



IBM Software Group

CICS Web Services Part 1: Development

Paul Cooper

Software Engineer; CICS Development, IBM Hursley



WebSphere® Support Technical Exchange



Agenda

- Overview of CICS Web services in CICS TS 3.1, 3.2 and 4.1
- Learn about the development, deployment, testing and debugging of CICS Web services
- Learn about the customisation opportunities
- Things to watch out for....



Useful Resources

- CICS Information Centers

TS 3.1	http://publib.boulder.ibm.com/infocenter/cicsts/v3r1/index.jsp
TS 3.2	http://publib.boulder.ibm.com/infocenter/cicsts/v3r2/index.jsp
TS 4.1	http://publib.boulder.ibm.com/infocenter/cicsts/v4r1/index.jsp

- IBM® Web Services Red Books

Architecture	http://www.redbooks.ibm.com/abstracts/sg245466.html?Open
Implementation	http://www.redbooks.ibm.com/abstracts/sg247206.html?Open
Performance	http://www.redbooks.ibm.com/abstracts/sg247687.html?Open
Security	http://www.redbooks.ibm.com/abstracts/sg247658.html?Open
WLM	http://www.redbooks.ibm.com/abstracts/sg247144.html?Open
Development	http://www.redbooks.ibm.com/abstracts/sg247126.html?Open

- Examples <http://www-01.ibm.com/support/docview.wss?uid=swg24020774>

- Knowledge Collection <http://www-01.ibm.com/support/docview.wss?uid=swg27010507>



Application Development 'Bottom Up'

- Start with an existing CICS application
 - COMMAREA described with language structures
 - COBOL, PL/I, C or C++
 - **DFHLS2WS** generates:
WSDL and **WSBind** file
- Coverage of data types is not 100%, e.g.:
 - Pointers
 - OCCURS DEPENDING ON
 - Level 66
 - Limited support for PICTURE

```
04  singleChar PICTURE X(1).
04  singleDouble COMP-2 SYNC.
04  singleChar2 PICTURE X(1).
04  singleDouble2 COMPUTATIONAL-2.
04  singleFloat COMP-1.
04  singleFloat2 COMPUTATIONAL-1 SYNC.
04  floatArray COMP-1 OCCURS 5.
04  structure.
07      filler PIC X(1).
07      fieldA PIC X(10).
07      substruc.
09          fieldB PIC S9(10) SIGN LEADING.
09          fieldC PIC X(1).
07          fieldD PIC X(1).
04  fieldE PIC S9(10) SIGN LEADING.
04  struc2.
05      struc3.
06          fieldF PIC X(1).
05          fieldG PIC X(1).
04  fieldH PIC X(1).
```

WSDL generated by DFHLS2WS

An example WSDL fragment:

```
<?xml version="1.0" ?>
<!--
  This document was generated using 'DFHLS2WS' at mapping level '1'
-->
<definitions targetNamespace="http://www.NULLPROG.testSup.com"
  xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:reqns="http://www.NULLPROG.testSup.com" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://www.NULLPROG.testSup.Request.com"
  <types>
    <xsd:schema attributeFormDefault="qualified"
      elementFormDefault="qualified" targetNamespace="http://www.NULLPROG.testSup.Request.com"
      xmlns:tns="http://www.NULLPROG.testSup.Request.com" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      <xsd:annotation>
        <xsd:documentation
          source="http://www.ibm.com/software/http/cics/annota
        </xsd:annotation>
        <xsd:annotation>
          <xsd:appinfo source="http://www.ibm.com/software/http/ci
com.ibm.cics.wsdl.properties.mappingLevel=2</xsd:appinfo>
        </xsd:annotation>
        <xsd:complexType abstract="false" block="#all" final="#all"
          mixed="false" name="ProgramInterface">
          <xsd:sequence>
            <xsd:element name="singleChar" nillable="false">
              <xsd:simpleType>
                <xsd:annotation>
                  <xsd:appinfo source="http://www.ibm.com,
com.ibm.cics.wsdl.properties.charlength=fixed
com.ibm.cics.wsdl.properties.synchronized=false</xsd:appinfo>
                </xsd:annotation>
                <xsd:restriction base="xsd:string">
                  <xsd:maxLength value="1" />
                  <xsd:whiteSpace value="preserve" />
                </xsd:restriction>
              </xsd:simpleType>
            </xsd:element>
            <xsd:element name="singleDouble" nillable="false">
              <xsd:simpleType>
                <xsd:restriction base="xsd:double" />
              </xsd:simpleType>
            </xsd:element>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:schema>
    </types>
  </definitions>
```

An example XML schema fragment from within the WSDL:

```
</xsd:simpleType>
</xsd:element>
<xsd:element name="singleFloat2" nillable="false">
  <xsd:simpleType>
    <xsd:restriction base="xsd:float" />
  </xsd:simpleType>
</xsd:element>
<xsd:element maxOccurs="5" minOccurs="5" name="floatArray"
```

The WSDL contains:

A programmatic description of the service
 A transport specific binding for the service
 The location of the service (a URI)

Using JCL to invoke DFHLS2WS

- Can be integrated into existing build systems

- Requires Java™ to be installed

- Can be archived for future use

```
//JAVAPROG EXEC DFHLS2WS,  
// JAVADIR='java50_31s/J5.0',  
// USSDIR='r650'  
//INPUT.SYSUT1 DD *  
MAPPING-LEVEL=2.2  
LOGFILE=/u/p9coopr/wsd1/l1s2ws.log  
WSDL=/u/p9coopr/wsd1/generated.wsd1  
PGMNAME=NULLPROG  
URI=/testing  
PGMINT=COMMAREA  
LANG=COBOL  
WSBIND=/u/p9coopr/mybindfile.wsbind  
PDSLIB=//P9COOPR.COBOL.LIBRARY  
REQMEM=TMP01  
RESPMEM=TMP01
```



Other 'Bottom Up' Notes

- Applications are:
 - Commarea based (but may use a Channel with a single Container)
(*can be channel based in CICS TS 4.1*)
 - Provider mode (CICS is the server, not the client)
 - For best results use Threadsafe applications
- **IBM Rational Application Developer for System Z (RD/z)** provides a rich interface with two 'bottom-up' modes:
 - Interpreted (DFHLS2WS under the covers)
 - Simple deployment model
 - Compiled
 - Better support for COBOL (e.g. OCCURS DEPENDING ON)
 - More user control for the mappings
- ▶ Both involve a WSBind file and have similar performance
- ▶ Similar mappings, but **not** identical (so can't be hot-swapped)

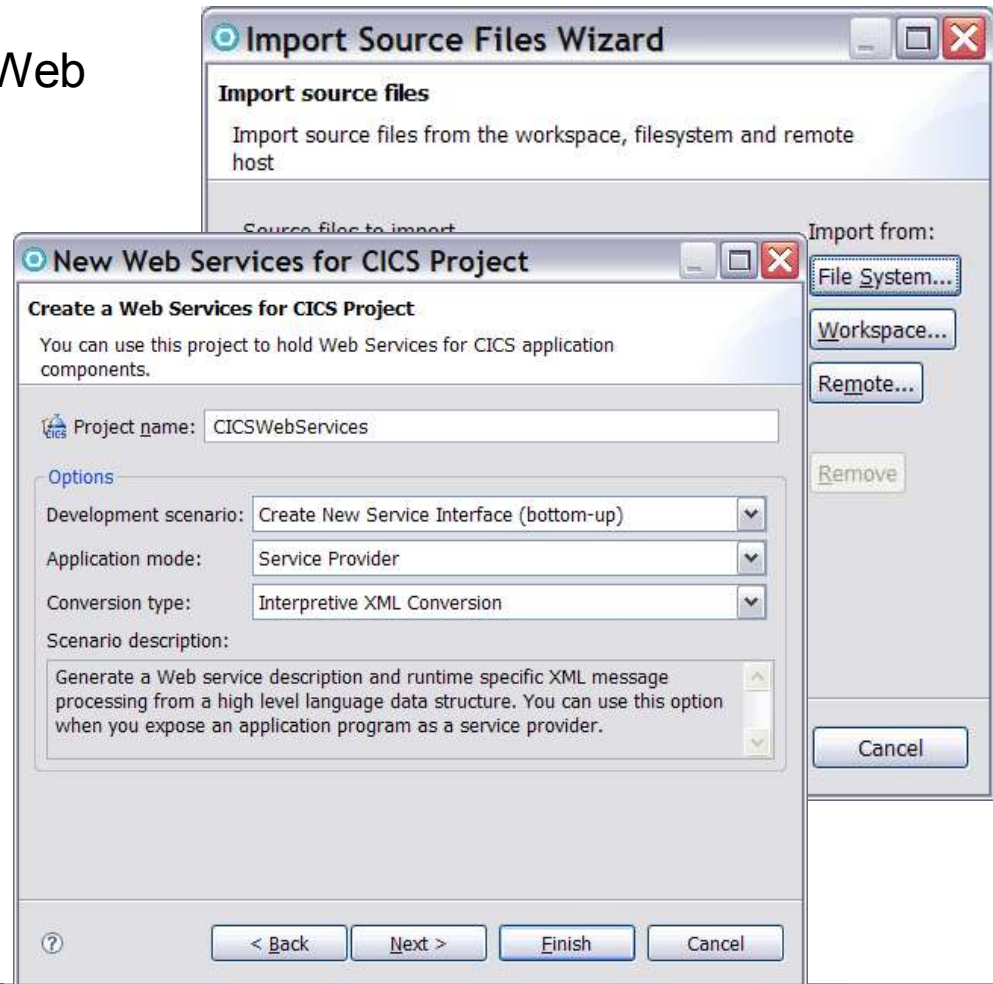


Using RD/z 7.5 to expose a Web Service

Use the wizard to create a new Web Services for CICS Project

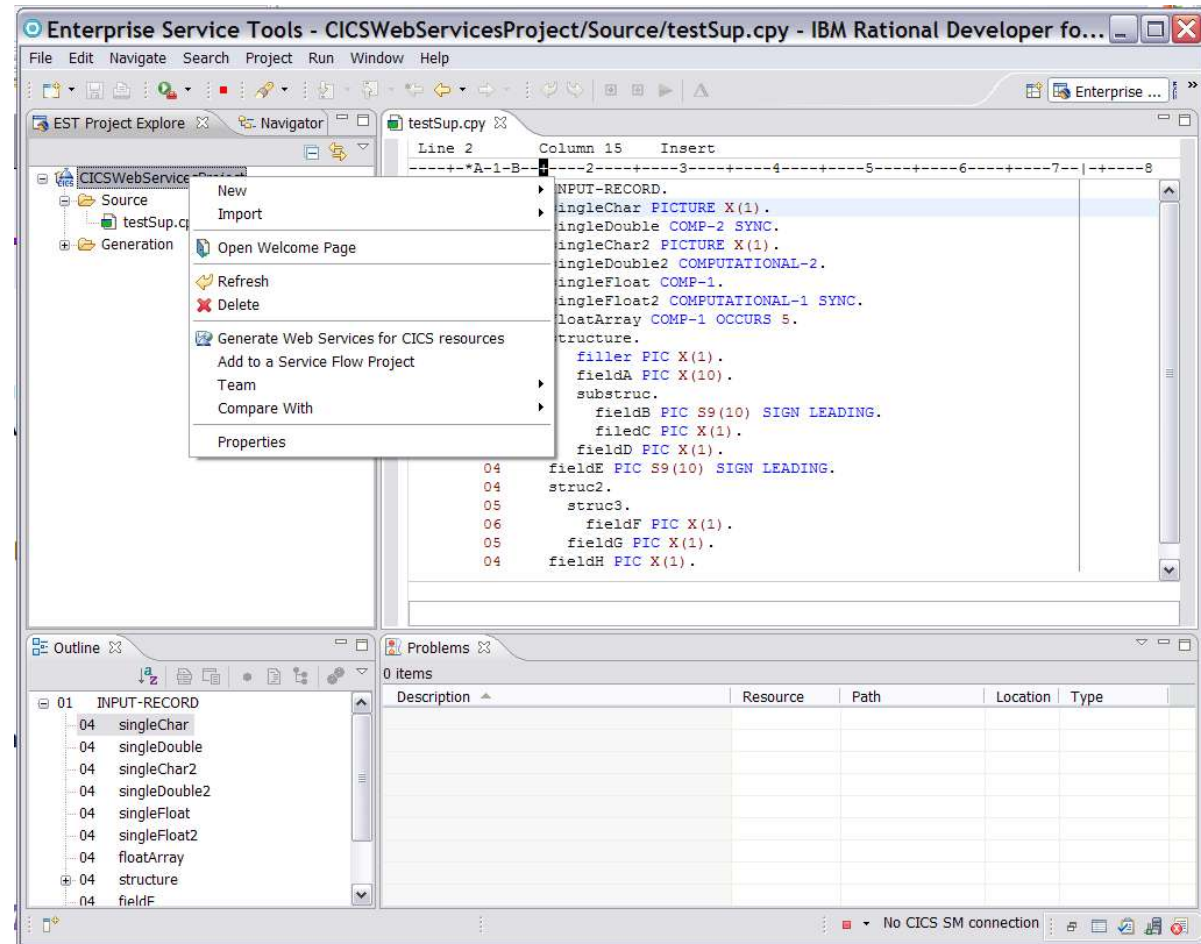
The supported scenarios are:
bottom-up
top-down
meet-in-middle

Input files can be local
or remote



Using RD/z 7.5 to expose a Web Service (2)

Generate the
Web Services
for CICS
resources

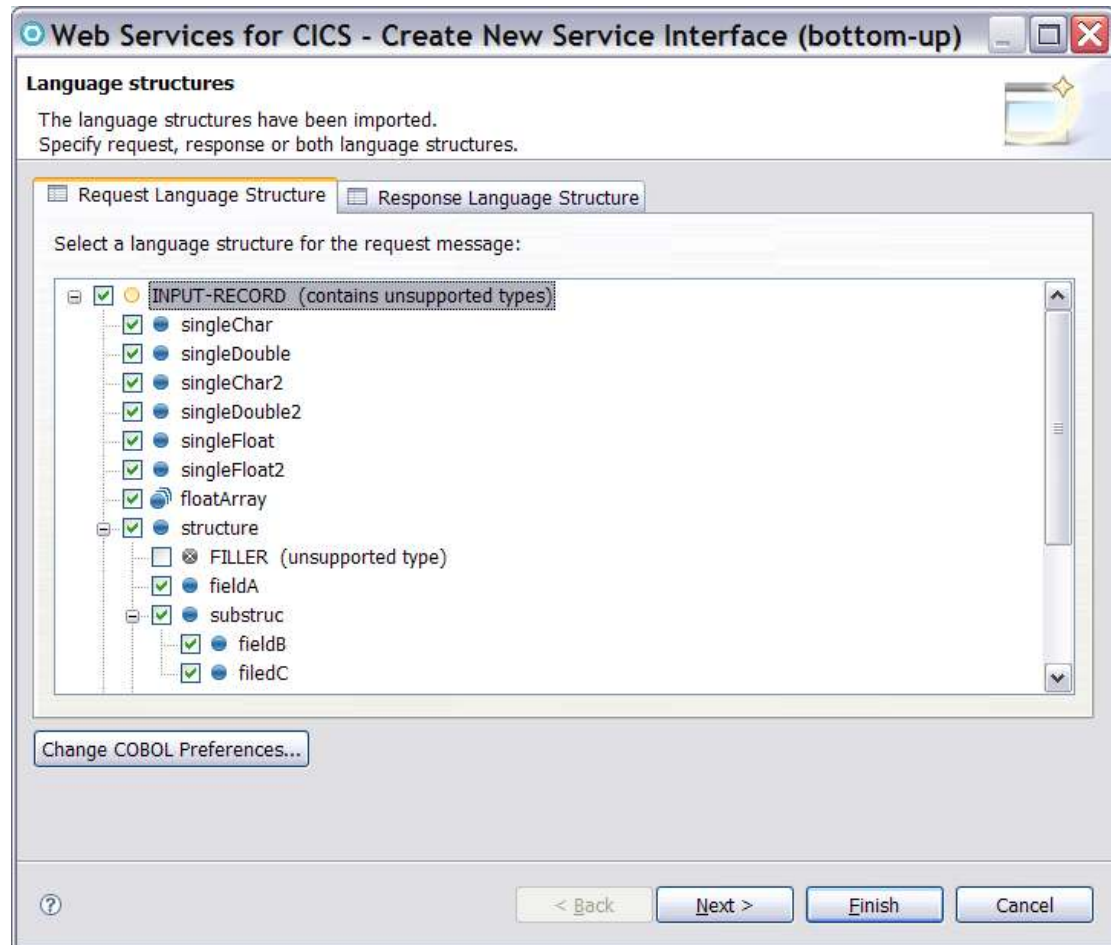


Using RD/z 7.5 to expose a Web Service (3)

Select which structures and fields to include in the Service.

Some fields may be request only whilst others are response only.

Note: using DFHLS2WS from JCL doesn't give you the option of omitting fields.

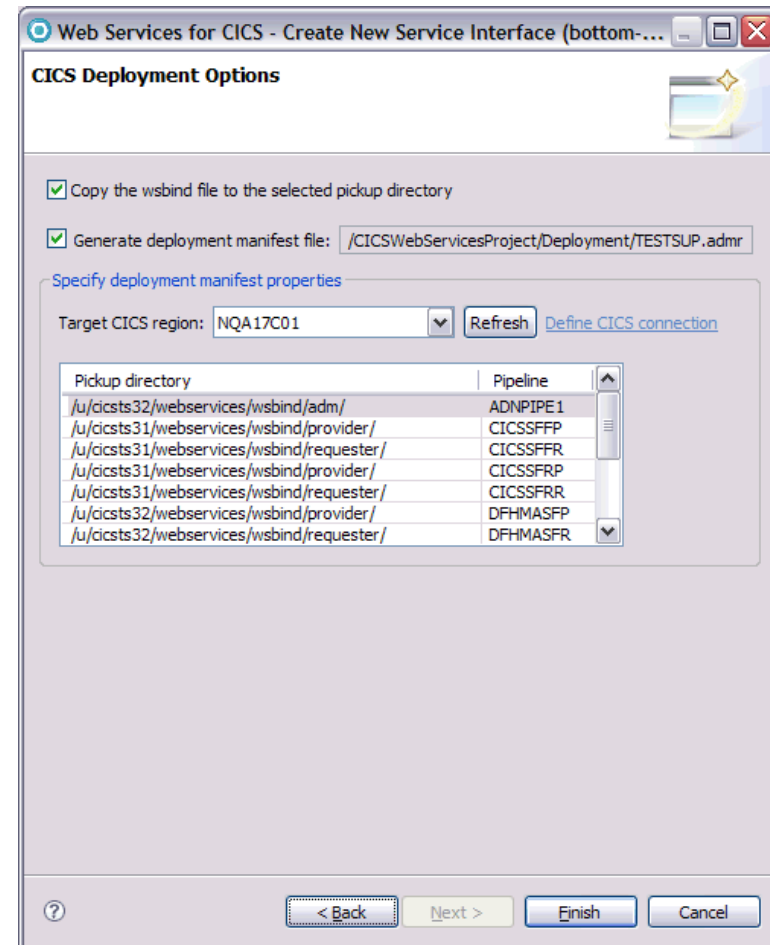


Using RD/z 7.5 to expose a Web Service (4)

And deploy the generated artefacts to CICS.

You can select a specific CICS region to deploy to, and a specific PIPELINE resource within that region.

Uses the Application Deployment Manager (ADM) to update a live CICS region.

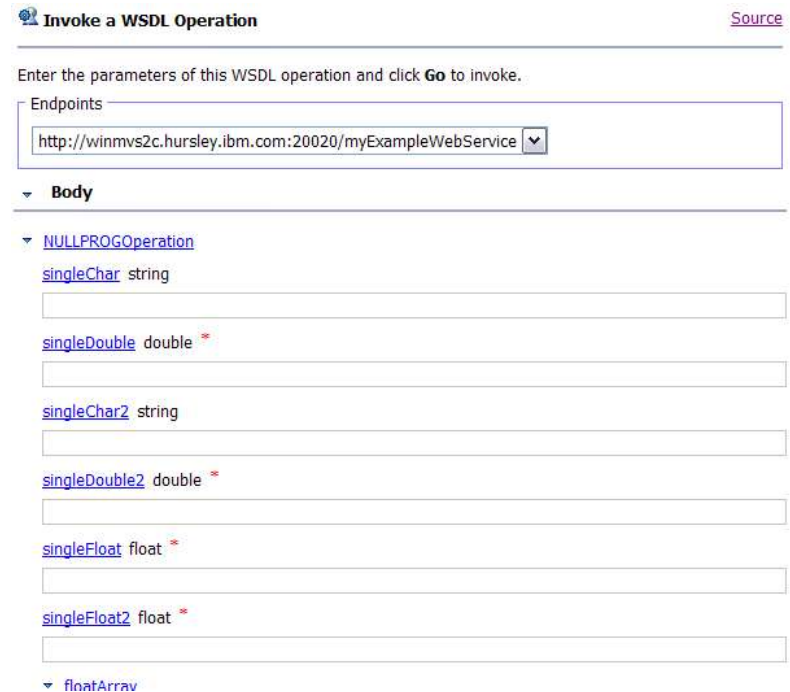


Testing the Web Service

- Many tools exist:
 - ▶ RD/z
 - ▶ Eclipse
 - ▶ Other vendors too.

I generated a test program for the Web service in Eclipse with only a few clicks of my mouse! The form opposite allows the values for the input fields to be set. For more details see:

<http://www-01.ibm.com/support/docview.wss?rs=0&uid=swg21268824>



The screenshot shows a web interface titled "Invoke a WSDL Operation" with a "Source" link. Below the title, it says "Enter the parameters of this WSDL operation and click **Go** to invoke." There is a section for "Endpoints" with a dropdown menu showing "http://winmvs2c.hursley.ibm.com:20020/myExampleWebService". Below this is a section for "Body" with a dropdown menu showing "NULLPROGOperation". Under "NULLPROGOperation", there are several input fields for parameters: "singleChar" (string), "singleDouble" (double), "singleChar2" (string), "singleDouble2" (double), "singleFloat" (float), and "singleFloat2" (float). Each field has a red asterisk indicating it is required. At the bottom, there is a dropdown menu for "floatArrav".

Testing the Web Service (2)

- A SOAP message is sent to CICS
- CICS:
 - Receives the SOAP
 - Converts it to application data
 - Links to the application
 - Converts the response data back into SOAP
 - Sends the SOAP back to the client
- Single day Web service enablement
.... Hopefully....

</soapenv:Header>
<soapenv:Body>

[Browse...](#) [Load](#) [Save As...](#)

```
<q0:NULLPROGOperation>  
  <q0:singleChar>A</q0:singleChar>  
  <q0:singleDouble>0.6789</q0:singleDouble>  
  <q0:singleChar2>B</q0:singleChar2>  
  <q0:singleDouble2>23</q0:singleDouble2>  
  <q0:singleFloat>4567</q0:singleFloat>  
  <q0:singleFloat2>890</q0:singleFloat2>  
  <q0:floatArray>  
    <q0:floatArray>1</q0:floatArray>  
  </q0:floatArray>  
  <q0:floatArray>  
    <q0:floatArray>2</q0:floatArray>  
  </q0:floatArray>  
  <q0:floatArray>  
    <q0:floatArray>3</q0:floatArray>  
  </q0:floatArray>  
  <q0:floatArray>  
    <q0:floatArray>4</q0:floatArray>  
  </q0:floatArray>  
</q0:NULLPROGOperation>
```

</soapenv:Body>
</soapenv:Envelope>

[Go](#) [Reset](#)

Debugging

- When something goes wrong:
 - ▶ A SOAP Fault message is sent back to the requester (client)
 - ▶ DFHPIxxxx messages are written to MSGUSR
 - ▶ Exception trace points are written

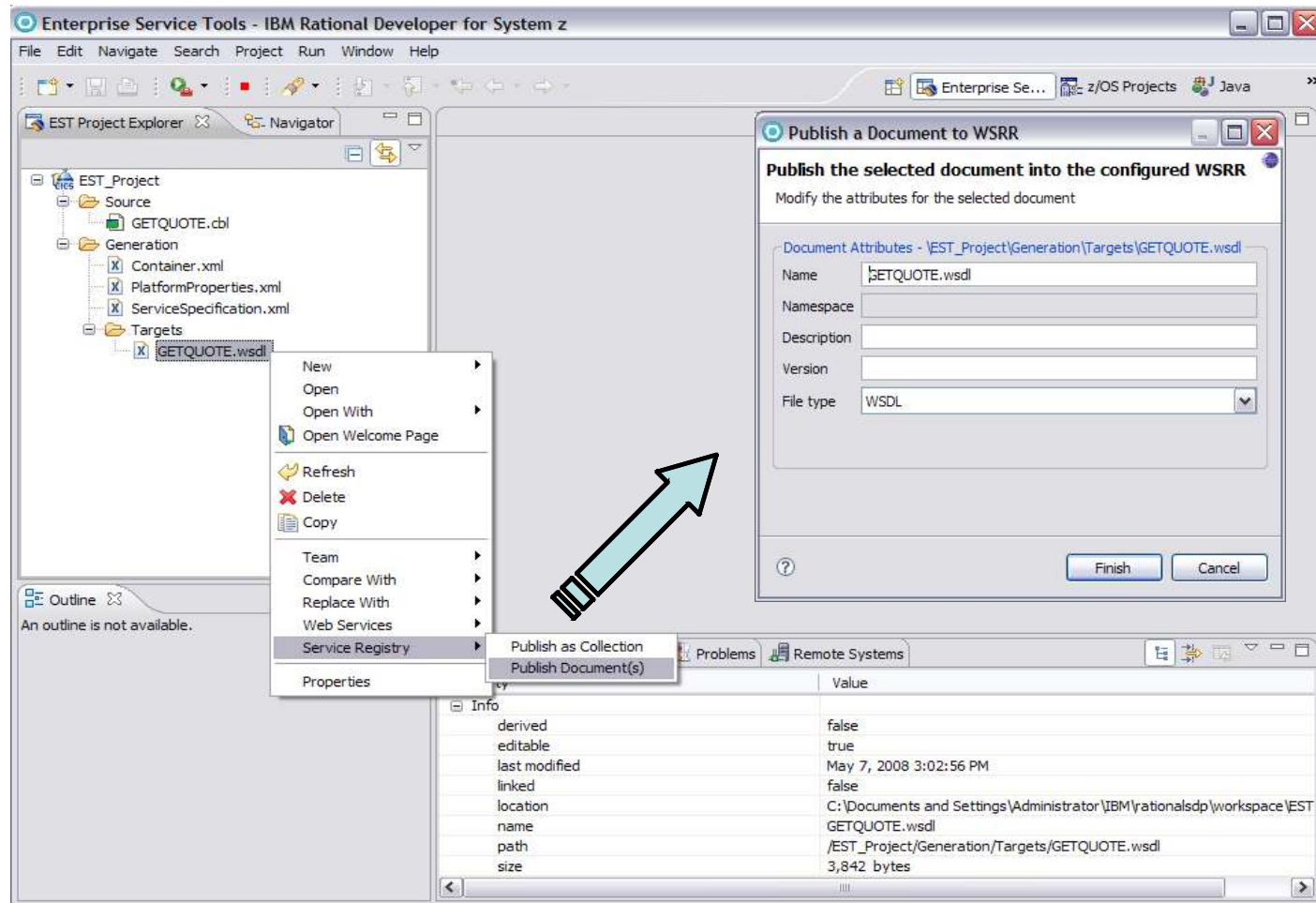
- WEBSERVICE validation may be enabled in CICS
 - ▶ Performs validation of the SOAP Body
 - ▶ Requires Java to be enabled in CICS (zAAP-able)
 - ▶ Very useful during application development



Publishing WSDL using RD/z 7.5

You can publish WSDL from RD/z into the Web services registry (IBM WebSphere Service Registry and Repository – WSRR)

See also SupportPac CA1N for publishing from JCL (this is integrated with DFHLS2WS in TS 4.1)



Should generated WSDL be published?

■ Probably not! (*but you can if you want to*)

▶ Customise it first in order to:

- Have meaningful field names or alternative data types
- Your choice of namespaces, operation names, service names, etc.
- Remove unwanted content (such as annotations or unused fields)
- Combine multiple generated WSDLs into one composite Service
- Plan for interface evolution (add version numbers or similar)
- Generally, to **have the external interface you want it to have**

▶ Then reprocess the customised generated WSDL using the top-down tooling

- This may require a 'wrapper' program to be written (aka meet-in-the-middle)
- The WSDL becomes the source of the interface (not the original copybooks)

▶ **This approach takes a lot more EFFORT, but you get much better RESULTS!**



Service Granularity

- Care should be taken to only expose appropriate PROGRAMs as Web services
 - ▶ Services should be at application boundaries (an application may be made up of many different PROGRAMs)
 - ▶ Consider using the Service Flow Modeller in RD/z to aggregate multiple CICS PROGRAMs into a single 'micro-flow' Web service
 - Including Terminal based interactions (BMS)
 - This minimises the network interactions
 - Further orchestration can be done with **IBM WebSphere Process Server (WPS)** or similar



Mapping and Runtime levels

- The Runtime Level is the minimum version of the CICS runtime the WSBInd file can be deployed into
- The Mapping Level is the version of the programmatic interface shared between the application and CICS
 - ▶ Usually the same as the runtime level
 - ▶ Allows old WSBInd files to be regenerated at a newer runtime level in order to opt-in to some new capabilities, but without requiring application changes.
- New capabilities are implemented at new mapping/runtime levels. **Use the most recent mapping level available for new applications.**

Mapping Level 1.0 – Original CICS TS 3.1 capabilities

Mapping Levels 1.1/1.2 – Enhancements added by APARs PK15904 and PK23547

Mapping Level 2.0 – Original CICS TS 3.2 capabilities

Mapping Levels 2.1/2.2 – Enhancements added by APARs PK59794 and PK69738

Mapping Level 3.0 – CICS TS 4.1 capabilities



Application Development 'Top Down'

- Start with WSDL
- Either **Provider** or **Requester** mode can be enabled
- **DFHWS2LS** generates:
 - ▶ language structures
 - ▶ WSBind file

```
<?xml version="1.0"?>
<definitions name="lengthTests"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:s0="http://test.org/"
  targetNamespace="http://test.org/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <types>
    <s:schema targetNamespace="http://test.org/"
      xmlns:s0="http://test.org/"
      xmlns:s="http://www.w3.org/2001/XMLSchema">
      <s:complexType name="customerType">
        <s:sequence>
          <s:element name="accountNumber" type="s:int" />
          <s:element name="name" type="s:stringy"/>
          <s:any/>
        </s:sequence>
      </s:complexType>
    </s:schema>
  </types>
```

Good editors can validate your WSDL. I validated my WSDL in Eclipse and it highlighted a problem that must be fixed. It's well worth validating your WSDL prior to processing it with DFHWS2LS.

Application Development Challenges

- **Complex WSDL** leads to **very complex COBOL**, so take care!
 - Just because you have a WSDL description of a service doesn't mean that you can easily call or implement it in COBOL. Keep it simple!
- An application program must be written to call or implement the service
 - Using the generated language structures
 - Comments are generated into the language structures
 - RD/z will generate some template code to get you started
 - Interpreted mode only (there is no compiled mode 'top down')
 - Many options exist in DFHWS2LS to tweak the data mapping



Performance Considerations

- Complexity will cost you!
 - ▶ Sending large volumes of data is expensive
(CICS is optimised for 32K even though you can send much more)
 - ▶ CPU costs increase with the number of XML tags used
(regardless of the length of data sent)

See also: <http://www.redbooks.ibm.com/abstracts/redp4344.html?Open> for a Red Paper that discusses a real CICS Web services application with real performance characteristics.



Requester Mode (top down)

- EXEC CICS INVOKE WEBSERVICE
 - ▶ Also EXEC CICS INVOKE **SERVICE** in TS 4.1
- Select the WSDL Operation(s) to enable
 - ▶ Avoid generating meta-data and language structures for unused operations
 - ▶ Only available for TS 3.2 and 4.1 (and TS 3.1 if using RD/z)
- Time-out:
 - ▶ DTIMOUT (TS 3.1)
 - ▶ Per PIPELINE (TS 3.2)
 - Or per request using the DFHWS-RESPWAIT container



Requester Mode (top down) (2)

- **URI is in the application or in the WSBind file**
 - ▶ Could be supplied in a handler program, e.g. using information from WSRR
 - ▶ Could come from a URIMAP using the INQUIRE URIMAP spi command
 - ▶ In TS 4.1 a client mode URIMAP may be named on the INVOKE WEBSERVICE command instead

- **SSL credentials**
 - ▶ In TS 3.1: CICS uses the default certificate for the region
 - ▶ In TS 3.2: CICS looks for a 'client' mode URIMAP that matches the URI
 - If found, that URIMAP is used (including SSL parameters)
 - Otherwise the default certificate for the region is used
 - ▶ In TS 4.1 a client mode URIMAP may be used



Provider Mode (top down)

- Similar to the bottom-up approach, except that a new application has to be written to implement the service
- Takes as input a Channel with Containers
 - ▶ The DFHWS-OPERATION container indicates which Operation is being called.
- The EXEC CICS SOAPFAULT api may be used to create application specific FAULT responses
 - ▶ For SOAP aware applications
 - ▶ ABENDs are turned into SOAP Fault messages by CICS



XML aware applications

- You can write CICS applications that work directly with the XML
 - ▶ Custom application handler program (provider mode)
 - ▶ EXEC linkable pipeline program – DFHPIRT (requester mode)
 - ▶ XML-ONLY in DFHWS2LS (CICS TS 3.2)
 - you get a WSBind file that tells CICS not to do any conversions
 - shared deployment model, support for validation, monitoring, INVOKE WEBSERVICE, etc..
 - the application populates/parses the DFHWS-BODY container.
 - ▶ XML parsing / generation can be done using Enterprise COBOL
 - Or using converter programs generated in RD/z
 - Or other vendor products
 - Or in Java with JAX-B, etc.
 - Or using EXEC CICS TRANSFORM (CICS TS 4.1 api for XML)



WSDL unsupported by DFHWS2LS

- Some WSDL is still unsupported by DFHWS2LS
 - ▶ Validate the WSDL and try again with the best mapping level available
 - ▶ Unsupported constructs include:
 - 'SOAP encoding'; 'minOccurs' and 'maxOccurs' on xsd:sequences
 - Recursion
- Other options:
 - ▶ Work directly with the XML
 - ▶ Modify a local copy of the WSDL (E.g. replace problematic pieces with xsd:any)
 - ▶ Introduce a middle-tier such as **IBM WebSphere Enterprise Service Bus (WESB)** that can support a simple interface for CICS and the full interface for the outside world
 - ▶ Use a different transformation technology such as **IBM WebSphere Transformation Extender (WTX)** or **IBM WebSphere DataPower**



CICS resources involved (provider mode)

WEBSERVICE

Identifies application specific processing

PIPELINE

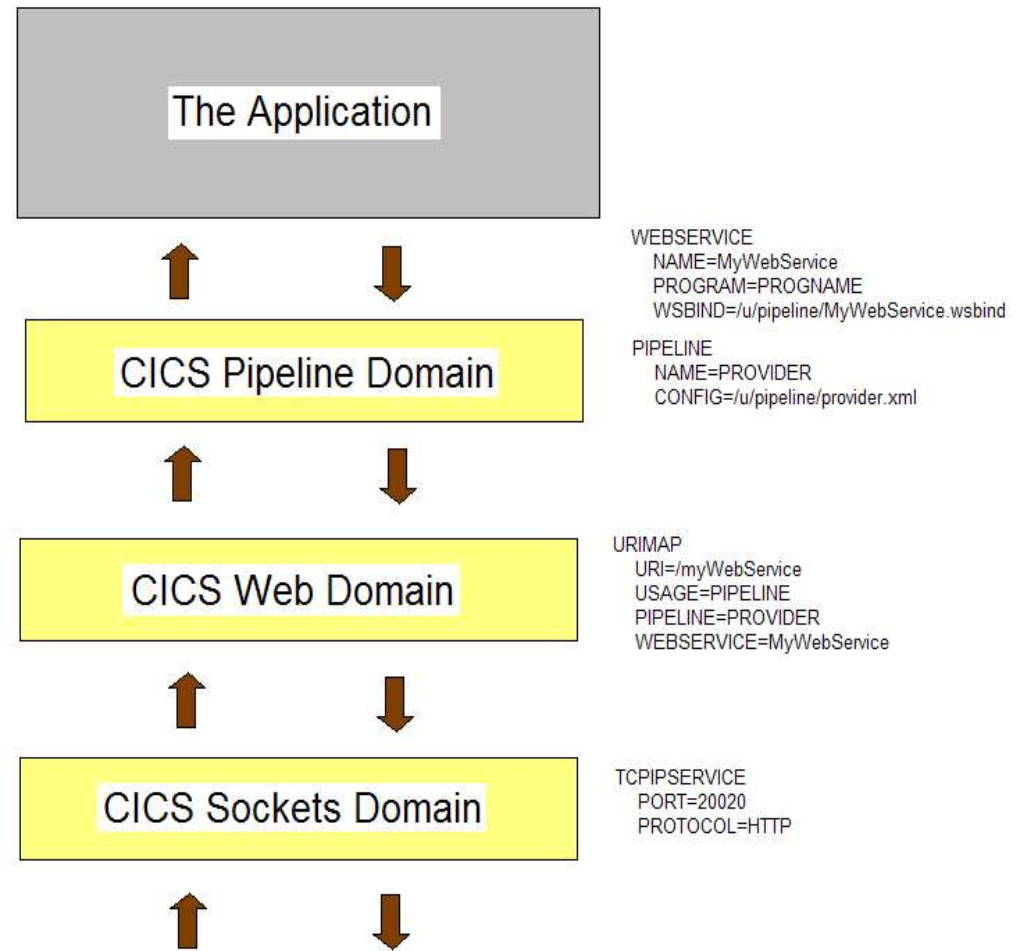
Identifies shared qualities of service

URIMAP

Identifies the type of processing required

TCPIPService

The listener process (if HTTP is used)



Web Service deployment

- The **WSBind** file contains:
 - ▶ meta-data for the runtime data conversions; deployment information
 - The URI of the service; And in provider mode:
 - The name of the PROGRAM to link to
 - Optionally the USERID and TRANSID to execute under
 - Can be viewed, edited and deployed from within RD/z
- For each Web service you will normally need a **WEBSERVICE** resource and a **URIMAP** resource
 - ▶ PIPELINE 'SCAN' will install a set of WEBSERVICE resources from WSBind files
 - ▶ If URIs are specified in the WSBind files then URIMAP resources are installed too (provider mode only)
 - ▶ You can define the WEBSERVICE and/or URIMAP resources via the CSD instead (*which may be useful for peace of mind in production regions*)

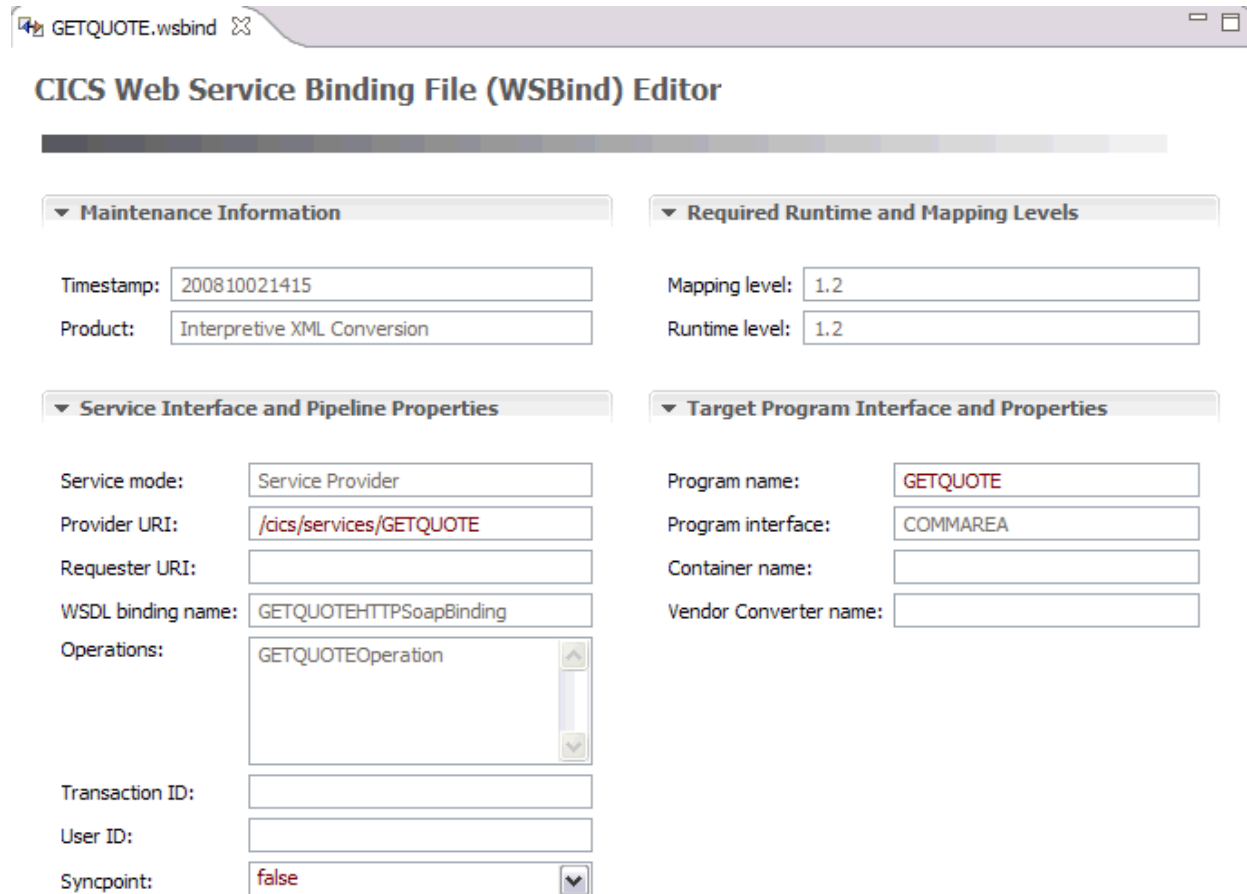


WSBind file editor (RD/z 7.5)

Useful if you want to change the deployment characteristics without regenerating the entire WSBind file:

- URI
- Transaction ID
- User ID
- PROGRAM Name
- SYNC-ON-RETURN

See also SupportPac CS04 for CICS TS 3.2 and CICS TS 4.1



CICS Web Service Binding File (WSBind) Editor

Maintenance Information

Timestamp: 200810021415
Product: Interpretive XML Conversion

Required Runtime and Mapping Levels

Mapping level: 1.2
Runtime level: 1.2

Service Interface and Pipeline Properties

Service mode: Service Provider
Provider URI: /cics/services/GETQUOTE
Requester URI:
WSDL binding name: GETQUOTEHTTPSoapBinding
Operations: GETQUOTEOperation
Transaction ID:
User ID:
Syncpoint: false

Target Program Interface and Properties

Program name: GETQUOTE
Program interface: COMMAREA
Container name:
Vendor Converter name:

The PIPELINE resource in CICS

- Specifies shared processing characteristics including:
 - ▶ Transport specific handler programs to invoke
 - HTTP or WMQ (IBM WebSphere MQ Series)
 - Or the CICS transport (TS 4.1 only)
 - ▶ Regular Handler programs to invoke
 - e.g. WS-Security, WS-AT, MTOM-XOP, etc.
 - ▶ A Terminal Handler (provider mode)
 - Usually a CICS supplied SOAP Handler, together with the CICS supplied application handler (DFHPITP)
- Data is passed through the pipeline using CICS Containers.
 - ▶ Handler programs may change the XML and the contents of any of the control containers



Local optimisation of EXEC CICS INVOKE

- If the requester and provider are both in CICS
 - ▶ TS 3.1 and TS 3.2
 - You can either EXEC CICS INVOKE a requester mode WEBSERVICE and send the SOAP request out to the network, which will then come in again to the provider mode WEBSERVICE
 - Or you can EXEC CICS INVOKE a provider mode WEBSERVICE and CICS will optimise the request down to the equivalent of EXEC CICS LINK
 - All or nothing: all of the pipeline or none of it
 - ▶ TS 4.1
 - You have new options to allow a requester mode pipeline to drive a provider mode pipeline without going out to the network
 - A compromise between flexibility and performance
- New applications can be deployed as Services but perform like Programs



CICS TS 3.2 vs CICS TS 3.1

- CICS TS 3.2 is much faster for most workloads
 - ▶ 64bit containers
 - ▶ Code page enhancements
 - ▶ More of CICS is Thread-safe
 - PIPELINE processing is done on an L8 TCB so thread-safety is relevant
- Support for more data mapping options
 - ▶ Easier to create applications top-down
 - ▶ Supports more WSDL documents
- Support for more specifications
 - ▶ MTOM/XOP
 - ▶ WSDL 2.0
 - ▶ WS-Trust (with **IBM Tivoli Federated Identity Manager – TFIM**)



CICS TS 4.1 vs CICS TS 3.2

- CICS TS 4.1 is much faster for most workloads
 - ▶ Mostly due to a rewrite of the SOAP node
 - ▶ A part of which is zAAP off-loadable
- Support for more data mapping options
 - ▶ Truncated (variable length) data
 - ▶ Bottom-up support for channel based applications
- A new XML processing API
 - ▶ EXEC CICS TRANSFORM XMLTODATA ...
 - ▶ EXEC CICS TRANSFORM DATATOXML ...
 - ▶ Useful for scenarios such as:
 - Writing PIPELINE handler programs that work with XML
 - Handling dynamic content in XML
- Support for more specifications
 - ▶ WS-Addressing



Summary

- CICS is a first class Web services end-point with a highly customisable technology stack.
- Web services enable interoperability with products and services from many different vendors.
- You can use industry standard and best-of-breed tools to interact with CICS & there's an ecosystem of associated products that add value.
- **Come back next week for Part 2** where we'll discuss deployment considerations including Security, WLM and Performance.



Additional Product Resources

- CICS Transaction Server Support Web page:
<http://www.ibm.com/software/http/cics/tserver/support/>
- CICS Featured documents:
<http://www.ibm.com/support/docview.wss?rs=1083&uid=swg27006900>
- Sign up to receive technical support emails:
<http://www.ibm.com/software/support/einfo.html>
- Follow IBM_CICS support news on Twitter:
<http://www.ibm.com/support/docview.wss?rs=1083&uid=swg21384915>
- Webcasts for CICS and OMEGAMON:
<http://www.ibm.com/support/docview.wss?rs=1083&uid=swg27007244>
- IBM Education Assistant modules:
<http://publib.boulder.ibm.com/infocenter/ieduasst/stgv1r0/index.jsp>



Join WebSphere Support Technical Exchange on Facebook!

The screenshot shows the Facebook page for 'WebSphere Support Technical Exchange'. The page header includes navigation tabs: Wall, Info, Photos, Discussions, Events, and a plus icon. The main content area features a 'What's on your mind?' post box with a 'Share' button. Below this, there are several event listings for 'WebSphere Support Technical Exchange' sessions, including 'Using IBM Tooling to improve your self-help abilities for WebSphere Commerce', 'Response Time Analysis for Databases and Web Services in WebSphere Application Server', 'Do-It-Yourself: WebSphere Commerce Problem Determination', 'WebSphere Application Server - Message Store Overview', and 'Introduction to the Java Consumability Tools and Java Guided Troubleshooting'. The left sidebar contains sections for 'IBM WebSphere tech experts share info', 'Information', 'Insights', 'Fans', and 'Photos'. The right sidebar features advertisements for 'Summerlicious at Drake' and 'CDI College'.

- Stay up-to-date on upcoming webcast sessions
- Suggest future topics
- Suggest program improvements
- Network with other product users
- And More...

Become a fan now!

<http://www.facebook.com/pages/WebSphere-Support-Technical-Exchange/121293581419>

Questions and Answers

