CICS Web Services Support

GSE/CICS working group
Brussels, 26.10.2006

ibm.com/software/ts/cics

K-H Marquardt
IBM Deutschland SWG
MarquKH@de.ibm.com

© 2006 IBM Corporation
Acknowledgements

- The following are trademarks of International Business Machines Corporation in the United States, other countries, or both: IBM, CICS, CICS TS, CICS Transaction Server, DB2, MQ, 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.

- Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation 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

- Short introduction to SOA (Service Oriented Architecture)
- Overview of CICS WebServices Support
- Development approaches for CICS WebServices
- The sample application
- WS Security introduction
- WS Atomic Transaction (WS-AT) introduction
- Some hints
- Summary
CICS in an On-Demand IT solution

- CICS Web Services Support
  - SOAP 1.1 + 1.2
  - in/outbound

Flexible Processes
Web Services Architecture

- flexible business processes require flexible applications
  - Business orientation, not technically oriented
  - `adhoc` re-configurable components
  - de-coupled & platform neutral communication
- Requires application modernisation
  - cost savings by integration / reuse
- SOA is a foundation for flexible applications to support an **ON Demand Business**

prerequisite: **Standards !**

Basic Technology

- HTTP
- MQ
- XML
- SOAP
- WSDL

UDDI - Universal Description, Discovery and Integration
WSDL - web services description language
WSDL – Web Service Description Language

Standardized XML constructs to describe a Web Service

WSDL Structure

- **Type**: data type definitions
- **Message**: input/ output message definition
- **portType**: supported operations
- **Binding**: binds operations to network protocol(s)
- **Service**: service name and location endpoint

![WSDL Diagram](image)
New Components for CICS WebServices

Resource definitions

- **Transport definition:** TCPIPSERVICE (in CICS, HTTP/HTTPS)
  - QLOCAL (in WebSphere MQ)
- **URIMAP**
  - maps URI to corresponding service
  - points to pipeline and webservice resource definitions
- **PIPELINE**
  - points to pipeline configuration file (XML file)
  - defines the QoS for a webservice (security, atomic trx....)
- **WEBSERVICE**
  - points to bind file and WSDL file, defines service programm
  - bind file is used for data mapping between XML doc and language structure

Message handlers

- implement requested SOAP protocol and QoS
  - can include private handlers
  - defined in the pipeline configuration file

Web services assistant (utility):

- generates language copybooks from a given WSDL (DFHWS2LS)
- generates WSDL from a language copybook (DFHLS2WS)
- generates the web service binding file (WSBIND)
CICS as web service provider

1. Develop
   • Server Application
   • WSDL or Language structure(copybook)

2. Generate
   • Language structure or WSDL
   • WSBIND file

3. Configure
   • TCPIPSERVICE or WebSphere MQ
   • URIMAP
   • WEBSERVICE
   • PIPELINE
   • Pipeline configuration file
How the Resource Definitions Relate

- different pipeline for different QoS requirements
- CICS as requester needs a separate requester pipeline

- wsdll and wsbind file are tightly related => if wsdll changed wsbind must be re-generated
CICS as web service requester

1. Develop
   - WSDL
   - Language structure
   - Client Application

2. Generate
   - Language structure
   - WSDL
   - WSBIND

3. Configure
   - TCPIPSERVICE or WebSphere MQ
   - WEBSERVICE
   - PIPELINE
   - Pipeline configuration

Service Requester

Service Provider

Transport

Server Application

Handle response

Pipeline

HTTP or WebSphere MQ

HTTP

WebSphere MQ

SOAP

SOAP body

XML

Data Mapping

Handler chain

Language structure

HFS

CSD

Pipeline config

WSDL

WSBind

URIMAP

PIPELINE

WEBSERVICE

Client Application

exec cics invoke webservice(x)
**Web Service Development Approaches**

- **“Top down” approach**
  - create a (new) service program from an existing WSDL (create new CICS Web service application)
    + better interfaces for the requester
    - development cost

- **“Bottom up” approach**
  - create a WSDL from an existing application
  - expose the application as a Web service
    + quicker implementation of the service
    - more complex interface for the requester

- **“Meet in the middle” approach**
  - You have existing WSDL and existing application
  - create a WSDL from an existing application
  - modify the WSDL and create a wrapper program from the modified WSDL
  - indirectly exposes the application as a Web service
    + more suitable interface for the requester
    - development cost, but tools can help
Tools to assist development

- **Web Service Assistant (WSA) Batch Tool** *(DFHLS2WS + WS2LS)*
  - generates WSDL or language structure and WSBIND
    - supports Cobol, PLI, C, C++
  - good for proof of technology
  - redefines not supported
  - generated WSDL may need modifications (e.g. nillable data elements)

- **Websphere Developer for zSeries (WDz)**
  - XML Services for the Enterprise (XSE)
    - generates WSDL and WSBIND files
    - provides better granularity, userfriendly element names etc…
  - can generate skeleton pgm for “meet in the middle” approach
  - can generate a test client for immediate tests
  - best suited for real projects

- **Service Flow Modeler (SFM)**
  - provides flow modeling and mapping
    - flow can include all types of CICS backends (DPL-Pgm, 3270 Trx…)
    - backends “packaged” as one web service
  - generates all artifacts for deployment in SF runtime (CICS or HATS)
Service Flow Modeler and WS Deployment to SFR

CICS TS 3.1

Service Flow Runtime (BTS Process)

EIS Applications

WS Interface

WebSphere Developer for zSeries

generates artifacts for deployment in SFR

generates WSDL

CICS Service Flow
Flow

3270 Commarea

Deploy
The Sample Application

Described in the CICS Information Center
http://publib.boulder.ibm.com/infocenter/cicsts/v3r1/index.jsp
CICS functions → Web Services → The CICS catalog manager example application
The Base Application

- Catalog Manager
- Data handler
  (stub)
- Data handler
- CICS COBOL
  programs
- Dispatch
  manager
- Stock
  manager
- VSAM
  External
  Dispatch
  Manager
CICS Comarea Pgm exposed as Web service

Web service requester

inquireSingle.wsdl

generated by DFHLS2WS

CICS Web service support

inquireSingle.wsbind

HFS

CICS COBOL programs

VSAM
Generate Web Service Components with WSA

//INPUT.SYSUT1 DD *
LOGFILE=/u/exampleapp/wsbind/inquireSingle.log
PDSLIB=CICSHLQ.SDFHSAMP
REQMEM=DFH0XCP4
RESPMEM=DFH0XCP4
LANG=COBOL
PGMNAME=DFH0XCMN
PGMINT=COMMAREA
URI=exampleApp/inquireSingle
WSBIND=/u/exampleapp/wsbind/inquireSingle.wsbind
WSDL=/u/exampleapp/wsd1/inquireSingle.wsd1
*/

DFHLS2WS
Batch Utility

Input:
- Log file (HFS)
- Library containing copybooks
- (CA) copybook for request from client
- (CA) copybook for response to client
- language
- CICS Server Pgm name
- Interface type (commarea or container)
- universal request identifier
- location and name of WSBIND file (HFS)
- location and name of WSDL file (HFS)
Generated WSBIND and WSDL files

inquireSingle.wsbind

/u/exampleapp/wsbind/inquireSingle.wsbind
WSDL=/u/exampleapp/wsdl/inquireSingle.wsdl DFH0XCMN

......
<Wrapper>
  <CA-REQUEST-ID>&CA>&CA--REQUESTREQUEST --IDID;</CA-REQUEST-ID>
  <CA-RETURN-CODE>&CA>&CA--RETURNRETURN --CODECODE;</CA-RETURN-CODE>
  <CA-RESPONSE-MESSAGE>&CA>&CA--RESPONSERESPONSE --MESSAGEMESSAGE ;</CA-RESPONSE-MESSAGE>
  &CA-INQUIRE-SINGLE;
</Wrapper>
<DFH0XCMN xmlns="http://www.DFH0XCMN.DFH0XCP4.Request.com">
  &Wrapper;
</DFH0XCMN>
<SOAP-ENV:Body >&DFH0XCMN;</SOAP-ENV:Body >

......
inquireSingle.wsdl

<?xml version="1.0" ?>

......
<xsd:schema attributeFormDefault="qualified"
  elementFormDefault="qualified"
  targetNamespace="http://www.DFH0XCMN.DFH0XCP4.Request.com"
  xmlns:tns="http://www.DFH0XCMN.DFH0XCP4.Request.com"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:complexType name="ProgramInterface">
    <xsd:sequence>
      <xsd:element name="CA-REQUEST-ID" nillable="false">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:length value="6"/>
            <xsd:whiteSpace value="preserve"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      ......more......
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
Testing the WebService enabled Application
Testing the WebService enabled Application

TCPIP-SERVICE

URIMAP

PIPELINE

WEB-SERVICE

CICS COBOL programs

VSAM

Servlet

WSDL

JSP

browser

http://9.100.195.47:03102/exampleApp/inquireSingle

App Server

Workstation
TCPIPSERVICE Definition

- for HTTP transport
- listens on port (03102) for incoming Web Service requests
- default web alias transaction is CWXN

```
I TCPIPSERVICE (SOAPTCP)
RESULT - OVERTYPE TO MODIFY
  Tcpipservice(SOAPTCP)
    Openstatus( Open )
    Port(03102)
    Protocol(Http)
    Sslytype(NoSSL)
    Transid(CWXN)
    Authenticate(NoAuthentic)
    Connections(00000)
    Backlog( 00005 )
    Maxdatalen( 000032 )
    Urm( NONE )
    Privacy(NotSupported)
    Ciphers()
    Ipaddress(9.100.195.47)
    Socketclose(Wait)
    CloseTimeout(00000)
    Dnsgroup()
    Dnsstatus(
```

© 2003 IBM Corporation
URIMAP maps Request to the WebService in CICS

URI\MAP maps Request to the WebService in CICS

CWXN receives SOAP doc from listener trx. and finds URIMAP definition

TCPIP-SERVICE

URIMAP

PIPELINE

WEB-SERVICE

CICS COBOL programs

VSAM

Servlet

JSP

App Server

http://9.100.195.47:03102/exampleApp/inquireSingle

browser

Workstation
URIMAP

- maps URI (/exampleAPP/inquireSingle) to desired processing Pipeline (SOAPIPE1) and Webservice (inquireSingle) resource definitions
- default pipeline trx is CPIH – now invoked to process pipeline
- URIMAP can be generated automatically and installed via pipeline scan comnd

I URIMAP __
RESULT - OVERTYPE TO MODIFY
Urormap($923470)
Usage(Pipe)
Enablestatus( Enabled )
Analyzerstat(Noanalyzer)
Scheme(Http)
Redirecttype( None )
Tcpipservice() 
Host(*)
Path(/exampleApp/inquireSingle)
Transaction(CPIH)
Converter()
Program()
Pipeline(SOAPIPE1)
Webservice(inquireSingle)
Userid()
Certificate()
Ciphers()
+ Templatename()
CPIH processes pipeline

```
<?xml version="1.0" encoding="UTF-8" ?>

xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  ...
</SOAP-ENV:Envelop>
```
The Pipeline definition

```
I PIPELINE
RESULT - OVERTYPE TO MODIFY
  Pipeline(SOAPIPE1)
  Enablestatus( Enabled )
  Configfile(/u/eric/exampleapp/pipelines/configurations/basicsoap11provid)
  Configfile(er.xml)
  Shelf(/u/eric/exampleapp/pipelines/shelf/)
  Wsdir(/u/eric/exampleapp/pipelines/wsdir/)
```

- **pipeline configuration file** specifies the pipeline attributes
  - defines Qualities of Service (WS-Security etc.)
  - specifies msg handler chain to process Web Service Request

- **Wsdir**: pickup directory for wsbind files
- **Shelf**: runtime copies from wsdir
The message handler chain

CPIH now processes the handler chain according to the contents of the pipeline config file. Finally the XML body is parsed and mapped to the Comarea using the wsbind file.

```
<SOAP-ENV:Body>
  <q0:DFH0XCMN>
    <q0:Wrapper>
      <q0:CA-REQUEST-ID>
        01INQS
      </q0:CA-REQUEST-ID>
      <q0:CA-INQUIRE-SINGLE>
        <q0:CA-ITEM-REF-REQ>
          0010
        </q0:CA-ITEM-REF-REQ>
      </q0:CA-INQUIRE-SINGLE>
    </q0:Wrapper>
  </q0:DFH0XCMN>
</SOAP-ENV:Body>
```
The WEBSERVICE definition

```plaintext
I WEBSERVICES _
RESULT - OVERTYPE TO MODIFY
  Webservice(inquireSingle)
  Pipeline(SOAPIPE1)
  Validationst( Novalidation )
  State(Inservice)
  Urimap($923470)
  Program(DFH0XCMN)
  Pgminterface(Commarea)
  Container()
  Datestamp(20041207)
  Timestamp(09:23:47)
  Wsdlfile()
  Wsbind(/u/eric/exampleapp/pipelines/wsdir/inquireSingle.wsbind)
  Endpoint()
  Binding(DFH0XCMNHTTPSoapBinding)
```

- defines the target CICS program (DFH0XCMN)
- defines the interface to the target program (Comarea or Container)
- defines location and name of wsbind file ( ....wsdir/inquireSingle.wsbind)
- can be generated automatically and installed via pipeline scan cmnd
Security considerations with SOAP messaging

- https is not always sufficient for secure webservices processing
  - encryption on transport level, “all or nothing”, point-to-point
- demand for security on the message level
- how to include security credentials in the message?
- how to implement element-wise encryption, i.e. expose some parts for routing, hide critical data from unauthorized parties
- how to use digital signatures
- security must persist from originator to processing end-point, for the life of the transaction
- security must survive calls to external business partner
- use with, or instead of, protocol-level security

defined and standardized in the WS-Security Standard
CICS Support for WS-Security

- CICS WS-Security Message handler, DFHWSSE1
  - shipped via APAR 22736
- Signature validation of inbound message signatures
  - RSA-SHA1 & DSA-SHA1
- Signature generation for the SOAP body on outbound messages
  - RSA-SHA1
- Decryption of encrypted data in inbound messages
  - AES 128,192, 256 or tripledes, with key wrap RSA 1_5 and AES 128,192, 256 or tripledes.
- Encryption of the SOAP body content with the above algorithms
- various mechanisms to derive a User ID from an inbound message
- see redbook SG24-7206
Configuring CICS to Support WS-Security

In the pipeline configuration file add a security handler to the service handler list

```xml
<service>
  <service_handler_list>
    <cics_wsse_handler>
      ............
    </cics_wsse_handler>
  </service_handler_list>
  <terminal_handler>
    <cics_soap_1.1_handler/>
  </terminal_handler>
</service>
```
Transactional Support for Web Services

- **Web Services Standard focusses on 3 topics**
  - **WS-Coordination – WS-C**
    - how to establish a coordinator that creates and submits a trx.context
    - WS-C defines a *framework* for deploying coordination protocol sets
      - Activation Service – begin / end of transaction
      - Registration Service – register “participation” in a transaction
      - Coordination Context – create and maintain a trx. context
  - **WS-AtomicTransaction – WS-AT**
    - the well known trx. model based on Atomicity, Consistency, Isolation, Durability / 2PC
    - short lived trx. where results are not made visible until commit or rollback
  - **WS-BusinessActivity – WS-BA**
    - long lived business trx. where results of operations are made visible before completion of entire unit of work
    - needs compensation logic rather than roll back to undo undesired work

- **more info available at**
  - IBM: Developer Works – Web Services: Technical Library/Standards
    http://www.ibm.com/developerworks/webservices
  - MSSoft: Web Services Development Center
    http://msdn.microsoft.com/webservices
  - BEA: Dev2Dev – WebServices
    http://dev2dev.bea.com/technologies/webservices/index.jsp
CICS Support for WS-AT – how it works

**Coordination Context:**

1. **WS request with Coordination Context**
2. **(req.to) Register**
3. **RegisteredResponse**
4. **WS Response**
5. **Prepare**
6. **Prepared**
7. **Commit**
8. **Committed**

**RS requestor pipeline (DFHWSATR):**

- **Coordinator:**
  - Create context and add to SOAP msg
  - Process WS-C header, invoke RS requester pipeline
  - Continue “normal” provider pipeline

- **Participant:**
  - Invoke RS requester pipeline to start commit process

**RS provider pipeline (DFHWSATP):**

- **Coordinator:**
  - Invoke RS provider pipeline
  - Continue “normal” provider pipeline

- **Participant:**
  - Continue “normal” provider pipeline

**Web Service requester app E.C. invoke..**

**Web Service provider app**

**Registration and Protocol Services (DFHPIRS):**

- **RS - URI:** requester.example.com/cicswsat/RegistrationService
- **RS-URI:** provider.example.com/cicswsat/RegistrationService
CICS Support for WS-AT - Resources

Resources in new group DFHWSAT

? Pipeline **DFHWSATP - registration service provider pipeline**
  ? provider pipeline that enables CICS to act as a WS-C coordinator and as a
  WS-AT 2PC protocol handler

? Pipeline **DFHWSATR - registration service requester pipeline**
  ? requester pipeline that allows CICS to communicate with external
  coordinators and WS-AT 2PC protocol handlers

? URIMAP **DFHRSURI** (registration service URI)
  ? associates inbound WS-C requests with provider pipeline **DFHWSATP**

? Program **DFHWSATH** (atomic trx. handler)
  ? defined in pipeline config. file of WS-requester and WS-Provider
  ? must be included to enable WS-Atomic trx.
  ? if CICS is the WS-Coordinator (in the role of WS requester)
    : creates coordination context and adds it to SOAP request message
  ? if CICS is the WS-Provider
    : receives coordination context and invokes its local RS-requester pipeline
    to request registration with the coordinator

? Program **DFHPIRS** (pipeline registration service)

? Program **DFHWSATX** (2PC handler)

? Program **DFHWSATR** (registration service requester pgm)
Updates to the pipeline config – WS Requester

...<cics_soap_1.1_handler>
 <headerprogram>
  <program_name>DFHWSATH</program_name>
  <namespace>
   http://schemas.xmlsoap.org/ws/2004/10/wscoor
  </namespace>
  <localname>CoordinationContext</localname>
  <mandatory>true</mandatory>
 </headerprogram>
</cics_soap_1.1_handler>
...

Address of the Registration service endpoint for the RS provider of this Region. Participants send Register requests to this address.

causes CoordinationContext to be created and added to the SOAP message before it is sent
Updates to the pipeline config. – WS Provider

```xml
<cics_soap_1.1_handler>
  <headerprogram>
    <program_name>DFHWSATH</program_name>
    <namespace>
      http://schemas.xmlsoap.org/ws/2004/10/wscoor
    </namespace>
    <localname>CoordinationContext</localname>
    <mandatory>false</mandatory>
  </headerprogram>
</cics_soap_1.1_handler>

<service_parameter_list>
  <registration_service_endpoint>
    address:port/cicswsat/RegistrationService
  </registration_service_endpoint>
</service_parameter_list>
```

extracts data from the CoordinationContext header and invokes registration request

address of the registration service endpoint for RS provider of this region. Coordinator sends prepare and commit (or Abort) requests to this address
Some hints …

- **WSA utilities have limitations**
  - see WSA documentation in web services guide
  - wrapper pgm (“meet in the middle”) most likely needed in real world projects
  - for FSUM error messages from WSA see z/OS 1.x UNIX System Services Messages & Codes
    → number is a reference to the line in script
  - provide correct Java paths settings

- **Establish appropriate USS Authorizations**

- **Invest time in webServices design**
  - compound services vs. very granular services
  - complex vs. simple interface for web service requesters
  - consider network traffic and XML parsing overhead
    • avoid deep XML nesting levels

- **Literature**
  - CICS TS 3.1 Web Services Guide SC34-6458
  - Implementing Web Services in CICS SG24-7206 (redbook)
  - Application Development for CICS Web Services SG24-7126 (redbook)

- **CICS Web Services Class in Montpellier**
Summary

Web Services Support opens new opportunities to integrate CICS applications in an SOA and to interoperate with application components on different platforms
- “loose coupling”, platform and language neutral
- flexible composition of services to support flexible business processes

CICS TS supports the key web services standards
- standard conformance is key to gain flexibility and it influences the speed for building an On Demand IT environment

HTTP or MQ transport
- transparent for the application
- MQ benefits may be for example “assured delivery” and usage of WBI

WebServices Support is an integrated CICS component
- simple configuration and system management
- monitoring, statistics and trace support
Thanks for Your attention