economy
• SOAP vs REST
• Integrated web services for IBM i
• REST support in integrated web services server
• Summary

© 2014 International Business Machines Corporation
What is... ?

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

... service oriented architecture (SOA)?
An IT architectural style that supports integrating your business as linked services

“SOA impacts every aspect of IT and business.”

Services are like musical notes

Each musical notes represents a business service
Checking Credit, Opening Account, Checking Inventory, Placing an Order, Tracking Shipment

Services allows for flexible composition of music
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

---

Why should I be interested in web services?

1. Create wide ranging new business opportunities and added customer value, by exposing services over the Internet. These opportunities include the notions of “software as a service” and utility based computing.

2. Revitalize existing applications and data for use within new, powerful, and integrated business solutions, gaining new insights.

3. Increase developer productivity, simplifying the task of distributed systems development and integration through standards based interoperability.
The big picture

Services – Efficiency in creation, reuse for execution...flexibility for change and growth

People & Applications

Business Process

Services

Operational Systems

Web apps: for people interacting with systems and processes
Dashboards: for people monitoring business processes and system performance

Goal – Increasing people’s productivity and the overall company performance
Goal – Highly flexible and automated business processes
Goal – Reusable services accessible across the enterprise
Goal – Deliver Information as a Service

External

Reuse: Using tools you may be entitled to
Modernize RPG and COBOL Batch Applications

RD Power & RBD: Web Services Wizard (formally WDSc)

Reuse: Use Host Access Transformation Services
Modernize RPG and COBOL [5250] OLTP Applications

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.


---

Reuse: Using IBM i Web Services Environment
Web Service enable RPG and COBOL Source Code

IBM i: Integrated Web Services Server SOAP & REST

Included with IBM i
Agenda

• Introduction to web services
• The API economy
• SOAP vs REST
• Integrated web services for IBM i
• REST support in integrated web services server
• Summary

What is the API economy?

• Cloud, mobile and social are fueling the hyper-growth of API-centric, business as-a-service economies

• Data has considerable value and can be monetized given an easy-to-consume API

• Developers building apps for the iPhone or Android have made APIs a standard requirement for building software that needs to integrate with other apps and services

• Enterprises have discovered how to harness the value of the ecosystem by exposing their “crown jewels” as APIs and letting 3rd party developers engage
The Web beyond the browser – the API economy

Businesses Are Evolving

Web APIs are the new, fast-growing business channel

Not having an API today is like not having a Web Site in the 90s

“$7bn worth of items on eBay through APIs”
Mark Carges (Ebay CTO)

The API which has easily 10 times more traffic then the website, has been really very important to us.”
Biz Stone (Co-founder, Twitter)

“The adoption of Amazon’s Web services is currently driving more network activity then everything Amazon does through their traditional web sites.”
Jeff Bar (Amazon evangelist) / Dion Hinchcliffe (Journalist)

The API-centric business services and processes

Integrated Business Functions
Packaged Applications
Custom Development
Long Project Planning and Development

Business Solutions

API-centric Business Services
As-a-service delivery
Composable Web
Short Innovation Lifecycle

© 2014 International Business Machines Corporation
About APIs

• What is an API?
  – API is a public persona for an enterprise; exposing defined assets, data or services for public consumption
  – API is simple for app developers to use, access and understand
  – API can be easily invoked via a browser, mobile device, etc.

• What “assets, data or services” are exposed via an API?:
  – Product catalogs
  – Phone listings
  – Insurance cases
  – Order status
  – Bank loan rates
  – Etc.

Extending the reach of your SOA

SOA in general is about providing access to the reusable business tasks (services) of your enterprise.

• The way you embrace your business communities is to unleash your assets to the web

• Every enterprise has valuable assets (information and services) locked up in databases, legacy transactions, or third-party software
Companies need to become an engaging enterprise

- Business Users want to engage Customers in new markets
- They need to Externalize the Enterprise
- They need to get Apps in front of these Customers
- Apps need APIs that Externalize the Enterprise
- App Developers use APIs
- App Developers are now External to the Enterprise

Enterprises wants to tap into innovation from a large community of developers, not just developers they employ.

While keeping this in mind...

Who are your developers?
Who can access your information?
Where do transactions happen?

Who are your developers?
Anyone

What is an application?

Who can access your information?
Everyone

Where do transactions happen?

Anything

Everything
Agenda

- Introduction to web services
- The API economy
- SOAP vs REST
- Integrated web services for IBM i
- REST support in integrated web services server
- Summary

SOAP-style web services

- What is it?
  - SOAP is not an acronym
  - SOAP is a protocol for exchanging XML-based messages over computer networks
  - Message being sent must adhere to the protocol
  - Protocol used to implement web services
  - Generally activity-oriented, where service operations are center of attention (resources may be involved but serves only to refine the context within which the activity is performed)

- Key technologies and standards
  - XML
  - SOAP
  - WSDL

© 2014 International Business Machines Corporation
**SOAP Overview**

- SOAP messages have structure
- Uses universal resource identifier (URI) to identify web service
- Uses Web Services Description Language (WSDL), an XML-based language for describing Web services (the contract between client and service provider)
  - Describes endpoints and their messages, regardless of what message formats or network protocols are used to communicate.
  - Complex for the beginner (even the expert), but there are lots of tooling around WSDL that hides the complexity.
  - Basic idea is that there is service defined and operations to invoke.
  - The parameters and responses are fully described and this promotes strong data typing.
- An eco-system built around SOAP and WS-* standards
  - E.g. WS-AtomicTransaction, WS-ReliableMessaging, etc.

**REST-style web services**

- What is it?
  - Acronym that stands for REpresentational State Transfer and is an architectural style or design pattern described in Roy Fielding’s doctoral dissertation, not a standard
  - Resource oriented rather than activity oriented (i.e. everything is a resource and resource is center of attention)
  - Resources interconnected via hyperlinks
  - Operations on resource limited to those provided by HTTP specification (GET, PUT, DELETE, POST, etc.)
  - REST used extensively in Web 2.0 world
- Key technologies and standards
  - HTTP
  - URI
REST Overview

- REST principles:
  - Network of web pages (a virtual state-machine)
  - Client progresses through an application by selecting links (state transitions)
  - This results in the next page (representing the next state of the application) being transferred to the user and rendered for their use
- Uses universal resource identifier (URI) to identify web service (resource)
- Operations on resource limited to those provided by HTTP specification (GET, PUT, DELETE, POST, etc.)

SOAP vs. REST comparison

<table>
<thead>
<tr>
<th></th>
<th>SOAP</th>
<th>REST</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity-oriented</td>
<td>Resource-oriented</td>
<td></td>
</tr>
<tr>
<td>Few endpoints, many custom methods</td>
<td>Many resources, few fixed methods</td>
<td></td>
</tr>
<tr>
<td>Few nouns, many verbs Example: lightbulbmanager.getState(&quot;tim&quot;)</td>
<td>Many nouns, few verbs Example: GET <a href="http://w3/lightbulbs/tim">http://w3/lightbulbs/tim</a></td>
<td></td>
</tr>
<tr>
<td>WSDL is the contract between client and server</td>
<td>No standard, but WADL is used and may become a standard</td>
<td></td>
</tr>
<tr>
<td>Lots of tooling, mainly to generate client stubs in order to communicate with web service and handle parsing/conversion of data</td>
<td>No tooling (application needs to parse the data returned and do any conversions)</td>
<td></td>
</tr>
</tbody>
</table>
SOAP vs. REST example

Is the light bulb currently on?

**SOAP**

Service: LightbulbManagement

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

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

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

**REST**

- GET http://w3/services/lightbulbMgmt
- POST http://w3/services/LightBulbManager HTTP/1.1
  - Host: example.com
  - Content-Type: text/xml; charset=UTF-8
  - SOAPAction: "LightBulbManager#getState"

**SOAP vs. REST example data flows (get information)**

**SOAP request**

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

<?xml version='1.0' encoding='utf-8'?>
<env:Envelope xmlns:env="" xmlns:ns1="">
  <env:Body>
    <ns1:getState xmlns:ns1="">
      <in0 xsi:type="xsd:string">tim</in0>
    </ns1:add>
  </env:Body>
</env:Envelope>

**REST request**

GET http://w3/lightbulbs/tim HTTP/1.1
Host: example.com
Accept: application/xml
What’s the philosophical difference?

• The key motivator of choosing REST-based web services is its simplicity and its ubiquity
  – It’s about delivering content in the simplest possible way
  – HTTP is available everywhere; it’s like the air around us

• With SOAP It’s not the body that matters, it’s the headers
  – WS-Security is about choice in the decision of encryption, identity tokens and digital signatures
  – WS-Addressing is about transport-neutral mechanisms of describing addresses
  – You might want to choose SOAP-based web services where you need qualities of service not easily addressed by HTTP/REST, such as reliable messaging, transactions, and transport independence. Or you need a hardened interface that is exposed to business partners.

SOAP or REST?

• The choice of REST-style versus SOAP-style is nothing more than a choice over a design strategy based on business and application need, but it is a choice that can profoundly impact how an application is used and evolves over time.

• Where both WS-* and REST API offered, REST API more widely used
  – REST APIs often easier to consume, especially with scripting languages
  – Browser-based experimentation also easy

• External web API protocol usage overwhelming REST-based. However, many web APIs implement SOAP and it will continue to be an important building block for enterprise service integration. One of the roles of web APIs and REST in such cases is complementary to the existing services, to allow services to be exposed to external consumers in a manner they can make sense of.
Agenda

- Introduction to web services
- The API economy
- SOAP vs REST
- **Integrated web services for IBM i**
- REST support in integrated web services server
- Summary

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/
Agenda

• Introduction to web services
• The API economy
• SOAP vs REST
• Integrated web services for IBM i
• REST support in integrated web services server
• Summary

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
Demo of SOAP based Methodology

Create web services server


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

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!

New ‘Permissions’ Support integrated!!!
- Now a *USER granted ‘permission’ can use the GUI
- Group profiles are now supported

Web Integration New Features

- 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
Create web services server (cont.)

Step 1: Specify server name.

Following panels created on 7.1 and are the same in 6.1 (assuming latest PTFs are installed), but will differ slightly from 5.4 due to many enhancements not in 5.4

Create web services server (cont.)

Step 2: User Profile for web container.

The server requires as IBM 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 has authority to all of the server's objects, such as files and directories.

Specify user ID for server
- Use default user ID
  - Note: The default server user ID is OWSERVICE
- Specify an existing user ID
- Create a new user ID
Create web services server (cont.)

Step 3: Create the server

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

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

Manage Web Services Server

Web services server created by the Create Web Services Server wizard.

The 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 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/

Install web service (cont.)

Step 1: Select Next to install a new web 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 (‘ILG’), or service program (‘SRV-ROB’). 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

Specify the library and program object for the Web service.  

- Specify IBM 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 ILE object name. This is the fastest and recommended way to locate the program object.

Library name: NVSRI

ILE Object name: SAS01030 tslib

ILE Object type: *SERVPGM* - ILE

Browse the integrated ISPF system for the ILE program object.

Back  Next  Cancel

Install web service (cont.)

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

Specify a unique name for this service.

Service name: FindCustomer

Service description: IPS0401_TRSI
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>FINDCITIES</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>GETFLIGHTSNOV</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>
</tbody>
</table>

- Position: input, char
- LISTTYPE: input, char
- COUNTREQ: input, int
- COUNTRET: output, int
- CUSTLIST: output, int

Step 5: Specify user profile for the web service

Deploy New Service

Specify User ID for this Service - Step 5 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
- 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

Deploy New Service

Specify library list for this 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 window:
- Library name
- MSIR
- fjthg00

Add | Remove | Remove All | Move Up | Move Down | Continue

Back | Next | Cancel

Install web service (cont.)

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

Deploy New Service

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

Back | Next | Cancel
Install web service (cont.)

Step 8: Specify WSDL options for web service

Step 9: Ready to deploy the new web service – Services tab

Deploy New Service

Specify WSDL Options

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

- Generate web service bindings for SOAP 1.1 protocol: Enable
- Generate web service bindings for SOAP 1.2 protocol: Disable
- Generate reliable elements for all fields: Disable
- Generate optional elements for all fields: Disable
- Expose service metadata: Enable

Deploy New Service

Summary - Step 9 of 9

When you click Finish the web service is deployed.

- Name: FindCustomers
- Description: Find Customers
- Service install path: /WebSphere/ixWebServices/services/FindCustomers
- User ID for service: MyUser
- Program: /WebSphere/ixWebServices/services/FindCustomers
- Library list for service: /WebSphere/ixWebServices/services/FindCustomers/FindCustomers

After the service is installed, use the URL to access the Web service definition: http://ibm2u2a4-10086/web/services/FindCustomers?wsdl
Install web service (cont.)

After a few seconds, service is installed and started

Manage the web service

You can view WSDL as long as server is active
Manage the web service (cont.)

View the WSDL file (partial listing below)

```xml
<complexType name="FindCustomer_BDL">
  <complexContent>
    <restriction base="tns:FindCustomerInput"/>
  </complexContent>
</complexType>

<complexType name="FindCustomerXMLResponse">
  <complexContent>
    <restriction base="tns:FindCustomer"
      targetNamespace="http://example.com/findcustomer">
      <sequence>
        <element name="езультат" type="string"/>
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

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

You can view and modify web service properties
Manage the web service (cont.)

Web service properties – General tab

About integrated web services server REST support

- Supported in IBM i 7.1 and 7.2
  - 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

- SOAP
  - A SOAP-based web service is a self-contained software component that exposes 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

REST – Specify Service Name
REST – Select the Export Procedures

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

Do not wrap input parameters

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>LISTTYPE</td>
<td>char</td>
<td>QUERY_PARAM</td>
<td>listtype</td>
<td>NONE</td>
</tr>
<tr>
<td>COUNTREQ</td>
<td>int</td>
<td>QUERY_PARAM</td>
<td>councreq</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.
- 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 program you want to install or update as a Web service may depend upon the Web service programs on your server. If a library list 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]
  - [Library name]
- [Add] [Remove] [Remove All] [Move up] [Move down] [Continue]
REST – Transport Information

REST - Finish
A very, very simple example

- Temperature conversion (Fahrenheit to Celsius and vice-versa)

- RPG program has packaged in a service program with 2 procedures:
  - ToCelsius
  -ToFahrenheit

- URIs will be of the following forms:
  - /temperatureService/ftoc/{temp}
  - /temperatureService/ctof/{temp}

- Only GET HTTP method needed

- Response should be JSON format

RPG code for example REST service - ToCelsius

```rpg
h nomain PGMINFO(*PCML:*MODULE)
  d ToCelsius pr
  d tempIn 8F const
  d tempOut 8F
  p ToCelsius b export
  d ToCelsius pi
  d tempIn 8F const
  d tempOut 8F
  /free
  tempOut = (5/9)*(tempIn - 32);
  /end-free
  p ToCelsius e
```

© 2014 International Business Machines Corporation
RPG code for example REST service - ToFahrenheit

```
h nomain PGMINFO(*PCML:*MODULE)
d ToFahrenheit   pr
  d  tempIn      8F   const
  d  tempOut     8F
  p  ToFahrenheit b       export
  d  ToFahrenheit pi
  d  tempIn      8F   const
  d  tempOut     8F
  /free
  tempOut = ((9/5)*tempIn) + 32;
  /end-free
  p  ToFahrenheit  e
```

Web Administration GUI deployment of web service

The following panels that will be shown is not the actual panel you would see when deploying a web service. For example, navigation bar and tabs are not shown. This was done in order to show as much of the panel related to deploying a web service.
Deploy New Service – Step 1

WSERVICE3 > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify Web service type - Step 1 of 9

Welcome to the Deploy New Service wizard. The wizard helps you externalize a

Specify Web service type:

- SOAP
- REST

NEW: Choice of SOAP or REST

A REST-based Web service exposes resources, where client requests to messages that are exchanged is defined by the resource itself.

Deploy New Service – Step 2

WSERVICE3 > Manage Deployed Services > Deploy New Service

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 I located on the system.

Specify the 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 name of the program object. This is the fastest and recommended wa

Library name: AMRA
ILE Object name: TEMPCONV
Deploy New Service – Step 3

**WSERVICE3 - V2.6 (web services)**

**WSERVICE3** > **Manage Deployed Services** > **Deploy New Service**

**Deploy New Service**

*Specify Name for Service - Step 3 of 9*

The Web service to be externalized is a resource. The URI path to it is relative to the context root and can be a simple string or restrict what is allowed.

- **Resource name:** temperature
- **Service description:** TEMPCNV
- **URI path template:** /temperature

**NEW:** The URI so far is `/temperature/`

---

Deploy New Service – Step 4

**WSERVICE3 - V2.6 (web services)**

**NEW:** You can change case of parameters

**Export procedures:**

<table>
<thead>
<tr>
<th>Select</th>
<th>Procedure name/Parameter name</th>
<th>Usage</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>TOFAHRENHEIT</td>
<td></td>
<td>float</td>
</tr>
<tr>
<td></td>
<td>TEMPIN</td>
<td>input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TEMPOUT</td>
<td>output</td>
<td></td>
</tr>
<tr>
<td>✓</td>
<td>TOCELSIUS</td>
<td></td>
<td>float</td>
</tr>
<tr>
<td></td>
<td>TEMPIN</td>
<td>input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TEMPOUT</td>
<td>output</td>
<td></td>
</tr>
</tbody>
</table>

**NEW:** You can change case of parameters
### Deploy New Service – Step 5

**WSERVICE3 - V2.6 (web services)**

**Specify Resource Method Information - Step 5 of 9**

Specify resource method information.

**Procedure name:** TOFAHRENHEIT

**URI path template for resource:** /

**HTTP request method:** GET

**URI path template for method:** /ctof/{temp}

**Allowed input media types:** *ALL

**Returned output media types:** *JSON

**Output parameter to designate as HTTP response code:** *NONE

**Output parameter to designate as HTTP header array:** *NONE

**Input parameter mappings:**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Data type</th>
<th>Annotation</th>
<th>Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEMPIN</td>
<td>float</td>
<td>*PATH_PARAM</td>
<td>[temp]</td>
</tr>
</tbody>
</table>

**Panel will be shown for each procedure**

**NEW Panel**

URI path for template modified with identifier

Value of TEMPIN will be path segment represented by “temp” in URI path template for method.
Deploy New Service – Step 6

WSERVICE3 - V2.6 (web services)

WSERVICE3 > Manage Deployed Services > Deploy New Service

Deploy New Service

Specify User ID for this Service - Step 6 of 9

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

Specify User ID for this Service:

- Use server’s user ID
  - The server’s user ID must have the necessary authority to this pr
- Specify an existing user ID

Deploy New Service – Step 7

WSERVICE3 - V2.6 (web services)

WSERVICE3 > Manage Deployed Services > Deploy New Service

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 needs to exist in a different library list.

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
- AMRA
Deploy New Service – Step 8

Deploy New Service

Specify Transport Information to Be Passed - Step 8 of 9

Specify transport information to be passed to the web service.

Information to be passed to web service implementation

Specify Transport Metadata:

<table>
<thead>
<tr>
<th>Transport Metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>REMOTE_ADDR</td>
</tr>
</tbody>
</table>

Specify HTTP Headers:

<table>
<thead>
<tr>
<th>HTTP Headers</th>
</tr>
</thead>
<tbody>
<tr>
<td>There are no entries for this table.</td>
</tr>
</tbody>
</table>

Deploy New Service – Step 9

Deploy New Service

Summary - Step 9 of 9

When you click Finish the web service is deployed.

<table>
<thead>
<tr>
<th>Service</th>
<th>Method</th>
<th>Request Information</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Name: temperature
Description: TEMPCONV
User ID for service: *SERVER (QWSERVICE)
Program: /QSYS.LIB/AMRA.LIB/TEMPCONV.SRVPGM

© 2014 International Business Machines Corporation
Results!

<table>
<thead>
<tr>
<th>Method</th>
<th>Endpoint</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td><a href="http://ip103st27.rch.stglabs.ibm.com:10022">http://ip103st27.rch.stglabs.ibm.com:10022</a></td>
<td>/web/services/temperatureService/fcToC/900</td>
</tr>
</tbody>
</table>

1. The first part of the resource path is the context root, and can be modified on a server basis (i.e. all services will have this context root).
2. The second part is the resource name, which is always appended with “Service”.
3. The last part maps to the URI path template specified for ToCelsius request method, so the procedure gets called to service the request. Note that the 900 is automatically converted to a float!

Agenda

- Introduction to web services
- The API economy
- SOAP vs REST
- Integrated web services for IBM i
- REST support in integrated web services server
- Summary
Summary

• Getting started with web services on IBM i is easier than ever with integrated web services for i. The bottom line is that flexible businesses requires flexible IT, and the path to flexible IT is web services.

• REST is here to stay (at least until the next best thing comes along) and is primarily driven by the API economy, which is fueled by cloud, mobile and social.

• We always welcome and appreciate feedback!

Questions and Answers
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 10504-1755 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.

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

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.

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

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

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

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

rPerf for AIX

rPerf (Relative Performance) is an estimate of commercial processing performance relative to other IBM UNIX systems. It is derived from an IBM analytical model which uses characteristics from IBM internal workloads, TPC and SPEC benchmarks. The rPerf model is not intended to represent any specific public benchmark results and should not be reasonably used in that way. The model simulates some of the system operations such as CPU, cache and memory. However, the model does not simulate disk or network I/O operations.

- rPerf estimates are calculated based on systems with the latest levels of AIX and other pertinent software at the time of system announcement. Actual performance will vary based on application and configuration specifics. The IBM eServer pSeries 640 is the baseline reference system and has a value of 1.0. Although rPerf may be used to approximate relative IBM UNIX commercial processing performance, actual system performance may vary and is dependent upon many factors including system hardware configuration and software design and configuration. Note that the rPerf methodology used for the POWER6 systems is identical to that used for the POWER5 systems. Variations in incremental system performance may be observed in commercial workloads due to changes in the underlying system architecture.

All performance estimates are provided "AS IS" and no warranties or guarantees are expressed or implied by IBM. Buyers should consult other sources of information, including system benchmarks, and application sizing guides to evaluate the performance of a system they are considering buying. For additional information about rPerf, contact your local IBM office or IBM authorized reseller.

CPW for IBM i

Commercial Processing Workload (CPW) is a relative measure of performance of processors running the IBM i operating system. Performance in customer environments may vary. The value is based on maximum configurations. More performance information is available in the Performance Capabilities Reference at: www.ibm.com/systems/i/solutions/perfmgmt/resource.html

Revised April 2, 2007