



The future runs on System z

CICS Transaction Server

Web Services



Acknowledgements

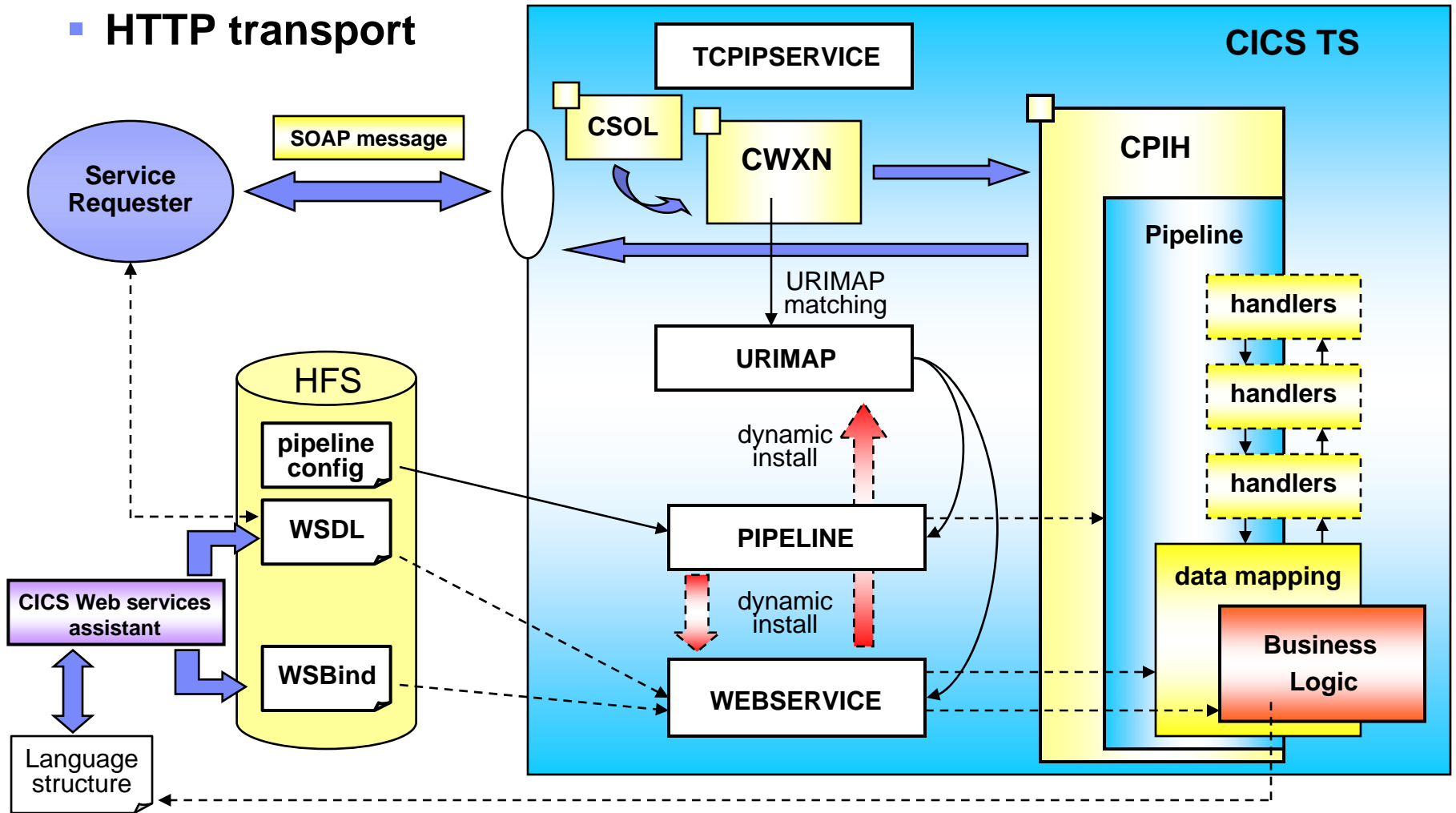
- The following are trademarks of International Business Machines Corporation in the United States, other countries, or both: IBM, CICS, CICS/ESA, CICS TS, CICS Transaction Server, DB2, MQSeries, OS/390, S/390, WebSphere, z/OS, zSeries, Parallel Sysplex.
- Java, and all Java-based trademarks and logos, are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
- Other company, product, and service names and logos may be trademarks or service marks of others.

Agenda

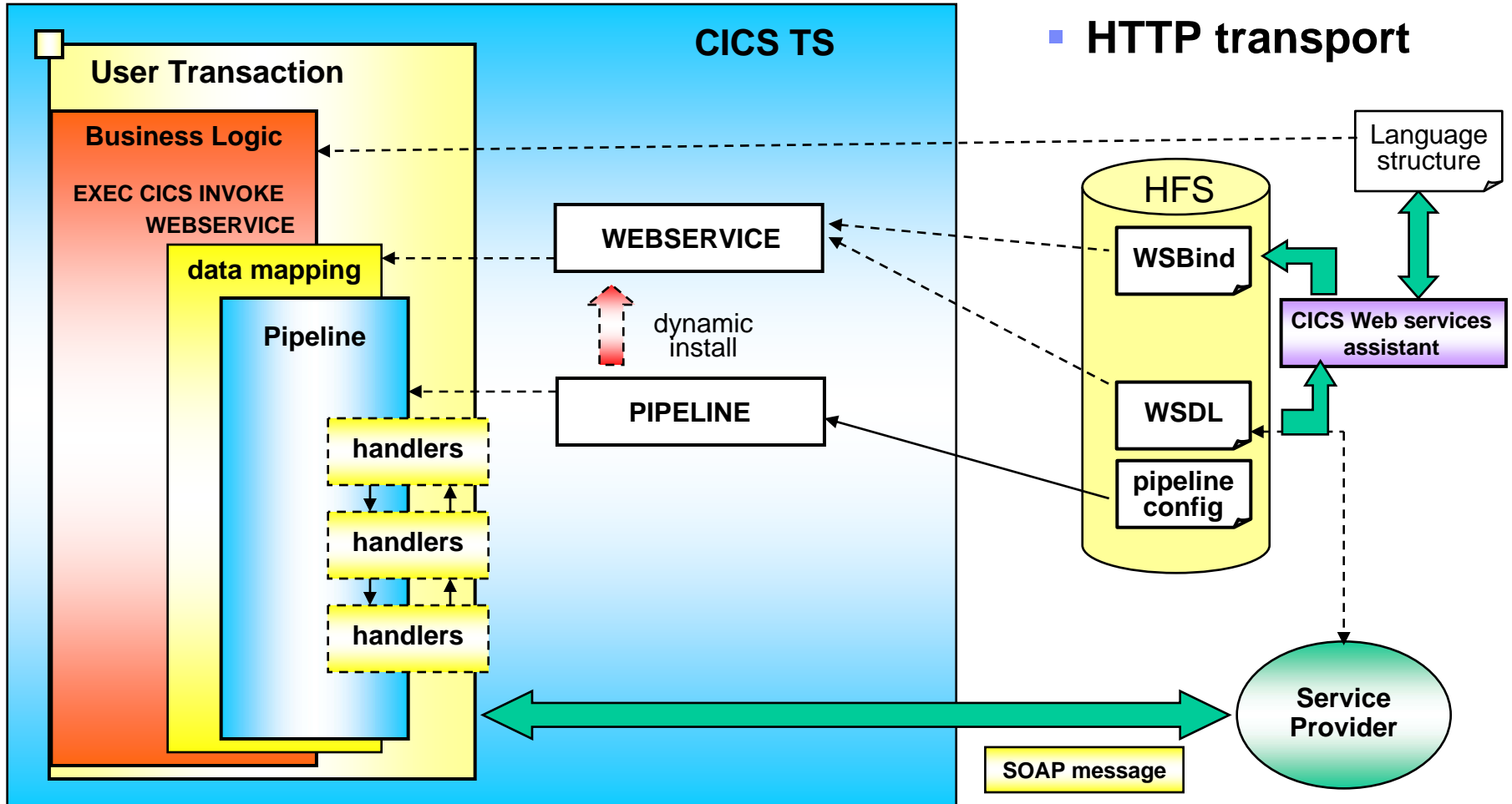
- **Introduction**
- **Web services**
 - Development
 - Deployment
 - Debugging
- **Summary**

CICS as a Service Provider

■ HTTP transport

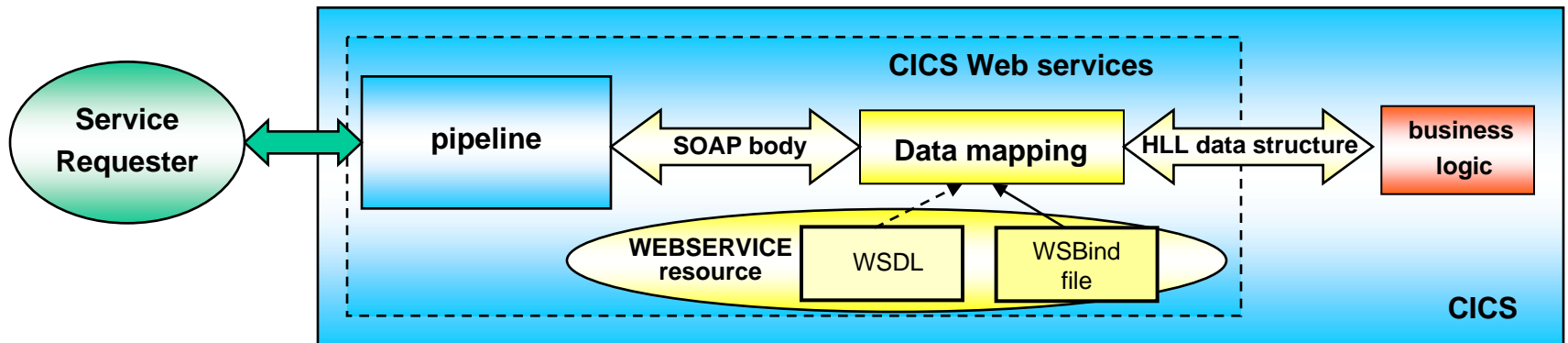


CICS as a Service Requester

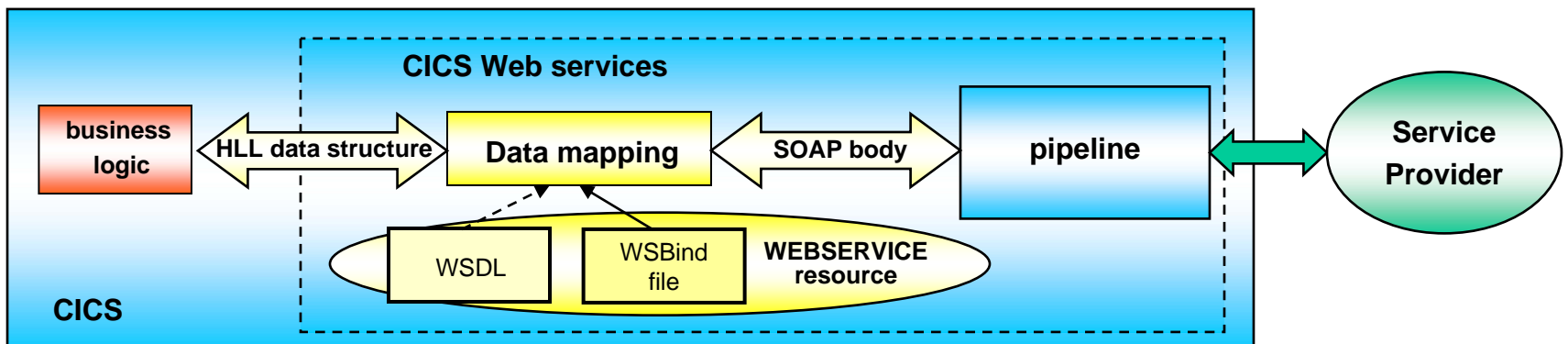


CICS usage of the WSBind file

- CICS as a service requester



- CICS as a service provider

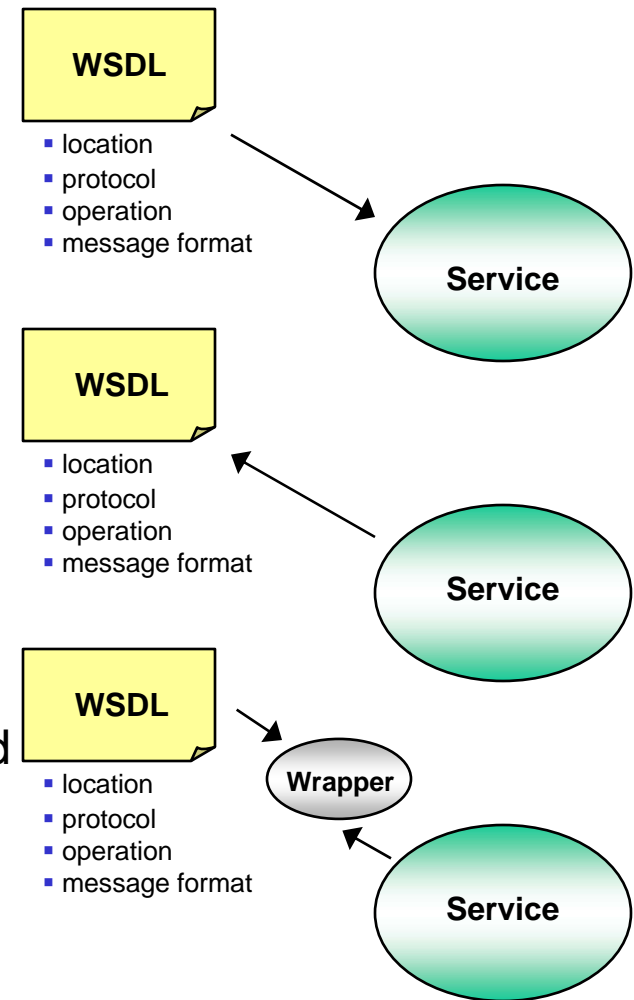


CICS Web Service Artifacts

- System Programmer
 - CICS Resource Definitions
 - TCPIPService, URIMAP, PIPELINE, WEBSERVICE
 - Handlers
 - Should not contain business logic
 - Likely to perform 'system' functions and used for several services
- Application Programmer
 - Business logic
 - Interface to the business logic when exposed as a CICS program
 - COMMAREA or Container layout
 - Interface to the business logic when exposed as a Web service
 - WSDL
 - Conversion between the business logic Web service interface and the business logic CICS program interface
 - WSBind file
 - Converter Programs (optional)

Develop Web Services Artifacts

- “Top down” approach
 - Create a service from an existing WSDL
 - Create a new Web service application
 - Better interfaces for the requester
 - New CICS code using the new languages structure
- “Bottom up” approach
 - Create a WSDL from an exiting application
 - Expose the application as a Web service
 - Quicker implementation of the service
 - Potentially more complex interface for the requester
- “Meet in the middle” approach
 - Create WSDL from an existing application, modify the WSDL and create a wrapper from the modified WSDL
 - Indirectly expose the application as a Web service
 - More suitable interface fro the requester
 - Minimal, if any, CICS development



Application Development: Bottom-up

- Start with an existing CICS application
 - Interface 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
- Use RD/z for modern interface;
Enhanced input selection;
Better data types support;
Field suppression

```
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         filedC 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 '2'
  -->
<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.com"
  <types>
    <xsd:schema attributeFormDefault="qualified"
      elementFormDefault="qualified" targetNamespace="http://www.NULLPROG.testSup.com"
      xmlns:tns="http://www.NULLPROG.testSup.com" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    <xsd:annotation base="http://www.ibm.com/software/htp/cics/annotat
      <xsd:documentation
        source="http://www.ibm.com/software/htp/cics/annotat
      </xsd:documentation>
    </xsd:annotation>
    <xsd:annotation
      <xsd:appinfo source="http://www.ibm.com/software/htp/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>
```

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 minOccurs="5" maxOccurs="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/ls2ws.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 Considerations

- Applications are:
 - Commarea based (but may use a Channel with a single Container) (*can use multiple containers in CICS TS 4.1*)
 - Provider mode (CICS is the server, not the client)
- 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 data types
 - 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)

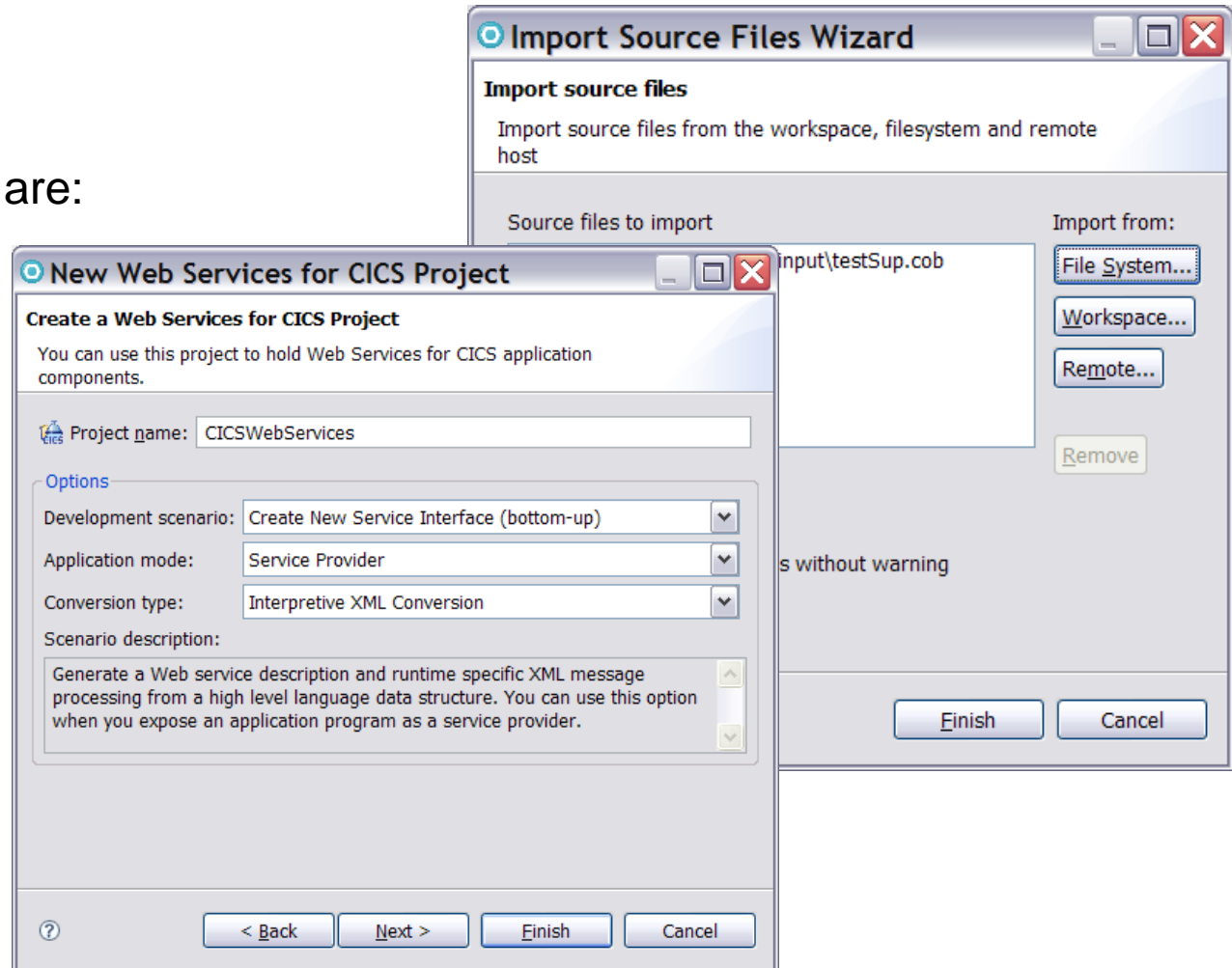
Using RD/z 7.5 to expose a Web service

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

The supported scenarios are:

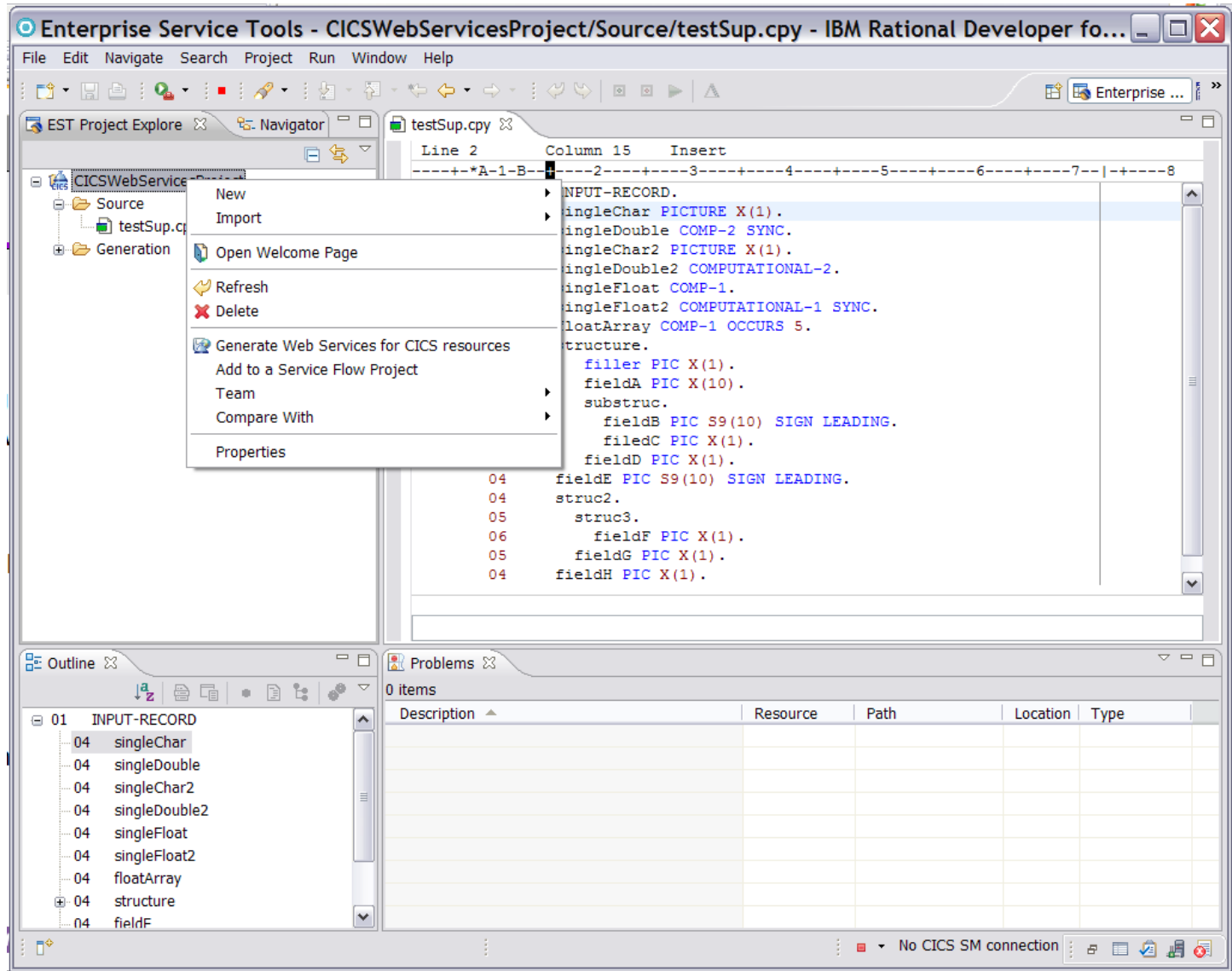
- bottom-up
- top-down
- meet-in-middle

Input files can be local
or remote



Using RD/z 7.5 to expose a Web service...

Generate the Web Services for CICS resources

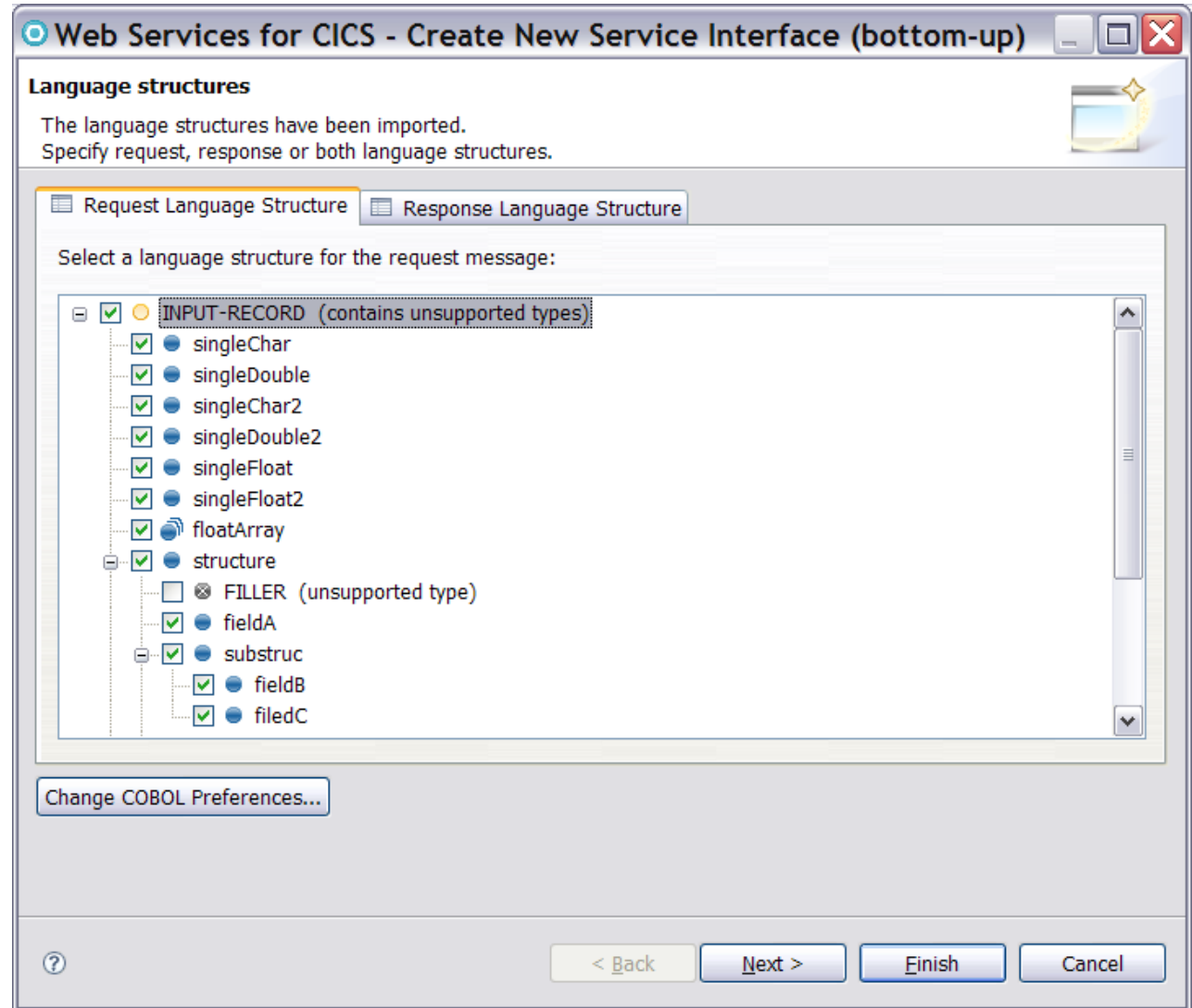


Using RD/z 7.5 to expose a Web service...

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...

Set any input parameters and configuration options that are required.

Web Services for CICS - Create New Service Interface (bottom-up)

Web Services for CICS
Specify generation options for the Web service binding file (WSBind)

Basic Options | Advanced Options

Specify targets for WSBind file

Generate to: Same project Remote location

WSBind file container: /CICSWebServicesProject/Generation/Targets

WSBind file name: testSup .wsbind

Log file name: testSup .log

Overwrite WSBind file

Specify application program properties

Program interface: COMMAREA

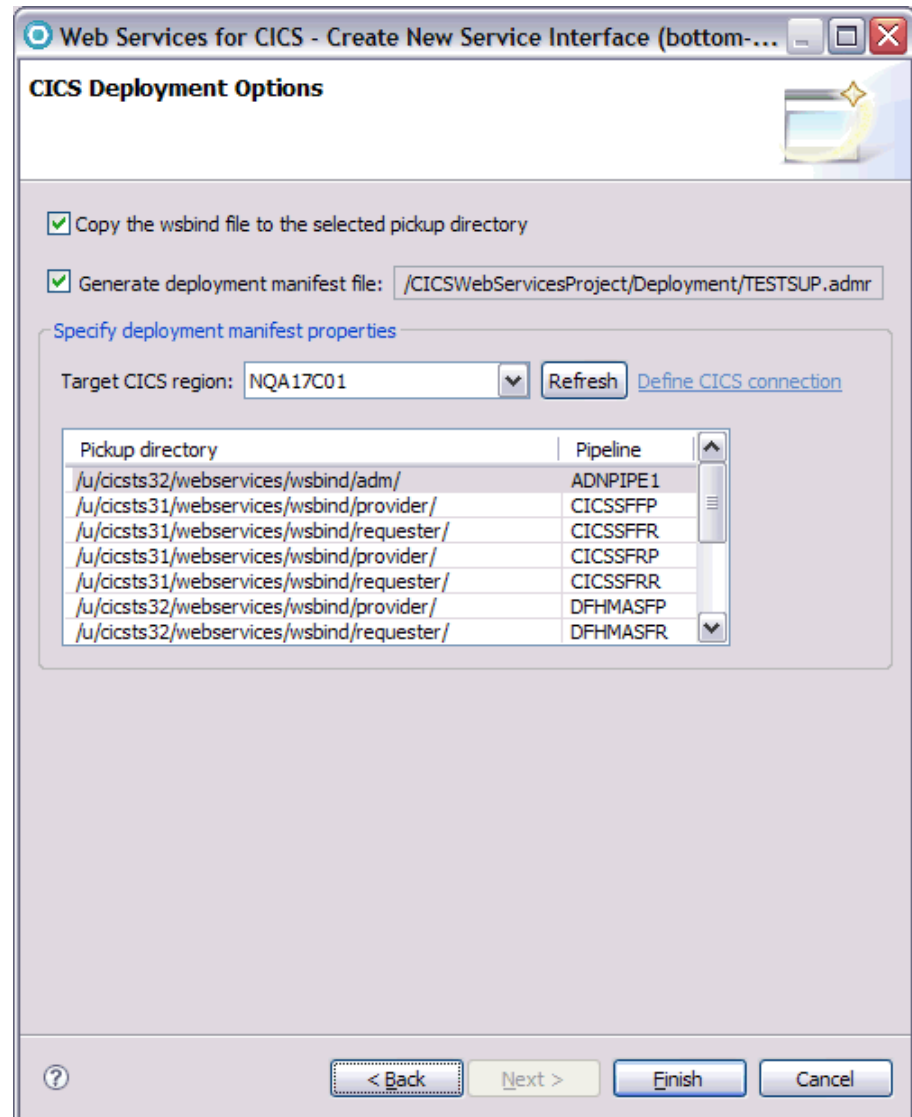
Container name: testSup

Using RD/z 7.5 to deploy a Web service...

Deploy the generated artifacts 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.



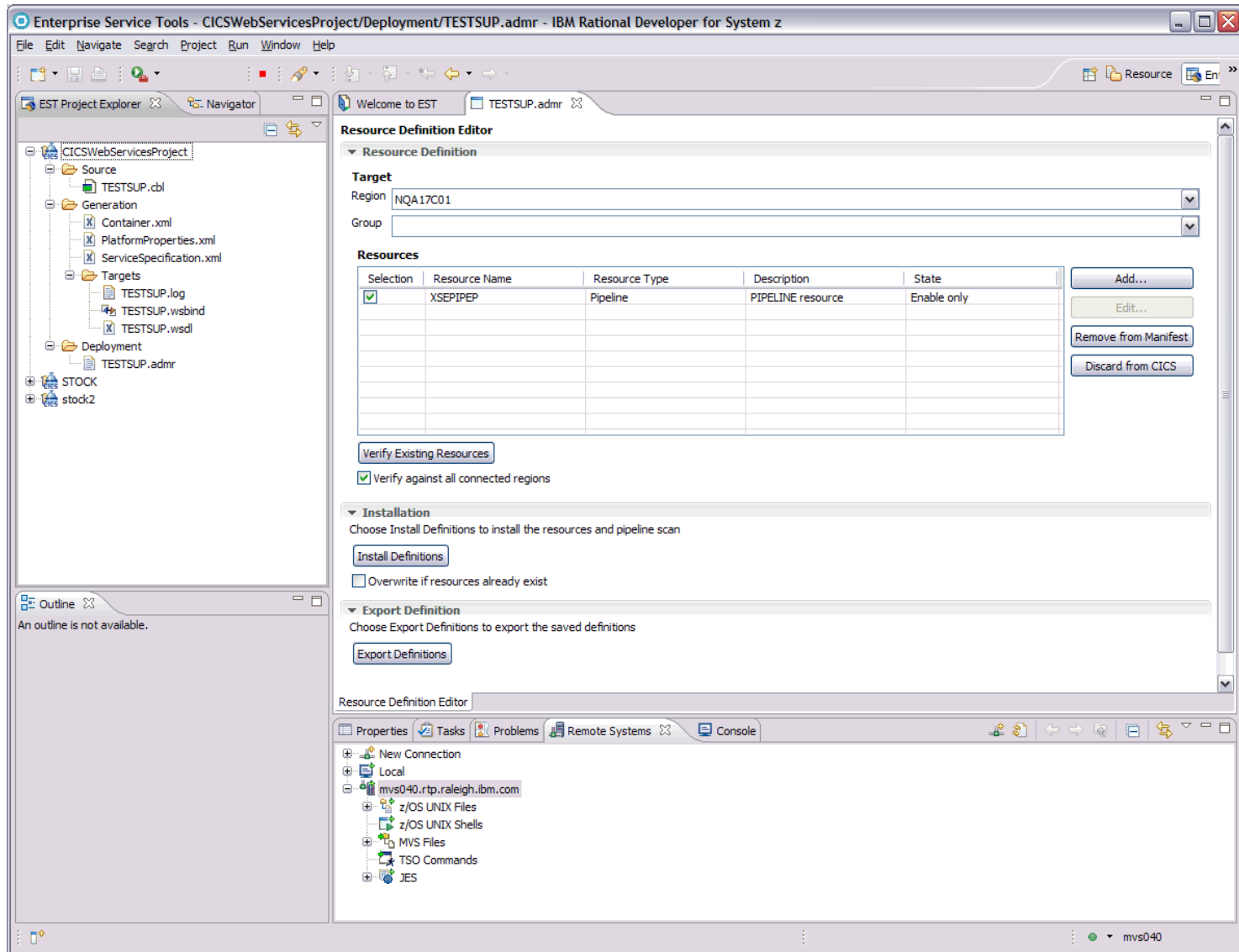
Artifacts generated in RD/z 7.5

The WSDL
(to describe the service)

The WSBind file
(needed by CICS)


The log file
(contains diagnostics)

An ADM Manifest file
(contains deployment information)



Testing the Web service

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

 **Invoke a WSDL Operation** [Source](#)

Enter the parameters of this WSDL operation and click **Go** to invoke.

Endpoints

▼ **Body**

▼ [NULLPROGOperation](#)

[singleChar](#) string

[singleDouble](#) double *

[singleChar2](#) string

[singleDouble2](#) double *

[singleFloat](#) float *

[singleFloat2](#) float *

▼ [floatArrav](#)

Testing the Web service...

- 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
- A web service in a single day

```

</soapenv:Header>
<soapenv:Body>
    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>
 

```

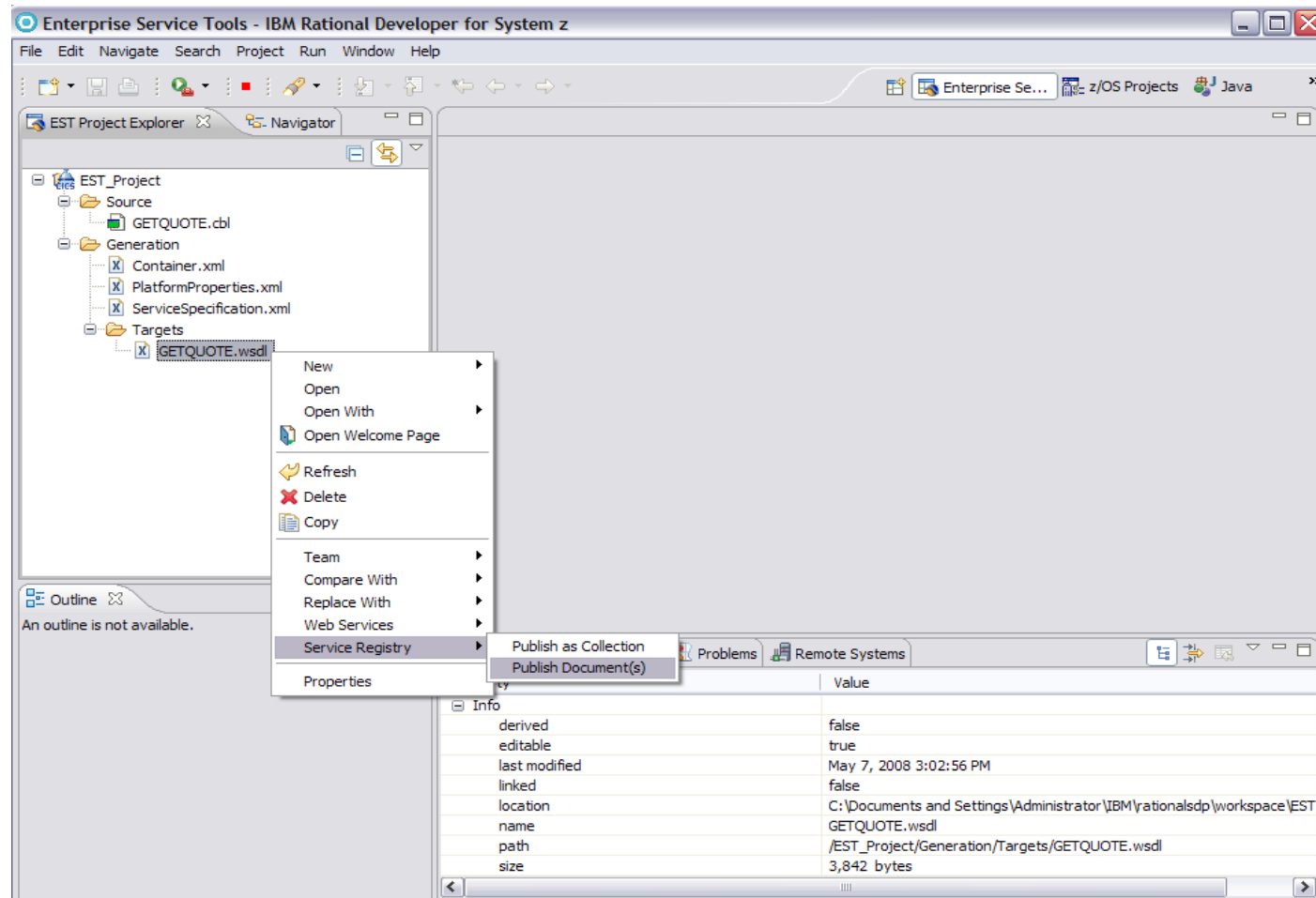
Debugging

- 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
 - 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 the CICS SupportPac CA1N for publishing from JCL (integrated with DFHLS2WS in TS V4.1)



Should generated WSDL be published?

- Probably not! (*but you can if you want to*)
 - Customize it first in order to:
 - » Have meaningful field names
 - » Your choice of namespaces, operation names, service names, etc.
 - » Remove unwanted content (such as annotations)
 - » Have the right endpoint and soap versions identified
 - Then reprocess the customised generated WSDL using the top-down tooling
 - » This may require a 'wrapper' program to be written
 - » See the CICS Information Center for more details

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 capability in RD/z to aggregate multiple CICS PROGRAMs into a single 'micro-flow' Web service
 - » Including Terminal based interactions (BMS)
 - » This minimizes the network interactions
 - » Further orchestration can be done with IBM WebSphere Process Server (WPS)

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/">

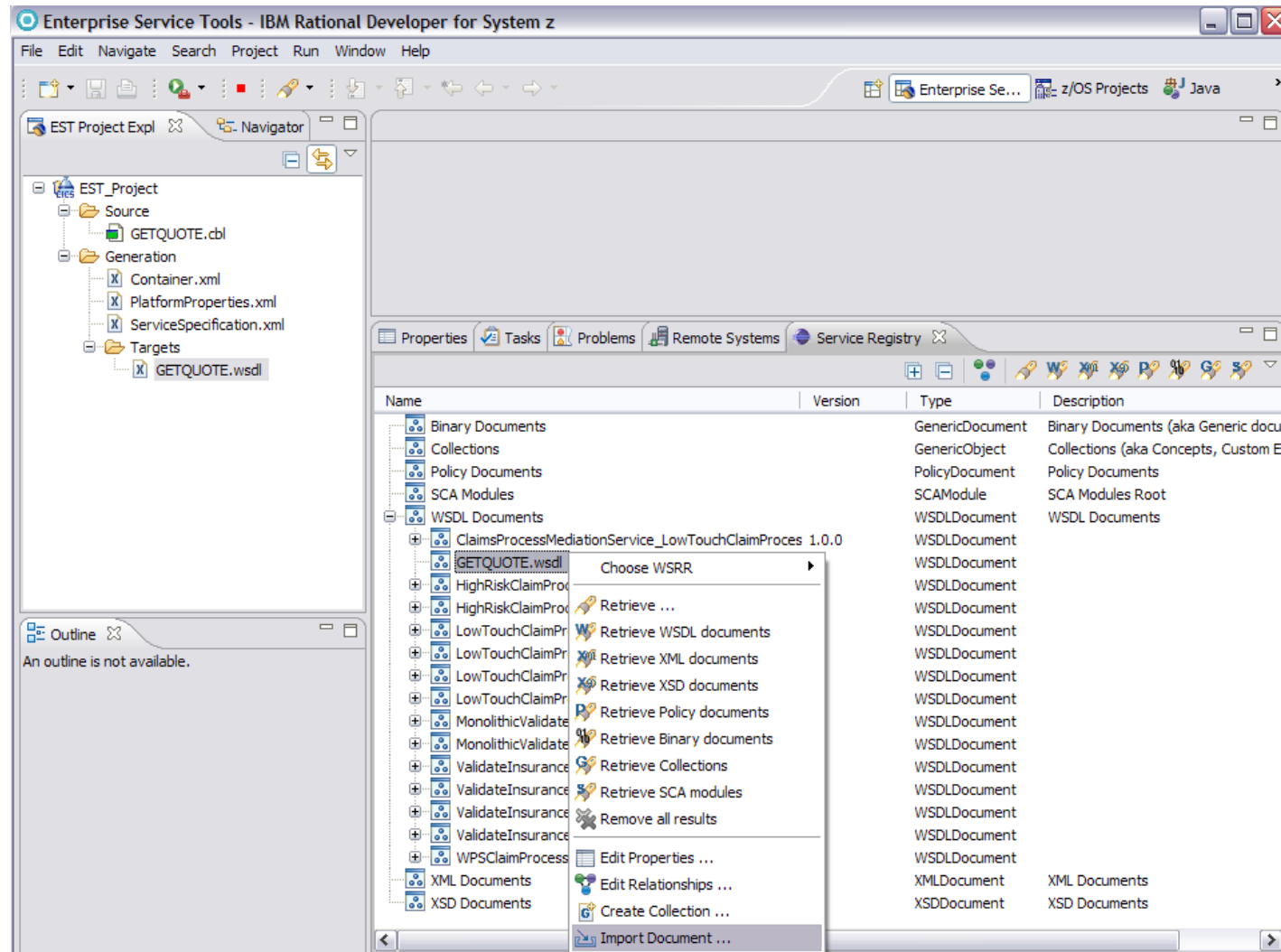
  <types>
    <s:schema targetNamespace="http://test.org/"
      xmlns:s0="http://test.org/"
      xmlns:s="http://www.w3.org/2001/XMLSchema">
      <s:complexType name="customerType">
        <s:sequence>
          <s:element name="accountNumber" type="s:int" />
          <s:element name="name" type="s:string"/>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

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

Importing WSDL from WSRR

You can import WSDL from WSRR into RD/z.

WSRR supports the complete life cycle of the WSDL and understands that there may be multiple test versions and production versions.



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 the service
 - Using the generated language structures
 - » Comments are generated into the language structures
 - Many options exist in DFHWS2LS to adjust the data mapping (e.g. default length for strings)
 - RD/z will generate some template code to get you started
 - » Interpreted mode only (not compiled mode)

Performance Considerations

- Complexity will cost you!
 - Sending large volumes of data is expensive
 - CPU costs increase with the number of XML tags used (regardless of the length of data sent)

See: <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
 - 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...

- **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 CICS TS V4.1 a client mode URIMAP may be named on the INVOKE SERVICE 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 specified 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** 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

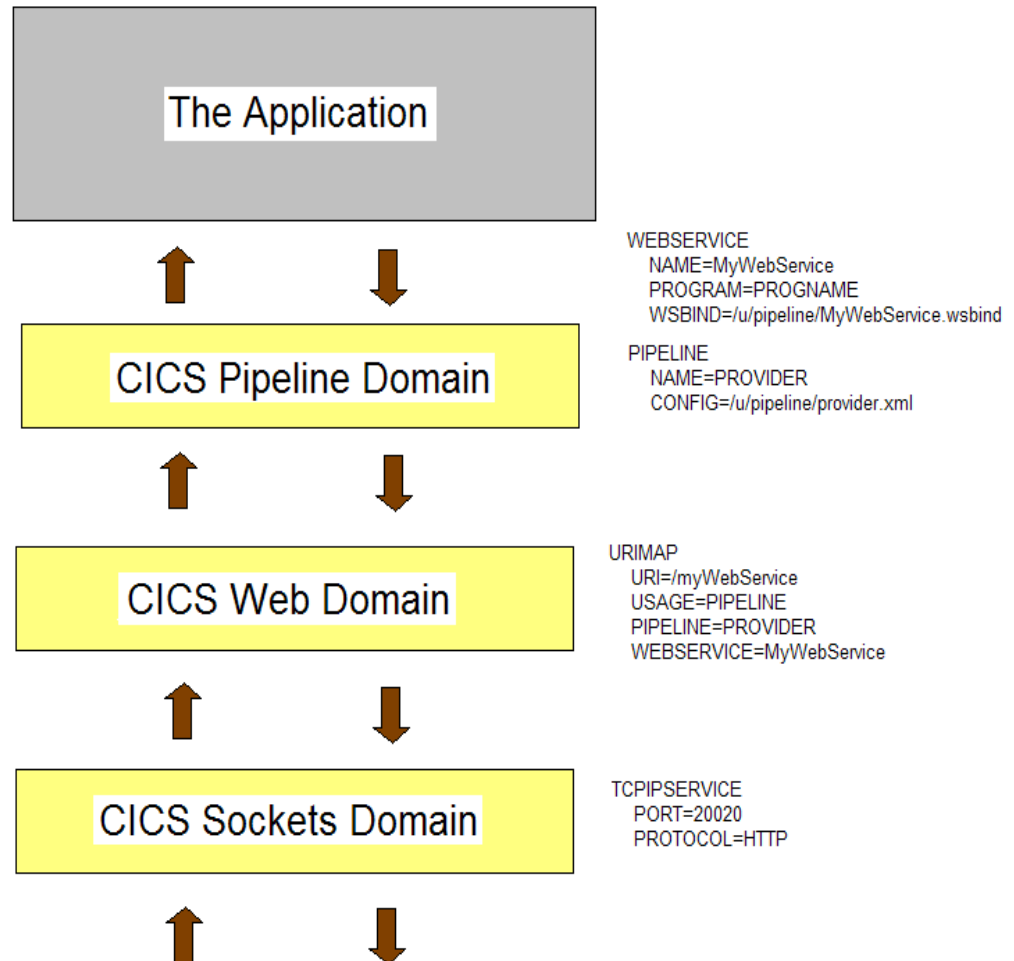
Identifies shared qualities of service

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 URIMAP resources via the CSD

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 V3.2 and CICS TS V4.1

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 a set of processing characteristics including:**
 - Transport specific handler programs to invoke
 - HTTP or WMQ (IBM WebSphere MQ Series)
 - 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
- **Data is passed through the pipeline using CICS Containers.**
 - Handler programs may change the XML and the contents of any of the control containers

The SOAP Handler

- **Either SOAP 1.1 or SOAP 1.2**
 - In provider mode the SOAP 1.2 Handler can support either version of SOAP
 - In requester mode the SOAP 1.2 Handler always generates SOAP version 1.2
- **Allows a set of SOAP Header Handler programs to be called**
- **In provider mode it links to an 'Application Handler'**
 - DFHPITP is the normal CICS supplied handler

Channel and Containers

- **Many Containers exist on the Channel**
 - Containers you may find useful include:
 - DFHWS-URI (the URI for the service)
 - DFHWS-SOAPACTION (the SOAP Action value)
 - DFHWS-WEBSERVICE (the resource name)
 - DFHWS-SOAPLEVEL (the SOAP version)
 - DFHREQUEST/DFHRESPONSE
 - DFHWS-RESPWAIT (the time-out in requester mode)
- **In CICS TS V3.2 the data is stored in 64bit storage**
 - But copied into 31bit storage when read

Customization and Extensibility options

- **The PIPELINE is designed to be customizable**
 - Configuration is in an XML configuration file
 - Which is more flexible than a CSD definition
 - You can change almost anything
 - e.g. It's also used for ATOM and PHP support (TS 3.2)
 - GLUE point in the PIPELINE (TS 3.2)
 - 12 GLUE points in TS 4.1
 - Vendor customization is possible
 - e.g. IBM Tivoli Composite Application Manager for SOA (ITCAM for SOA)

Security

- **HTTPS is supported**

- Good for point-to-point security
- Simple identity propagation

- **WS-Security**

- Useful if the data has to be secured irrespective of the point-to-point security (e.g. through proxy servers, or through a distributed ESB such as IBM WebSphere Enterprise Service Bus)
- Digital Signatures, encryption, X.509 certificates, etc.
- Sophisticated identity propagation
- Can be very expensive in terms of CPU
 - » Consider offloading processing to IBM WebSphere DataPower boxes

Distributed Transactions

- **WS-Atomic Transactions (WS-AT)**
 - Distributed 2-Phase commit over SOAP
 - » Distributed transactions are slow.
 - » Avoid using them unless you have a real requirement for them.
 - Also supported in IBM WebSphere Application Server (WAS)
 - » And by other vendors

CICS Topology

- **A common practise is to use Web Service Owning Regions**
 - WORs can be CICS TS V3.2 whilst AORs are CICS TS V3.1
 - » CPSM can be used for routing and balancing
 - Early access to the newest mapping levels for data mapping prior to migrating the AORs
 - » The Web services support in CICS has evolved at a rapid pace with valuable new function and performance improvements in CICS TS V3.2 and in V4.1
 - AORs can link to applications in the WOR for INVOKE WEBSERVICE requests

Management in CPSM

WEBSERVICE resources can be found under:

- CICS Operations Views:
- TCP/IP service operations views:
- Web Service

The screenshot shows the 'Web service' management interface. The left-hand navigation menu includes options like Alerts, Regions, Activity, Connectivity, Files & DB2, Journals, Queues, Transactions, Programs, Enterprise Java, History, Administration, and Favorites. The main content area displays a search filter for 'EYUVC1280' with 44 records collected at 10/08/08 14:15:03. Below the search bar is a table of web services.

Record	CICS system name	Name	Web service status	Number of times web service used	Pipeline in which this web service is installed	Dynamically installed URI map for this web service
1	YCWJOG3	catalogue	INSERVICE	0	BASIC11P	\$816270
2	YCWJOG3	catalogueChannel	INSERVICE	0	BASIC11P	\$107580
3	YCWJOG3	catalogueM21	INSERVICE	0	BASIC11P	\$241330
4	YCWJOG3	catalogueReq	INSERVICE	0	BASIC11R	
5	YCWJOG3	catalogueWS2LS	INSERVICE	0	REDTEST2	\$310300
6	YCWJOG3	jogcatal	INSERVICE	0	REDTEST2	\$124200
7	YCWJOG3	testlogue	INSERVICE	0	BASIC11P	
8	YCWJOG3	toxback	INSERVICE	0	BASIC11P	\$220000
9	YCWJOG3	toxcatalogue	INSERVICE	0	REDTEST2	\$335430
10	YCWJOG3	toxretrn	INSERVICE	0	BASIC11P	\$415230

Management with IBM Omegamon XE

IBM Omegamon XE for CICS also has a Web Services Analysis view

Web Services: Detail - TAMORAN - SYSADMIN

View: Physical

Web Services: Summary

System ID	CICS Region Name	Web Service Name	Status	Use Count	Pipeline Name	Program Name	URIMAP Name	Container Name	Program Interface
MV2C	CICSGBA1	dispatchOrder	Inservice	0	EXPIPE02				n/a
MV2C	CICSGBA1	dispatchOrderEndpoint	Inservice	0	EXPIPE01	DFH0X0DE	\$320460		Commarea
MV2C	CICSGBA1	inquireCatalog	Inservice	0	EXPIPE01	DFH0XCMN	\$320461		Commarea
MV2C	CICSGBA1	inquireCatalogClient	Inservice	0	EXPIPE02				n/a
MV2C	CICSGBA1	inquireCatalogWrapper	Inservice	0	EXPIPE01	DFH0X0CW	\$320462	DFHWS-DATA	Channel
MV2C	CICSGBA1	inquireSingle	Inservice	0	EXPIPE01	DFH0XCMN	\$320463		Commarea
MV2C	CICSGBA1	inquireSingleClient	Inservice	0	EXPIPE02				n/a
MV2C	CICSGBA1	inquireSingleWrapper	Inservice	0	EXPIPE01	DFH0X0SW	\$320464	DFHWS-DATA	Channel
MV2C	CICSGBA1	placeOrder	Inservice	0	EXPIPE01	DFH0XCMN	\$320465		Commarea
MV2C	CICSGBA1	placeOrderClient	Inservice	0	EXPIPE02				n/a
MV2C	CICSGBA1	placeOrderWrapper	Inservice	0	EXPIPE01	DFH0XPOW	\$320466	DFHWS-DATA	Channel

Web Service Details

CICS Region Name	Web Service Name	System ID	Status	Use Count	Pipeline Name	Program Name	URIMAP Name	Container Name	Last Modified	Program Interface	Validation Indicator	Binding	Endpoint
CICSGBA1	dispatchOrder	MV2C	Inservice	0	EXPIPE02				03/31/05 13 20 46	n/a	No	dispatchOrderSoapBinding	http://my-server:9080/exampleApp/dispatchOn

Hub Time: Mon, 01/23/2006 03:43 PM Server Available Web Services: Detail - TAMORAN - SYSADMIN

Local Optimization: INVOKE WEBSERVICE

- **If the the service requester and the service provider are co-located in CICS then the invocation can be optimized into a LINK.**
- **New architectural option for future COBOL development:**
 - Architect the interfaces between your new COBOL applications in UML using modern tooling (e.g. IBM Rational Software Architect - RSA)
 - Generate WSDL from the UML
 - Generate COBOL from the WSDL
 - Write loosely coupled COBOL applications that implement those interfaces and interact using EXEC CICS INVOKE SERVICE
 - » But that perform like EXEC CICS LINK
 - » With a well defined services based interface that can be re-hosted at will with little or no application changes
 - Similar to the 'service component architecture' support in CICS TS V4.1

CICS TS V3.2 vs CICS TS V3.1

- **CICS TS V3.2 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 V4.1 vs CICS TS V3.2

- **CICS TS V4.1 is much faster for most workloads**
 - Mostly due to a rewrite of the SOAP node
 - A part of which is 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

Documentation

- **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>

Security <http://www.redbooks.ibm.com/abstracts/sg247144.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>

- **Technical Support**

<http://www-01.ibm.com/software/htp/cics/support/>

Summary

- **CICS is a first class Web services end-point with a highly customizable 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**