XML Strategies for the CICS Transaction Server Environment

Dennis Weiand
IBM

Tuesday, August 3, 2010
Session 6903

Abstract

- As the use of XML continues to become more prevalent, knowing the options available in the CICS Transaction Server environment become increasingly important. CICS's Web service support, CICS's TRANSFORM API command, COBOL's PARSE verb, and Java parsers are some of the available options, but each of them are surrounded with many details. This session is intended to discuss the various XML usage options in the CICS Transaction Server environment, with a focus on using mapping level capabilities for dealing with common XML-related issues such as string handling and occurring item. Use of zAAP processors for offloading some of the XML parsing will also be discussed.

Trademarks

- The following terms are trademarks of the International Business Machines Corporation or/and Lotus Development Corporation in the United States, other countries, or both:
  - Redbooks(logo)™, AIX®, alphaWorks®, CICS®, DB2®, IBM®, IMS™, Informix®, MQSeries®, VisualAge®, WebSphere®
- The following terms are trademarks of other companies:
  - Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation.
  - Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc.
  - CORBA, CORBAServices, and IIOP are trademarks of the Object Management Group, Inc.
  - UNIX is a registered trademark of The Open Group in the United States and other countries.
  - Other company, product, and service names may be trademarks or service marks of others.

Notices

- This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this presentation in other countries.
- INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PRESENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF NON-INFRINGEMENT, MERCHANTABILTY OR FITNESS FOR A PARTICULAR PURPOSE.
- This information could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this presentation at any time without notice.
- Any references in this presentation to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Agenda

- **Parsing Options**
- **Places where we parse and generate XML in CICS**
  - Web services support
  - EXEC CICS TRANSFORM
  - RESTful Requests
  - ATOM
  - CICS Dynamic scripting
- **Common issues when using CICS's built-in parsing capability**
  - xsd:String
  - Occurring Items
  - Adding headers on Web service requests
  - xsd:any and xsd:anyType
  - Multiple Containers

XML Parsing Options on z/OS

- **XMLSS (z/OS XML System Services)**
  - General-purpose, validating parser, can handle huge messages
  - Parsing is zAAP eligible
- **XML Toolkit (XML4C** – can work with XMLSS)
- **IBM's Java** 6 comes with XML4J and XL TXE-J (Java is zAAP eligible)
  - XML4J
    - Based on Apache Xerces-J 2.9.0
    - Validating Parser
    - Supports JAXP
  - XL TXE-J
    - Non-validating Parser
- **COBOL PARSE** verb
  - With XMLSS compile option is zAAP eligible (Enterprise COBOL V4.1+)
  - Interactions with XMLSS improved in Enterprise COBOL V4.2
- **PLI**SAXA/PLISAXB/PLISAXC subroutines
- **Special-purpose**
  - CICS's 'shredder' – non-validating, fast
- **Custom/RYO**
  - Special-built code for your XML vocabulary could be fast

XML In CICS Applications

- **Process XML before getting to CICS**
- **CICS Web services**
- **SCA** (when exposing component or composite as a Web service)
- **EXEC CICS TRANSFORM**
- **RESTful Interfaces**
- **ATOM**
- **Event Processing**
  - WBE Format
  - CBE Format
  - Custom Adapter
- **Dynamic Scripting**
  - ATOM Feeds
  - From Java, PHP, Groovy

Each are discussed on upcoming pages

XML Conversion outside of CICS

- **Endpoint of Web service in WAS or ESB**
  - Receive Web service request outside CICS, then communicate to CICS using CICS TG, WOLA, WMQ, etc
- **WebSphere Message Broker or WESB**
  - Request received by ESB – ESB communicates to CICS
- **WebSphere Transformation eXtender**
  - Used to extend capabilities of ESB
  - Industry Transformation Packs
    - SEPA, SWIFT, HIPA EDI, ACORD, more
- **DataPower**
  - Appliance – can be used for XML transformation
- **DB2 pureXML**
  - DB2 understands XML

WAS = WebSphere Application Server

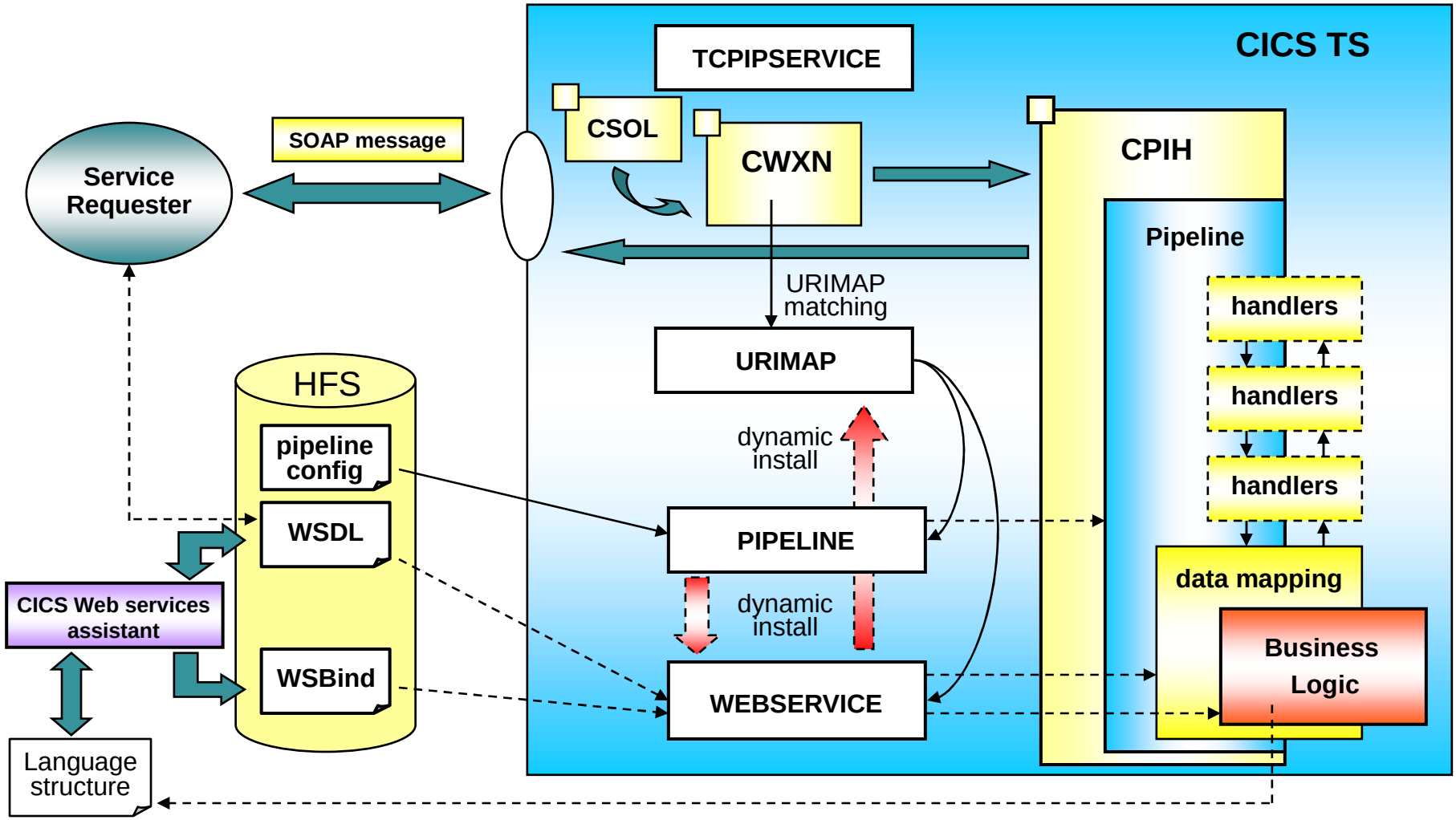
WESB = WebSphere Enterprise Services Bus

WebSphere TX=WebSphere Transformation eXtender

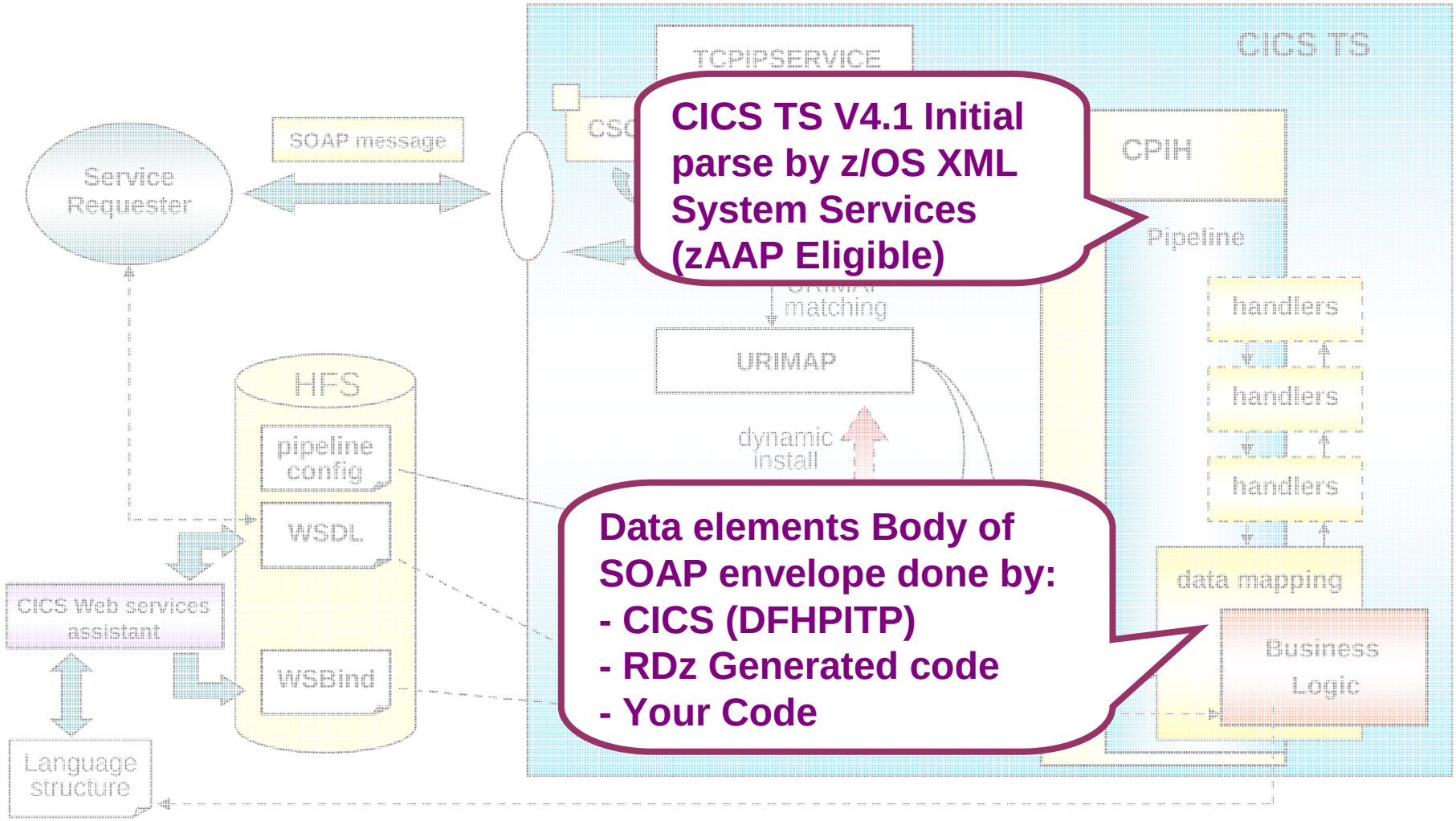
Leave the XML processing to DB2

- **DB2 – pureXML – DB2 9.1+**
  - Store XML with no parsing, mapping or conversion
  - Can be modified using XPath (DB2 V10.1)
  - See developerWorks article for more info on a CICS/DB2 pureXML web service
  - See DB2 InfoCenter for more information on pureXML
- **Using CICS with DB2 pureXML**
  - http://www.ibm.com/developerworks/data/library/techarticle/dm-1004cicsdb2purexml/index.html

CICS' Web Service Support

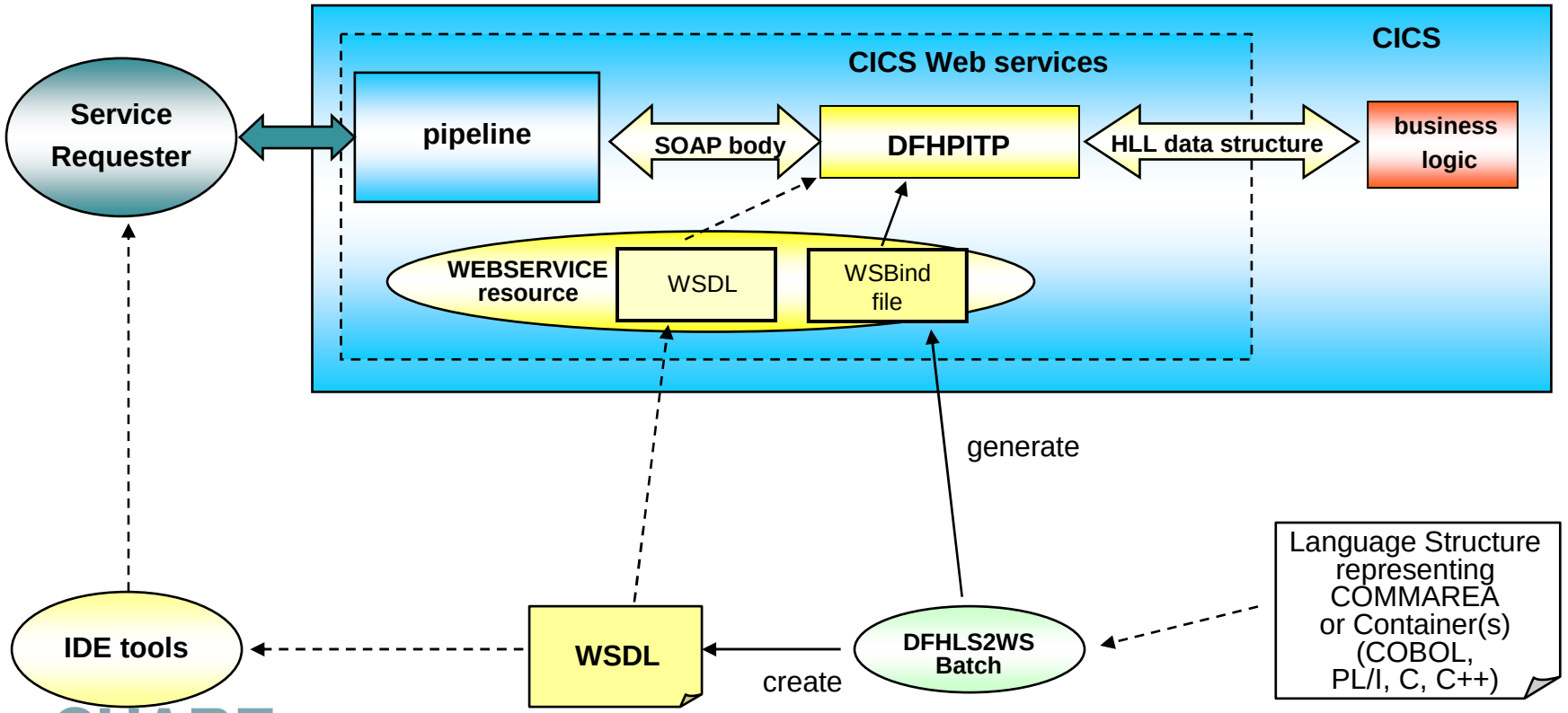


CICS' Web Service Support



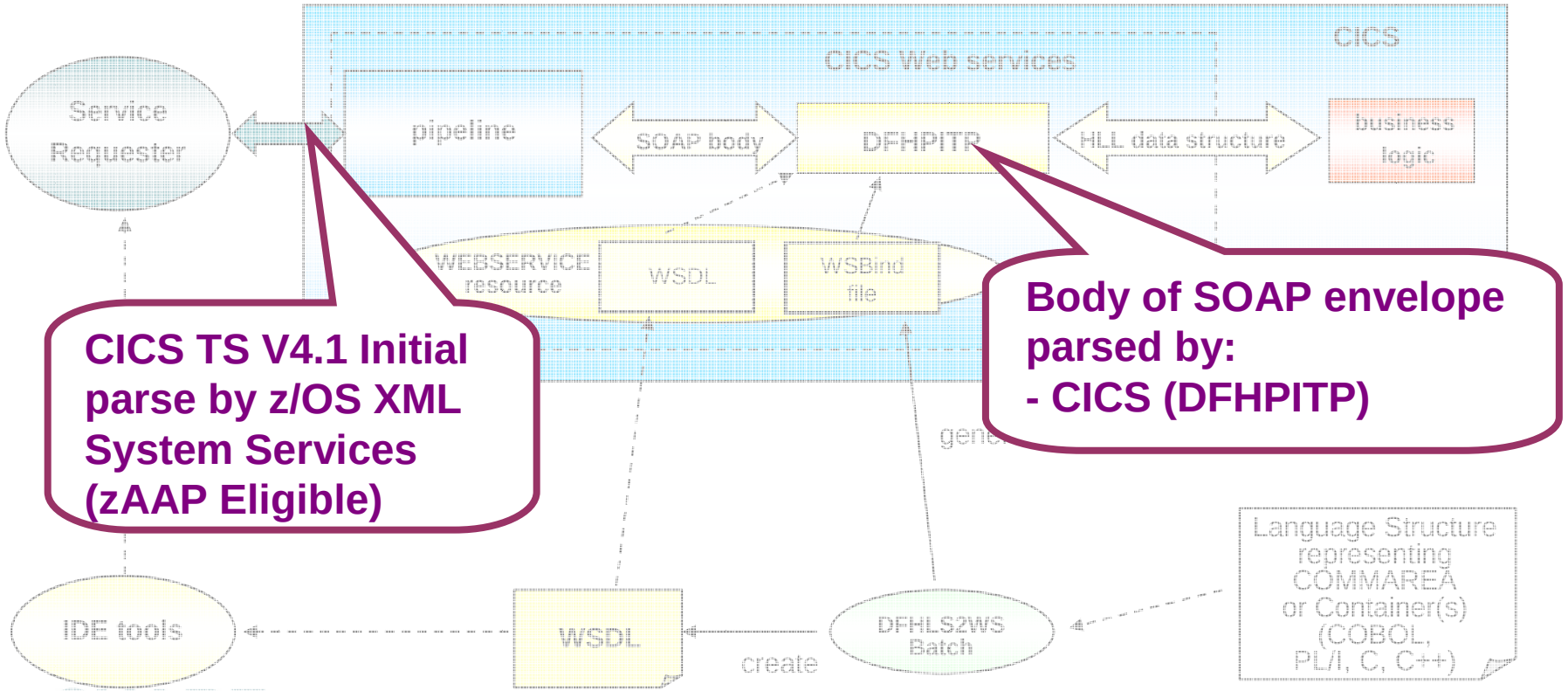
XML with CICS Web Services Assistant

- Batch utilities (DFHWS2LS and DFHLS2WS)
- Initial parse before pipeline processing
  - Done by XMLSS in CICS TS V4.1; zAAP eligible
- Parse of application specific data in body not zAAP eligible



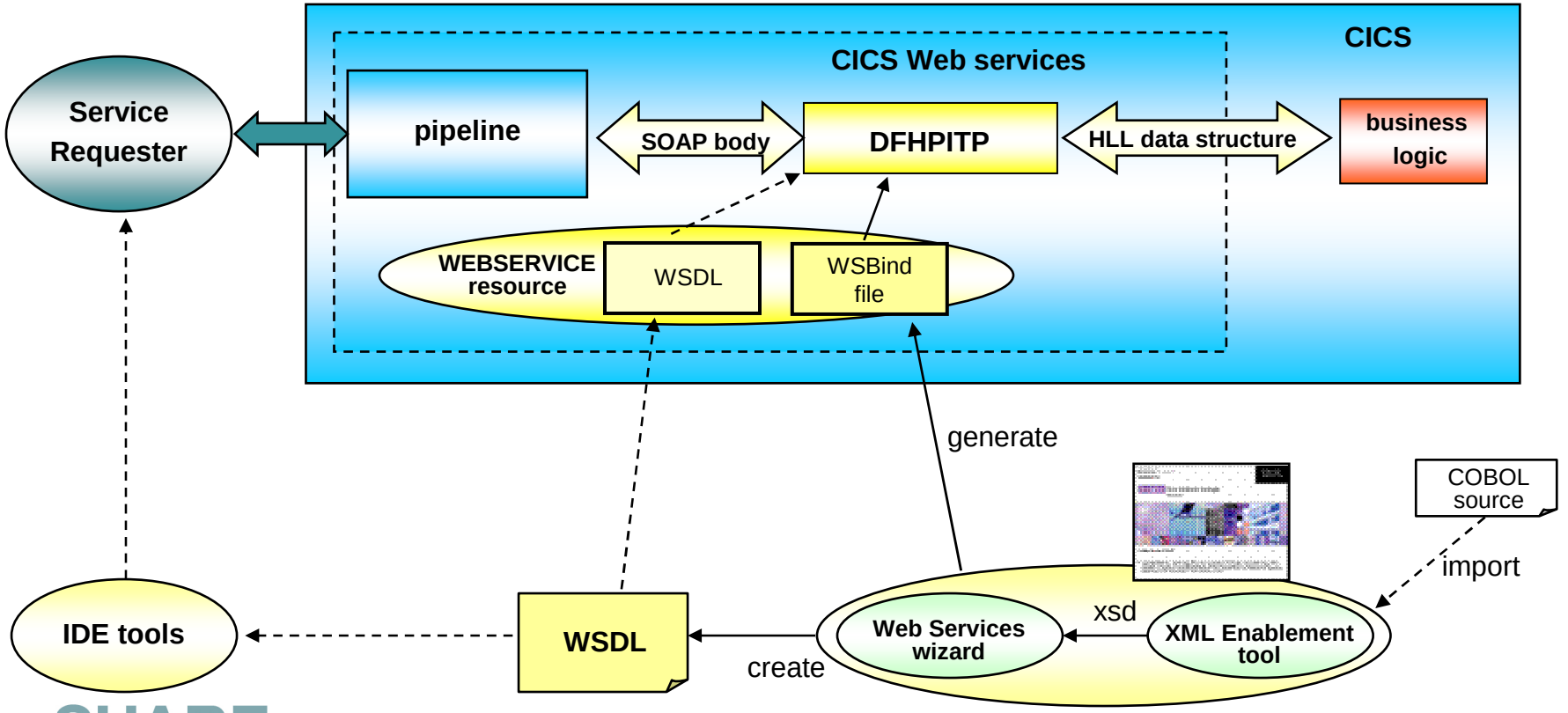
XML with CICS Web Services Assistant

- Batch utilities (DFHWS2LS and DFHLS2WS)
- Initial parse before pipeline processing
  - Done by XMLSS in CICS TS V4.1; zAAP eligible
- Parse of application specific data in body not zAAP eligible



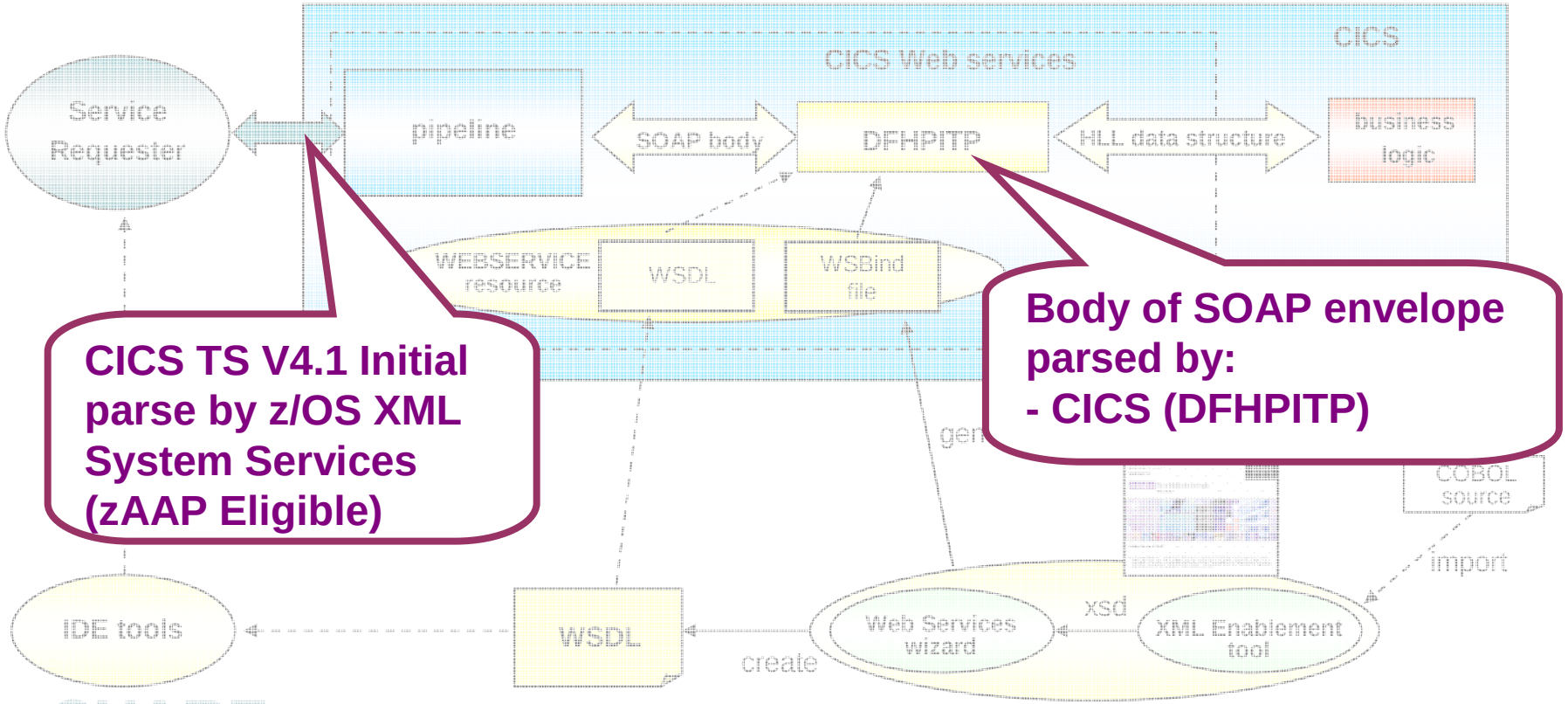
RDz “Interpretive XML Conversion”

- Rational Developer for System z (RDz)
  - Run wizards at workstation – COBOL/PLI input
- Initial parse before pipeline processing
  - Done by XMLSS in CICS TS V4.1; zAAP eligible
- Parse of application specific data in body not zAAP eligible



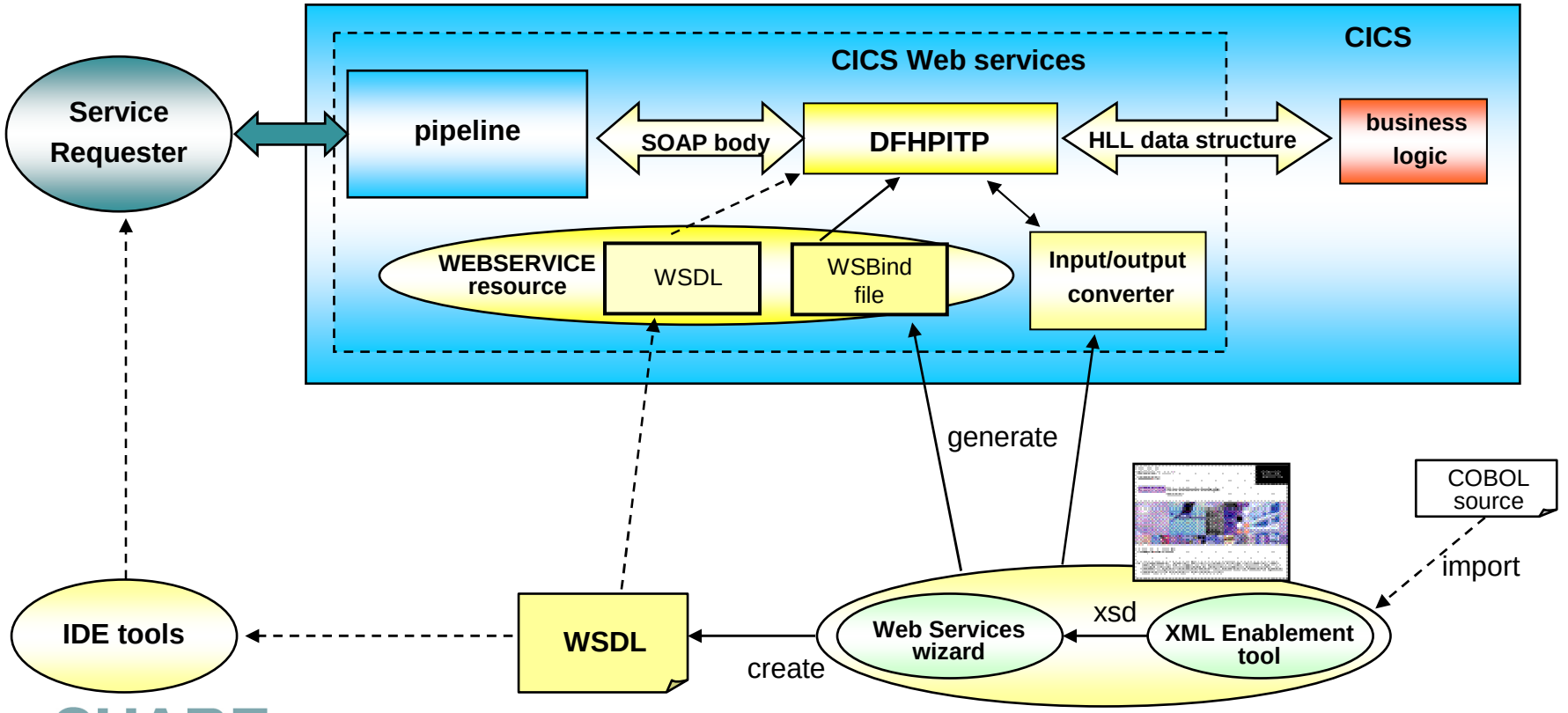
RDz “Interpretive XML Conversion”

- Rational Developer for System z (RDz)
  - Run wizards at workstation – COBOL/PLI input
- Initial parse before pipeline processing
  - Done by XMLSS in CICS TS V4.1; zAAP eligible
- Parse of application specific data in body not zAAP eligible



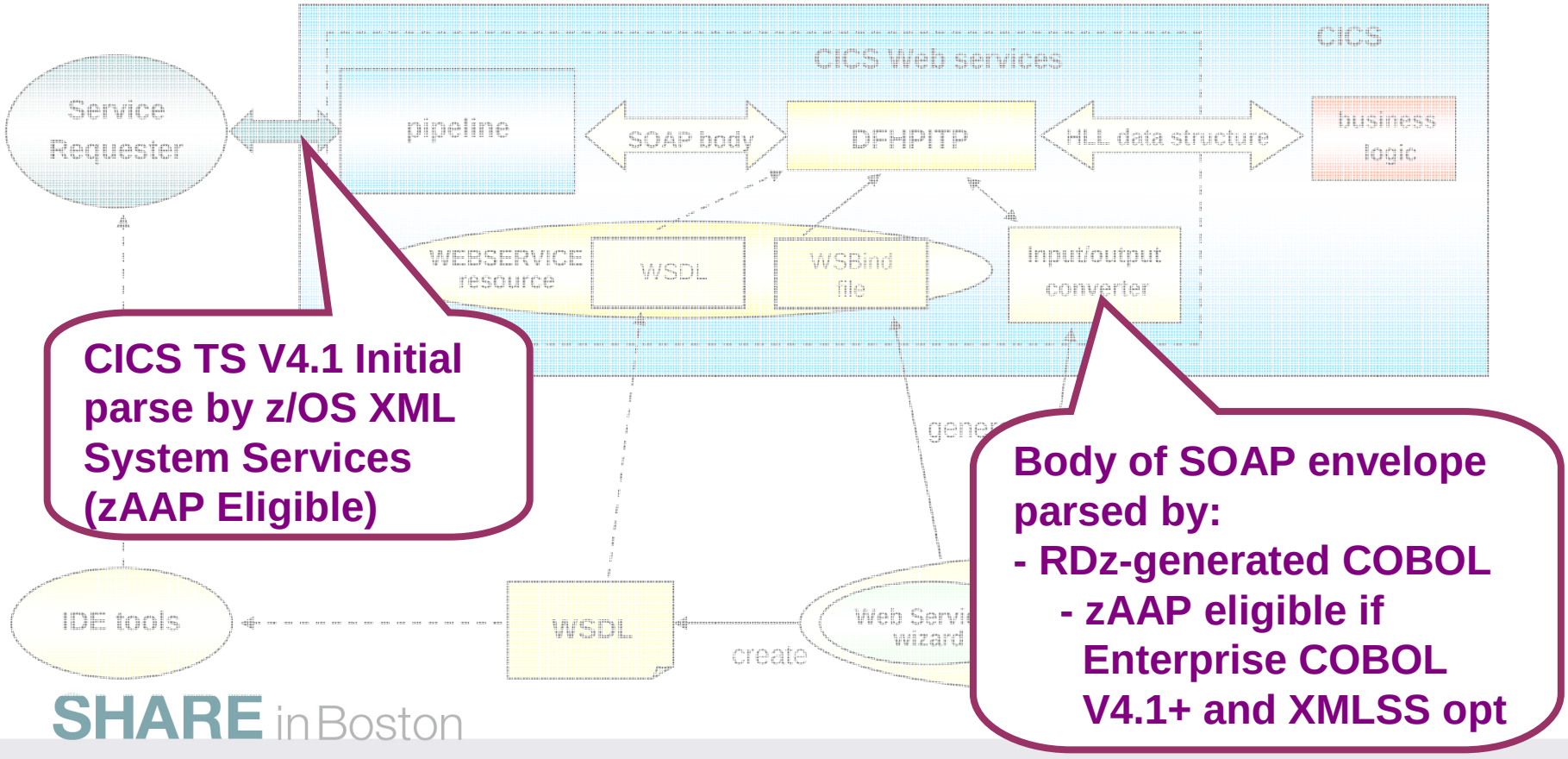
RDz “Compiled XML Conversion”

- RDz generates ‘converter’ programs that contain a COBOL parse verb
  - zAAP eligible when compiled with Enterprise COBOL v4.1 or higher
    - Needs XMLSS compile option
  - Can be used with CICS TX V3 or V4



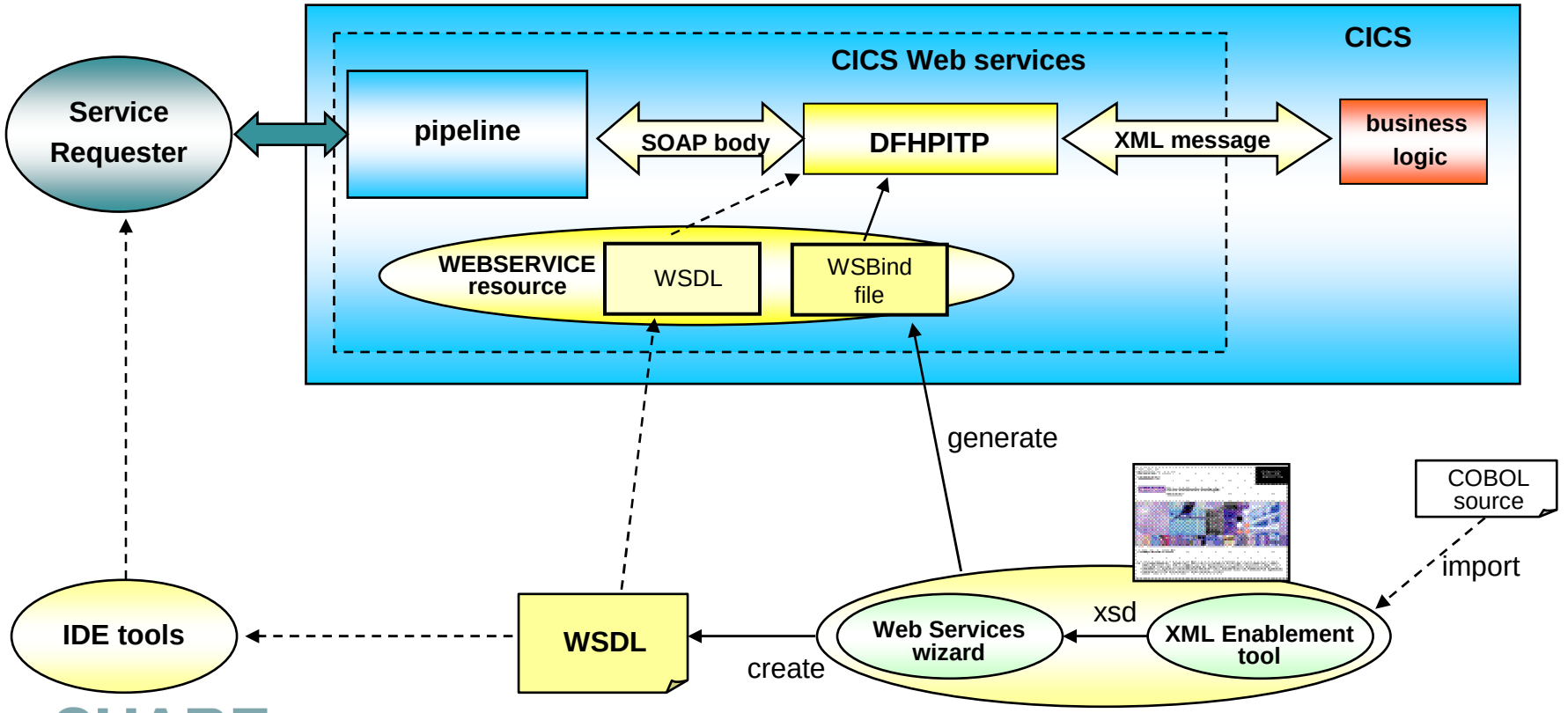
RDz “Compiled XML Conversion”

- RDz generates ‘converter’ programs that contain a COBOL parse verb
  - zAAP eligible when compiled with Enterprise COBOL v4.1 or higher
    - Needs XMLSS compile option
  - Can be used with CICS TX V3 or V4



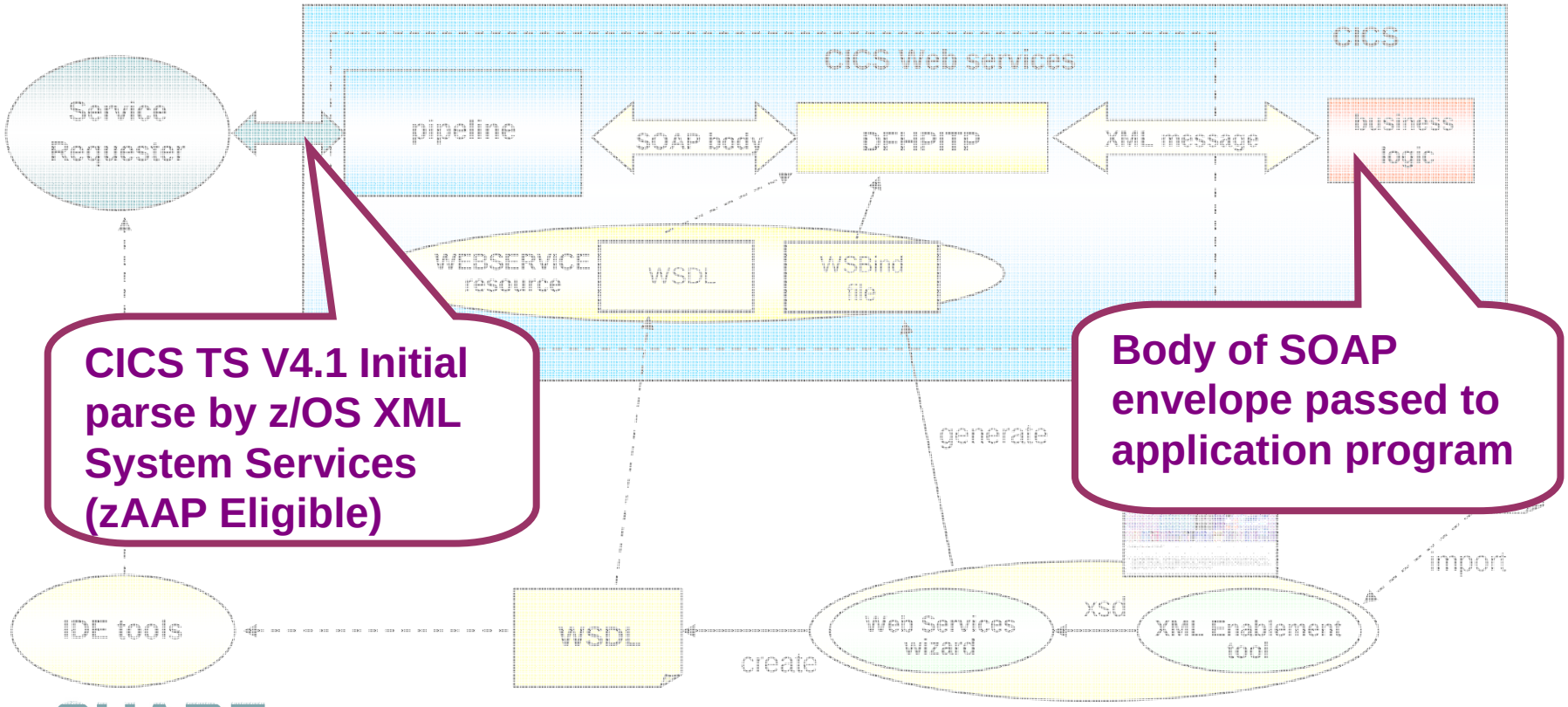
XML with CICS Web Services Assistant XML-ONLY=TRUE

- Can specify XML-ONLY=True with CICS Web Services Assistant or RDz
- MAPPING-LEVEL=2.1 or higher
- SOAP message passed to you, you can use:
  - COBOL PARSE verb, Java (XML4J or others), pass data to DB2 pureXML, use vendor product, or use CICS's TRANSFORM verb



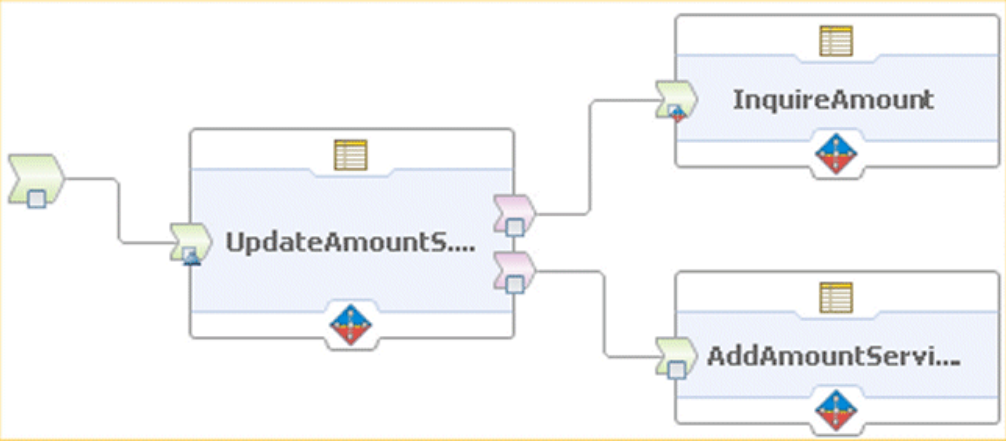
XML with CICS Web Services Assistant XML-ONLY=TRUE

- Can specify XML-ONLY=True with CICS Web Services Assistant or RDz
- SOAP message passed to you, you can use:
  - COBOL PARSE verb, Java (XML4J or others), pass data to DB2 pureXML, use vendor product, or use CICS's TRANSFORM verb



SCA: Service Component Architecture

- **Provide capability to easily develop flexible and reusable CICS application components**
  - Rapid assembly and deployment of new Services
  - Express existing applications as re-usable components
- **Separation of bindings from application code allows flexible infrastructure changes**
- **Reduce skills and effort required to view and manage business applications**



SCA: Service Component Architecture

- Provide capability to easily develop flexible and reusable CICS application components
  - Rapid assembly and deployment of new Services
  - Express existing applications as re-usable components
- Separation of bindings from application code allows flexible infrastructure changes
- Reduce skills and effort required to view and manage business applications

EXEC CICS TRANSFORM

- **Available with CICS TS V4.1**
- **XML to Language Structure or vice versa**
- **Uses an XSDBind file**
  - Created with the CICS XML Assistant or RDz
  - Installed using a BUNDLE resource
  - XSDBind also used with ATOM feeds (not in BUNDLE)
- **Can ask for document root element name**
- **Can request validation**
  - Starts Java program for validation
- **Can use “interpretive XML conversion” approach**
- **Can use “compiled XML conversion” approach**
  - RDz V7.6+

RESTful Interfaces

- **Similar in concept to hyperlinked data**
- **Lightweight data transfer** that **leverage the HTTP protocol**
- **Representational State Transfer**
  - Nouns (URLs) indicate what is being worked on
  - Verbs (GET, PUT, POST, DELETE) indicate the action to be performed (List, Create, Read, Update, Delete)
- **Format of results is not defined**
  - Popular formats of returned data are XML and JSON
- **Approaches in CICS**
  - CICS WEB API
  - ATOM Feed (CICS TS V4.1 and SupportPac CA8K)
  - PHP (SupportPac CA1S)
  - CICS Dynamic Scripting

REST request

Request URI	HTTP Method	Event
Collection URI, e.g.: **`http://xyz.com/prefix/myResource`**	GET	List
	POST	Create
	PUT	putCollection
	DELETE	deleteCollection
Member URI, e.g.: **`http://xyz.com/prefix/myResource/resourceID`**	GET	Retrieve
	POST	postMember
	PUT	update
	DELETE	delete

REST simple sample

- Request

```
GET /mortgage/231677 HTTP/1.1
  Host: www.example.com
  Accept-Language: en
  Charset: UTF-8
```

- Response

or

```
HTTP/1.1 200 OK
  Language: en_us
  Charset: UTF-8
  Content-Type: text/json
{“principal”:”238000”,”rate”:”3.5”, “type”:”5/1 ARM”}
```

```
HTTP/1.1 200 OK
  Language: en_us
  Charset: UTF-8
  Content-Type: text/xml
<mortgage><principal>238000</principal><rate>3.5</rate><type>5/1 ARM</type></mortgage>
```

XML with RESTful Interfaces

- **REST** leverages the HTTP Protocol
- **Request** - **Response** data commonly in XML or JSON
  - XML layout can be documented in an XSD
- **Receive request - return response**
  - CICS Dynamic Scripting or SupportPac CA1S
  - CICS Web Support (EXEC CICS WEB commands)
    - You create the XML:
      - *COBOL GENERATE verb, CICS TRANSFORM verb, CICS DOCUMENT API, read the XML from DB2, whatever*
    - EXEC CICS WEB SEND
- **Can use ATOM** Support to provide RESTful interface
  - CICS TS V4.1 ATOM support or SupportPac CA8K
  - CICS Dynamic Scripting

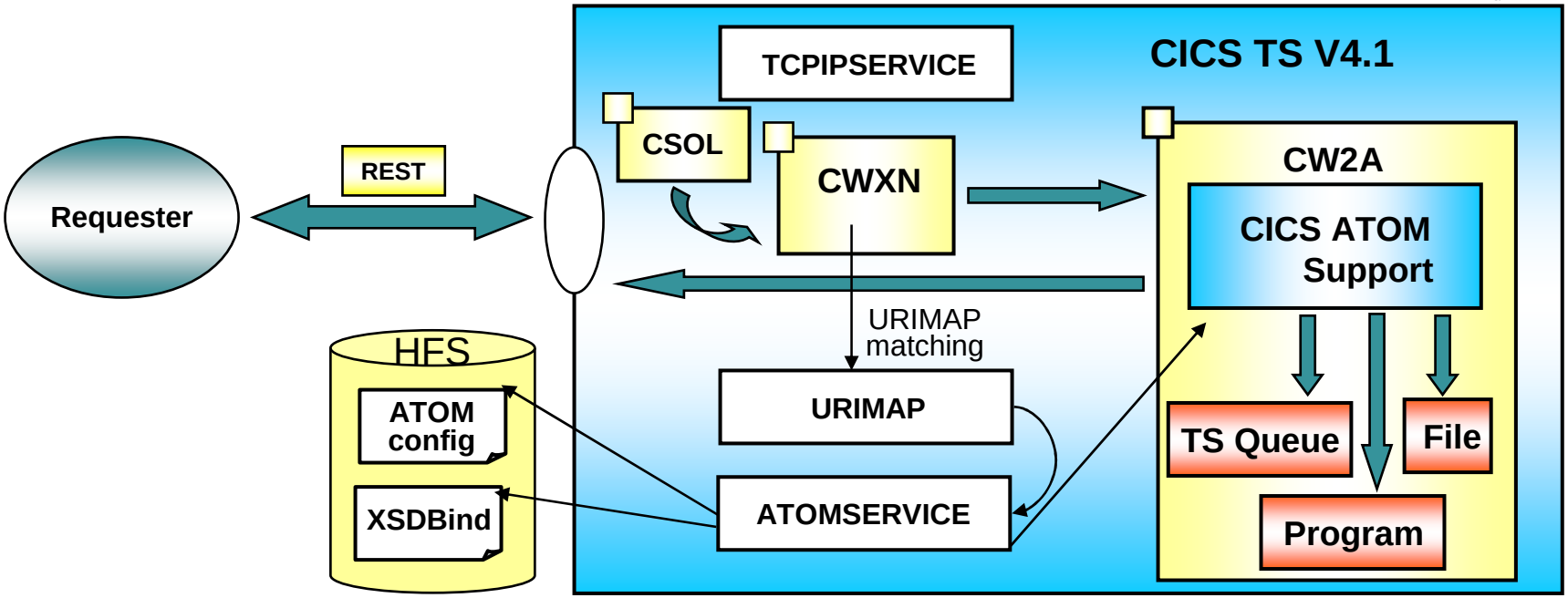
HTTP request with XML

- **REST is more than just returning XML**
  - REST leverages the HTTP protocol, uses HTTP status codes, uses HTTP architected headers, uses commonly accepted approaches to adding filters or search criteria
- **CICS TRANSFORM** verb
  - XML content documented in an XSD
- **COBOL PARSE and GENERATE** verbs
- **Java** classes
- CICS **DOCUMENT API**
- Get the XML from **DB2**
- **Whatever** (COBOL string verb / RYO Assembler)

XML with ATOM Feeds

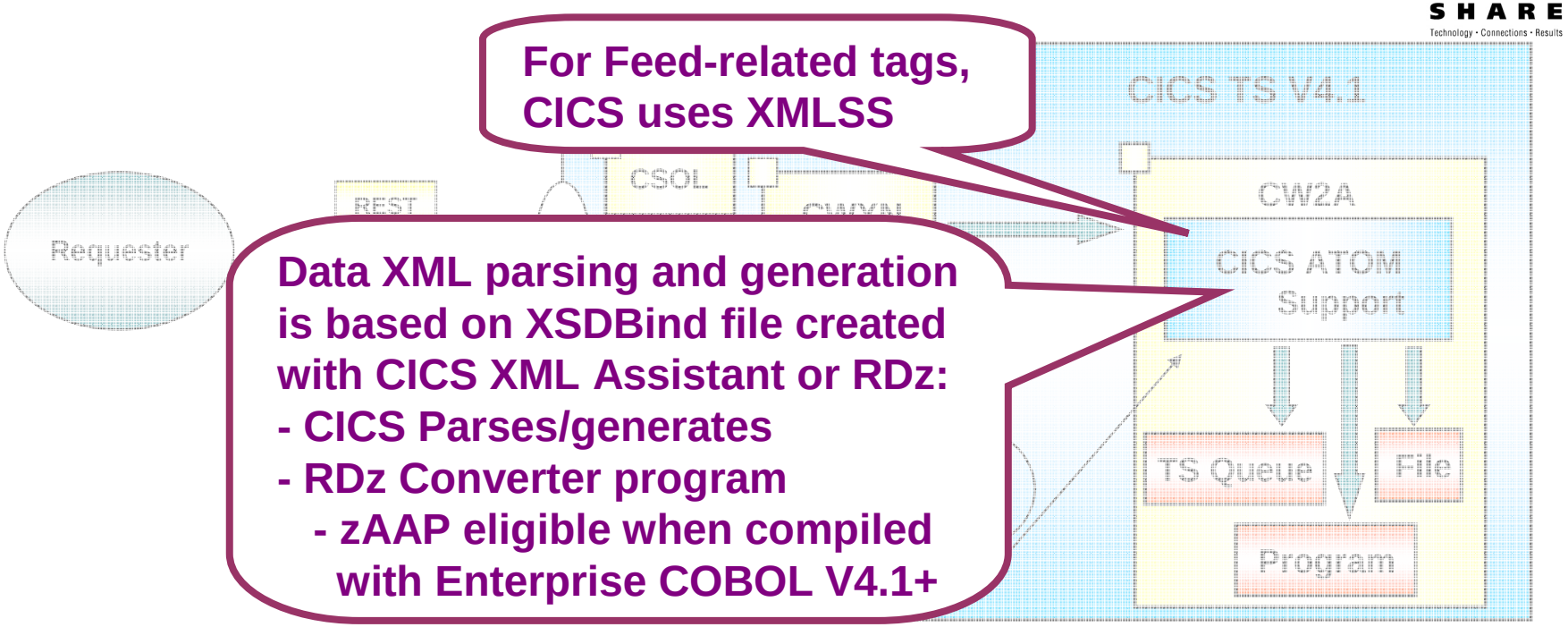
- Expose **VSAM file, TS Queue, or program** as ATOM feed
- Provides a RESTful interface documented in **RFC5023**
- ATOM feeds have a **defined set of XML tags**
  - As per the ATOM Publishing Format (**RFC4287**)
- **General tags** (author, etc) values
  - From your ATOM configuration file
  - From file (using 'unstructured' content)
- **Data values** formulated based on XSDBind file
  - Created with CICS XML Assistant or RDz
  - Same as with TRANSFORM, but not in a BUNDLE
- **Data** (in the content tag in the entry tag) can be documented in an XSD
- **Dynamic Scripting**

XML with ATOM Feeds



- **ATOM configuration file contains**
  - Default values for required tags such as ‘author’
  - Describes SERVICE, CATEGORY, COLLECTION or FEED
  - Contains paths for feed or collection, plus more
- **XSDBind file contains**
  - Data layout information so CICS knows how to encapsulate the data in XML tags

XML with ATOM Feeds



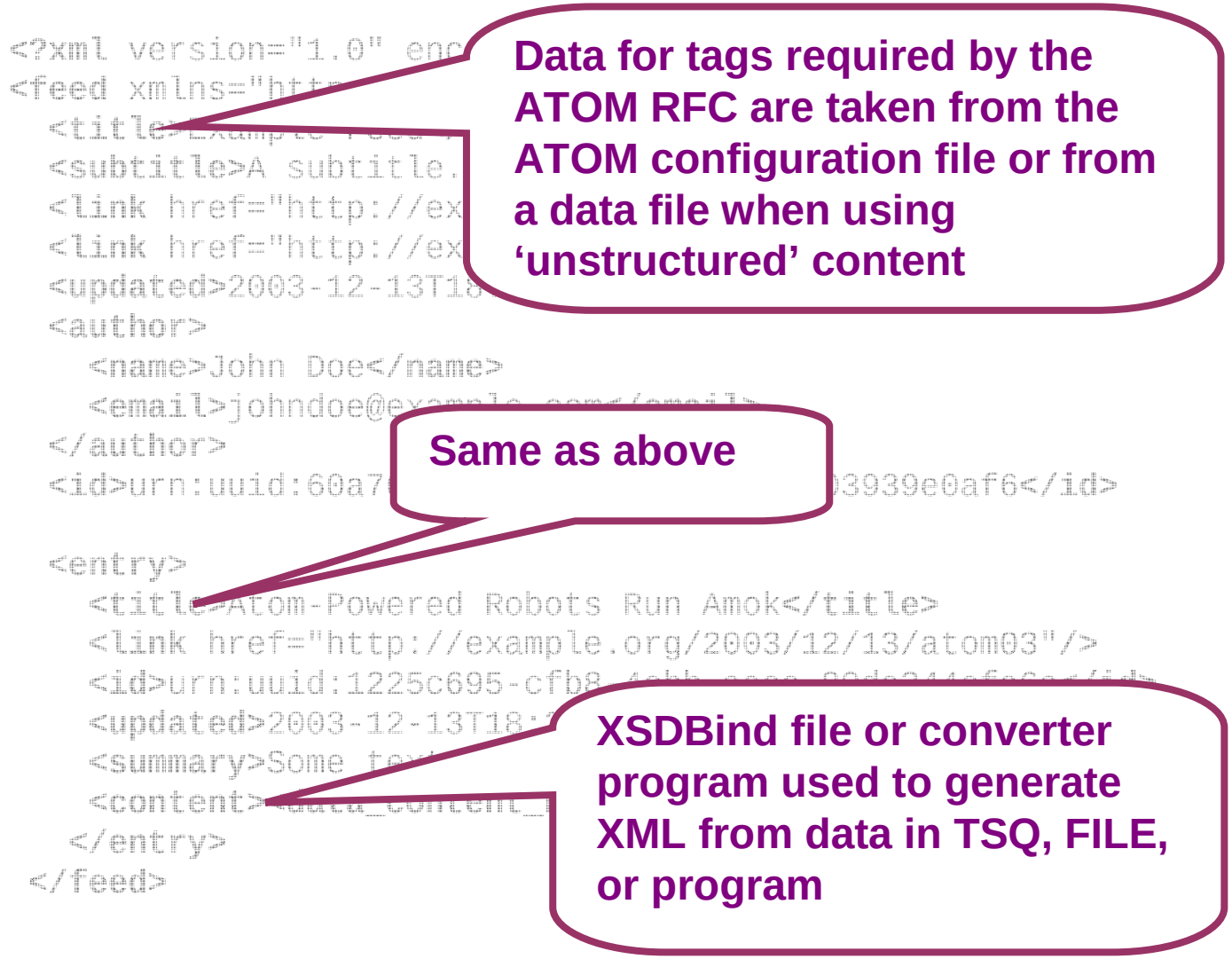
- **ATOM configuration file contains**
  - Default values for required tags such as ‘author’
  - Describes SERVICE, CATEGORY, COLLECTION or FEED
  - Contains paths for feed or collection, plus more
- **XSDBind file contains**
  - Data layout information so CICS knows how to encapsulate the data in XML tags

ATOM: Example XML

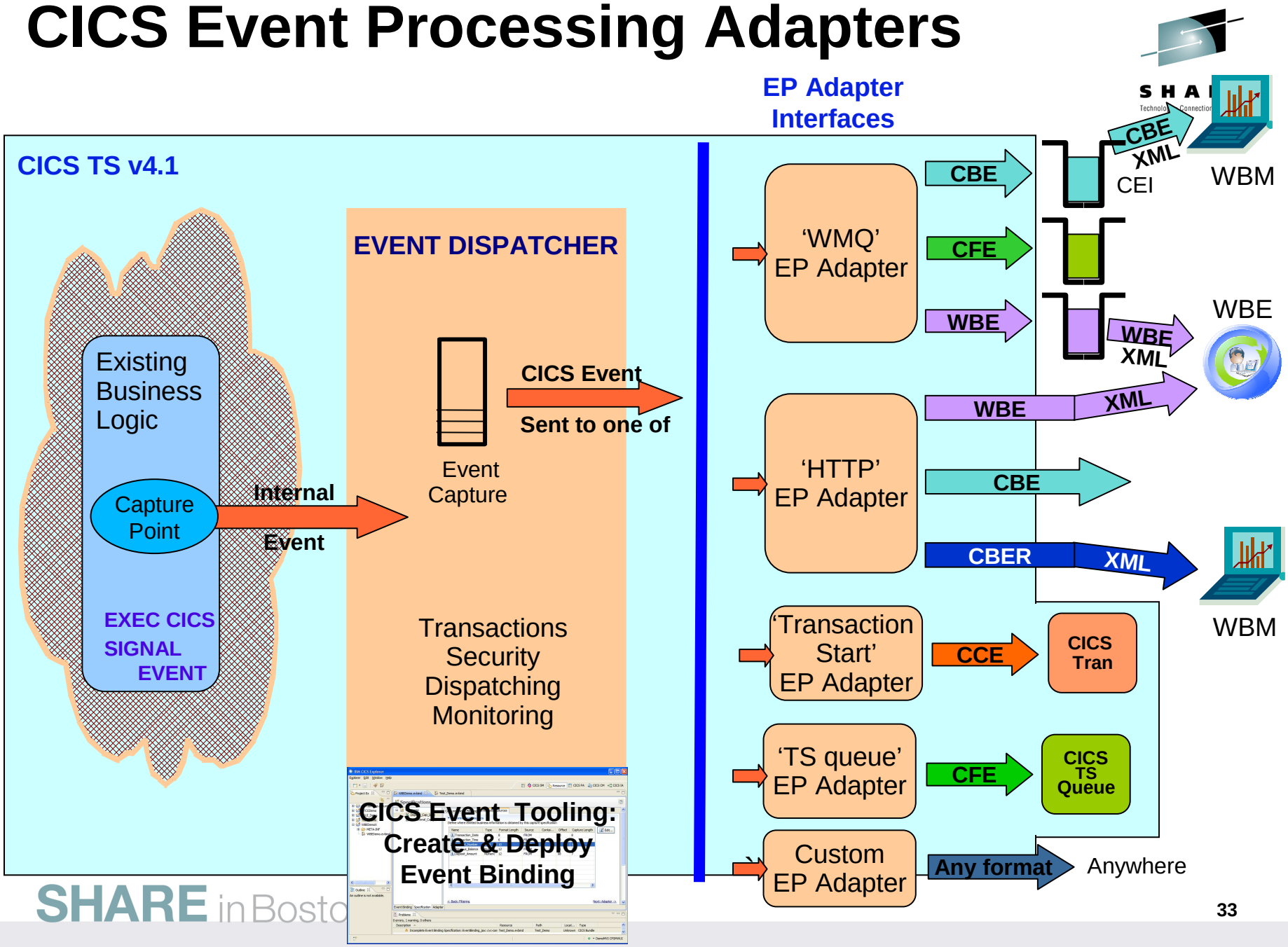
```
<?xml version="1.0" encoding="utf-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">
  <title>Example Feed</title>
  <subtitle>A subtitle.</subtitle>
  <link href="http://example.org/feed/" rel="self"/>
  <link href="http://example.org/"/>
  <updated>2003-12-13T18:30:02Z</updated>
  <author>
    <name>John Doe</name>
    <email>johndoe@example.com</email>
  </author>
  <id>urn:uuid:60a76c80-d399-11d9-b91C-0003939e0af6</id>

  <entry>
    <title>Atom-Powered Robots Run Amok</title>
    <link href="http://example.org/2003/12/13/atom03"/>
    <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
    <updated>2003-12-13T18:30:02Z</updated>
    <summary>Some text.</summary>
    <content><data_content_here/></content>
   </entry>
 </feed>
```

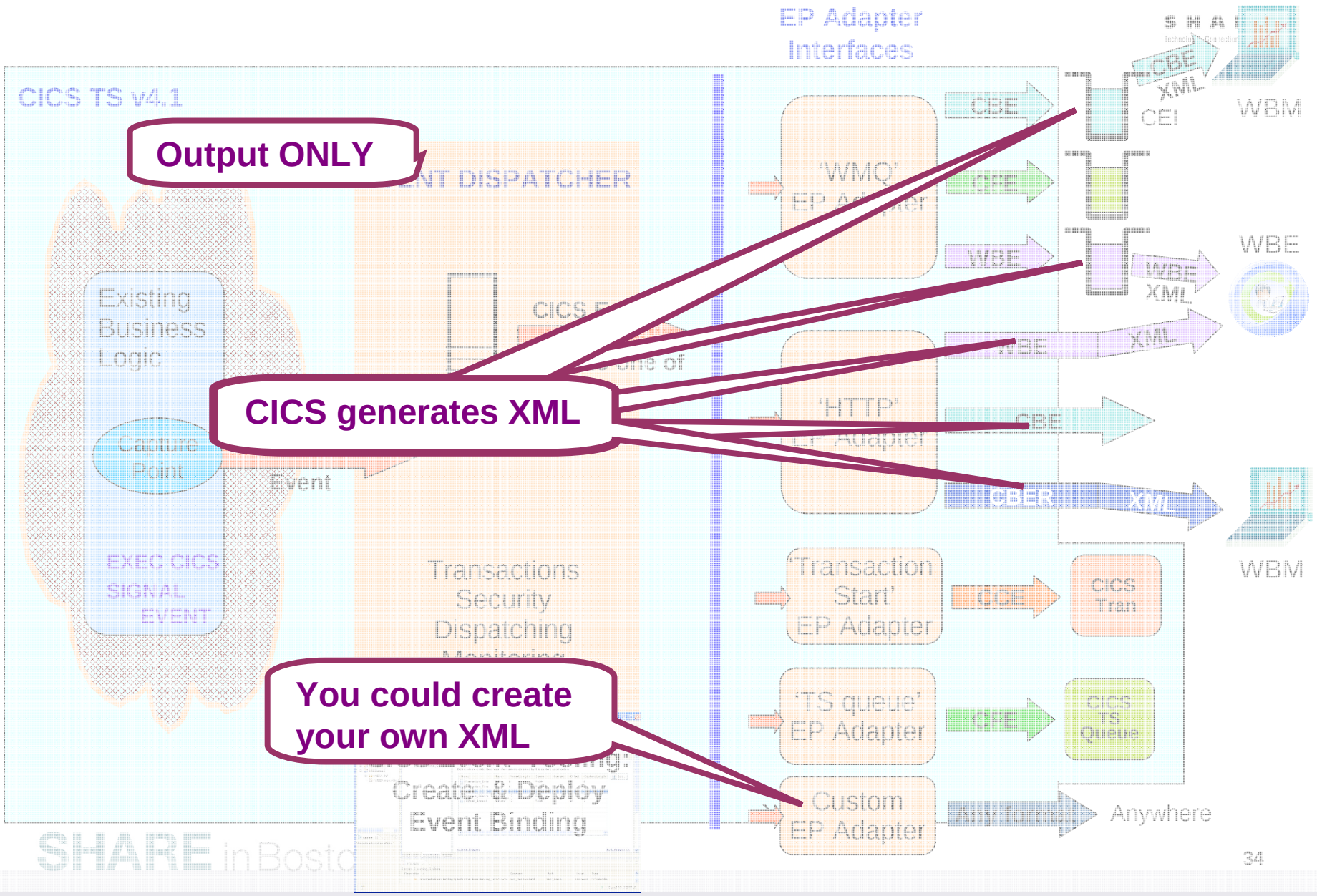
ATOM: Example XML



CICS Event Processing Adapters



CICS Event Processing Adapters



Common Issues: xsd:string

- Mapping level 1.0: assumed to be 255 bytes
- Mapping level 1.1: xsd:string > 32767 mapped to container
- Mapping level 1.2: DEFAULT-CHAR-MAXLENGTH
  - Allowed you to set the length assumption
- Mapping level 1.2: CHAR-VARYING
  - NO: string mapped to fix-length field (same as 1.0)
  - NULL: string mapped to fix-length field with null delimiter
  - yes: generates a field for the length and a field for the container
    - Could specify DEFAULT-CHAR-MAXLENGTH=32768 to have any xsd:string mapped to a container
- Mapping level 1.2: CHAR-VARYING-LIMIT
  - Point at which we map xsd:string to a container
  - CHAR-VARYING-LIMIT=0 will have any xsd:string mapped to a container
- Mapping level 2.1: CHAR-VARYING=COLLAPSE is the default

xsd:string – example 1

```
<element name="getHelloString">
   <complexType>
      <sequence>
         <element name="aString" nillable="true" type="xsd:string"/>
      </sequence>
   </complexType>
</element>
```

```
DFHWS2LS:
    MAPPING-LEVEL=2.1
```

```
05  getHelloString.
    10  aString-length          PIC S9999 COMP-5 SYNC.
    10  aString                 PIC X(255).
    10  attr-nil-aString-value PIC X DISPLAY.
```

This is just <u>one way</u> of working with an xsd:string. There are additional ways. This approach only allows for a maximum xsd:string size of 255. Could use DEFAULT-CHAR-MAXLENGTH to indicate the max allowed length.

```
     EXEC CICS GET CONTAINER(CONTAINERID)
         INTO(getHelloString)
         RESP(RESP-FLD) RESP2(RESP2-FLD)
         END-EXEC.
     If Resp-Fld NOT = DFHRESP(NORMAL)
         EXEC CICS ABEND ABCODE('DDW0') END-EXEC
     End-If.
•Would add code here to check if nillable flag set,
*  If not nil, would do the next line
     String 'The input value was:'
         aString(1:aString-length)
         Delimited by size into MY-Message.
```

xsd:string – example 2

```
<element name="getHelloString">
   <complexType>
      <sequence>
         <element name="aString" nillable="true" type="xsd:string"/>
      </sequence>
   </complexType>
</element>
```

```
DFHWS2LS:
    MAPPING-LEVEL=2.1
    CHAR-VARYING-LIMIT=0
```

```
05  getHelloStringResponse.
    10 getHelloStringReturn-cont     PIC X(16).
    10 attr-nil-getHelloStrin-value  PIC X DISPLAY.
```

```
      EXEC CICS GET CONTAINER(CONTAINERID)
           INTO(getHelloString)
           RESP(RESP-FLD) RESP2(RESP2-FLD)
           END-EXEC.
      If Resp-Fld NOT = DFHRESP(NORMAL)
           EXEC CICS ABEND ABCODE('DDW0') END-EXEC
      End-If.
*   Would add code here to check if nillable flag set,
*   If not nil, would do the next line
      EXEC CICS GET CONTAINER(aString-cont)
           FLENGTH(Received-Container-Len)
           SET(ADDRESS OF Input-Area)
           RESP(Resp-Fld) RESP2(Resp2-FLD)
           END-EXEC.
      If Resp-Fld NOT = DFHRESP(NORMAL)
           EXEC CICS ABEND ABCODE('DDW1') END-EXEC
      End-If.
```

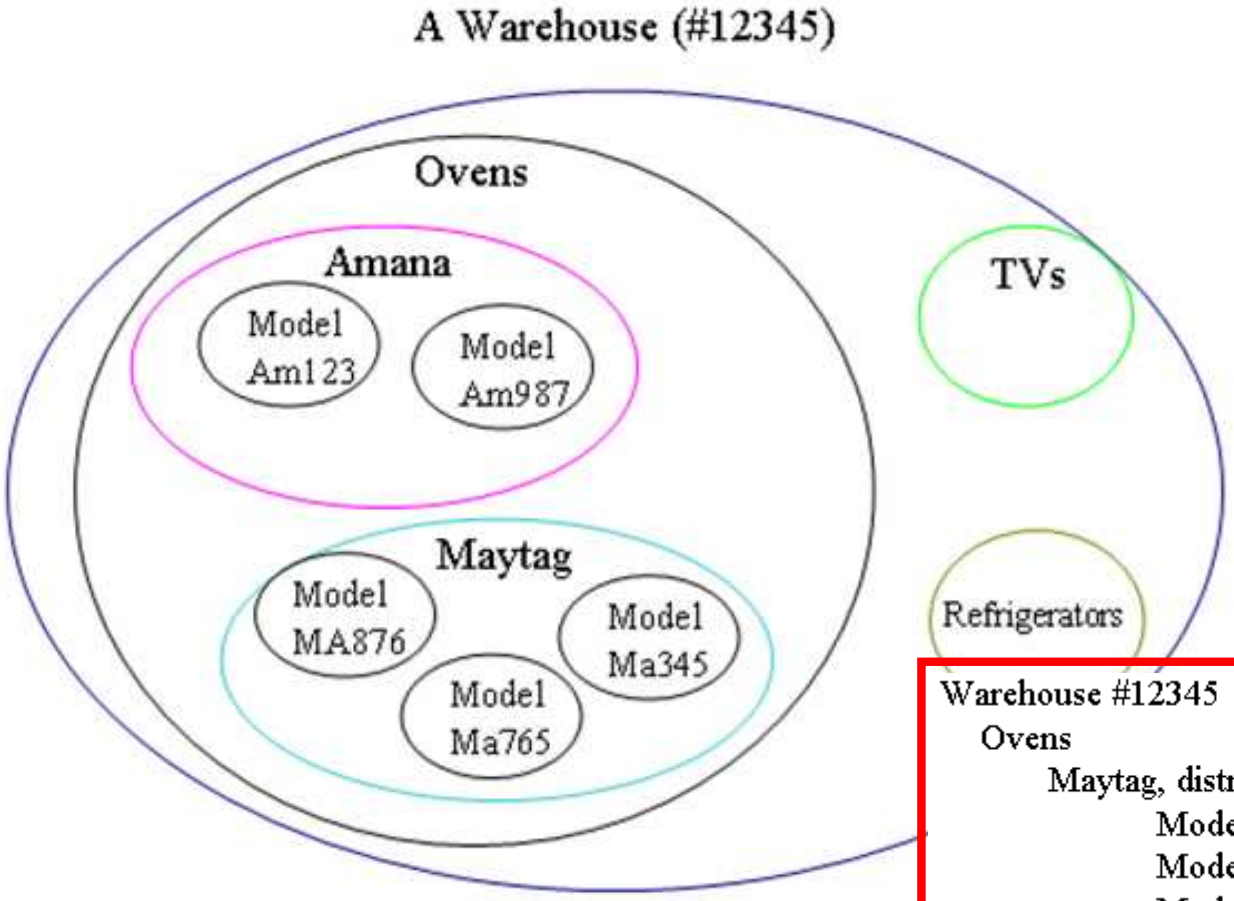
This is just one way of working with an xsd:string. There are additional ways. This approach accommodates an xsd:string of any length.

Common Issues: occurring items

- WSDL may contain **minoccurs** and **maxoccurs**
- DFHWS2LS generates
  - A field for the number of occurrences
  - A field for a container name
  - The container holds all occurrences
- **Mapping level 2.1**
  - Allows you to generate in-line occurring elements
  - **INLINE-MAXOCCURS-LIMIT** (point at which occurs is not in-line)
  - Supports occurring at group level
    - xsd:sequence, xsd:choice, and xsd:all

Occurring items - getOvenInfo

- Scenario – oven info from specified warehouse



Warehouse #12345
  Ovens
    Maytag, distributor=Joes Oven Supply
      Model MA876, price=200, inStock=3
      Model MA345, price=389, inStock=60
      Model MA765, price=1200, inStock=9
    Amana, distributor=Ovens-are-us
      Model AM123, price= 50, inStock=450
      Model AM987, price=487, inStock=67

Occurring Items – get Oven Info

One occurrence of a Model

Model MA876 modelid, price, number in stock

Multiple occurrences of a model

Maytag-Models

MA876 info	MA345 info	MA765 info

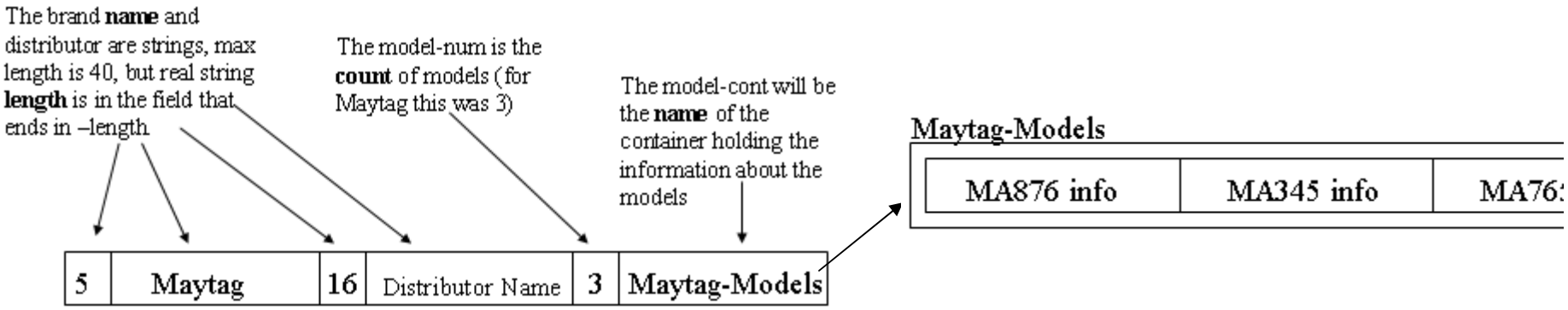
Amana-Models

AM123 info	AM987 info

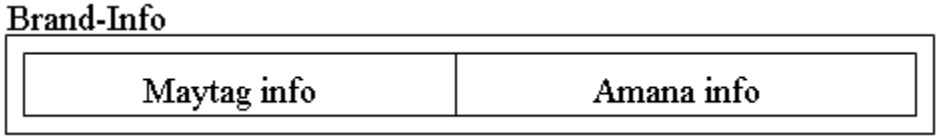
Warehouse #12345
  Ovens
    Maytag, distributor=Joes Oven Supply
      Model MA876, price=200, inStock=3
      Model MA345, price=389, inStock=60
      Model MA765, price=1200, inStock=9
    Amana, distributor=Ovens-are-us
      Model AM123, price= 50, inStock=450
      Model AM987, price=487, inStock=67

Occurring Items – getOvenInfo

```
03  brand.
    06 theName-length          PIC S9999 COMP-5 SYNC.
    06 theName                 PIC X(40).
    06 distributor-length      PIC S9999 COMP-5 SYNC.
    06 distributor             PIC X(40).
    06 model-num               PIC S9(9) COMP-5 SYNC.
    06 model-cont              PIC X(16).
```



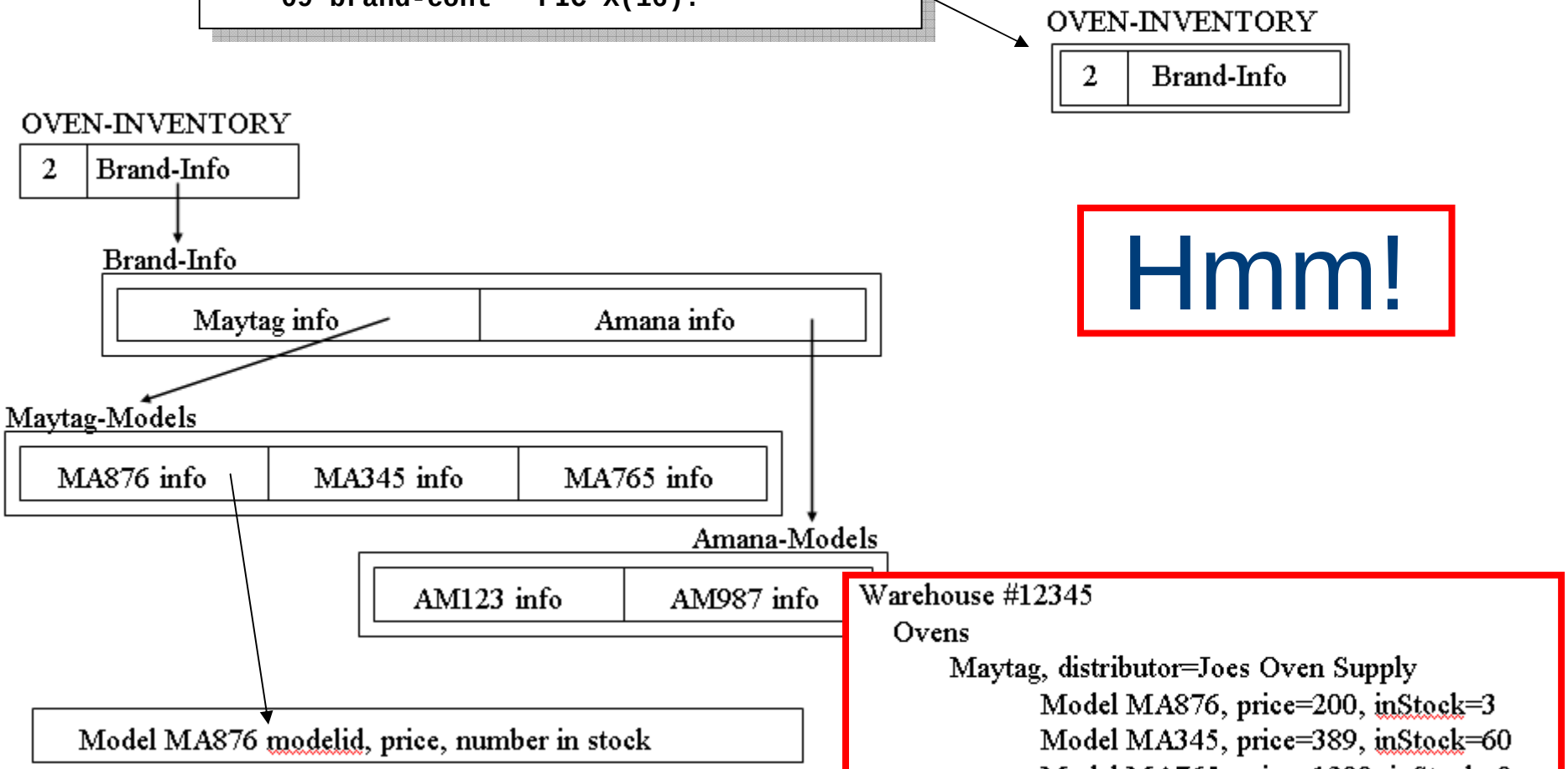
And there are multiple Brands



Warehouse #12345
- Ovens
  - Maytag, distributor=Joes Oven Supply
    - Model MA876, price=200, inStock=3
    - Model MA345, price=389, inStock=60
    - Model MA765, price=1200, inStock=9
  - Amana, distributor=Ovens-are-us
    - Model AM123, price= 50, inStock=450
    - Model AM987, price=487, inStock=67

Occurring Items - getOvenInfo

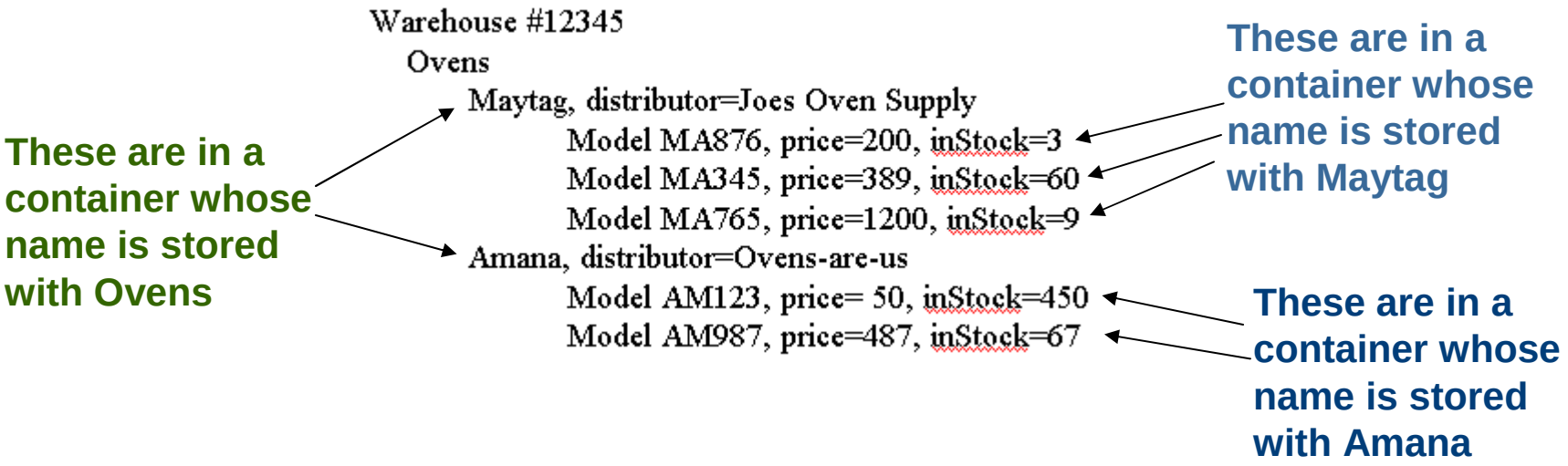
```
06  ovenStock.
    09 brand-num    PIC S9(9) COMP-5 SYNC.
    09 brand-cont   PIC X(16).
```



Hmm!

Occurring Items - getOvenInfo

- A different way to look at it.

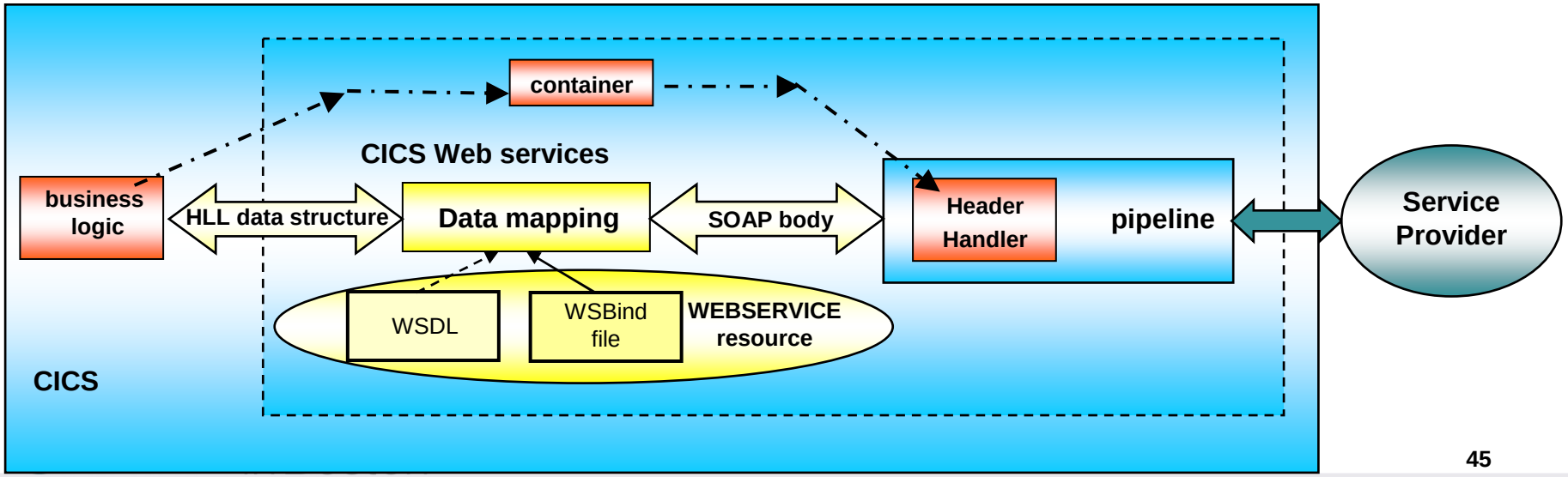


Common Issues: xsd:any xsd:anyType

- **Supported in mapping level 2.1**
- **xsd:any** – a way of specifying undefined content in a Web service message
  - Any well-formed XML
  - One xsd:any per scope, but it can have multiple occurrences
- **xsd:anyType** – base data type from which all simple and complex data types are derived
  - Similar to the way that all Java objects inherit from “Object”
- Generates a field for a container name for data
  - The specified container will contain the data for that element
- Generates a field for a container name for the XMLNSs
  - The specified container will contain the XML namespaces used in the elements data
- Elements with Abstract=“true” are treated like xsd:any

Common Issues: Custom Headers

- Want to specify custom header information based on information in the business logic program….
  - Business logic program (or wrapper) must use channels and containers
  - Business logic program places information to be placed in the header into a container on the current channel (before INVOKE WEBSERVICE)
  - Header handler GETs information from the current channel
    - Constructs a SOAP header
    - Places the SOAP header in the DFHHEADER container
    - CICS places the header in the SOAP message



Mapping Level 3.0 – Multiple Containers

- DFHLS2WS has **REQUEST-CHANNEL=** and **RESPONSE-CHANNEL=**
- **Bottom-up**
- These point to a **channel description document**

A container can have a single element – great for Java (or other languages)

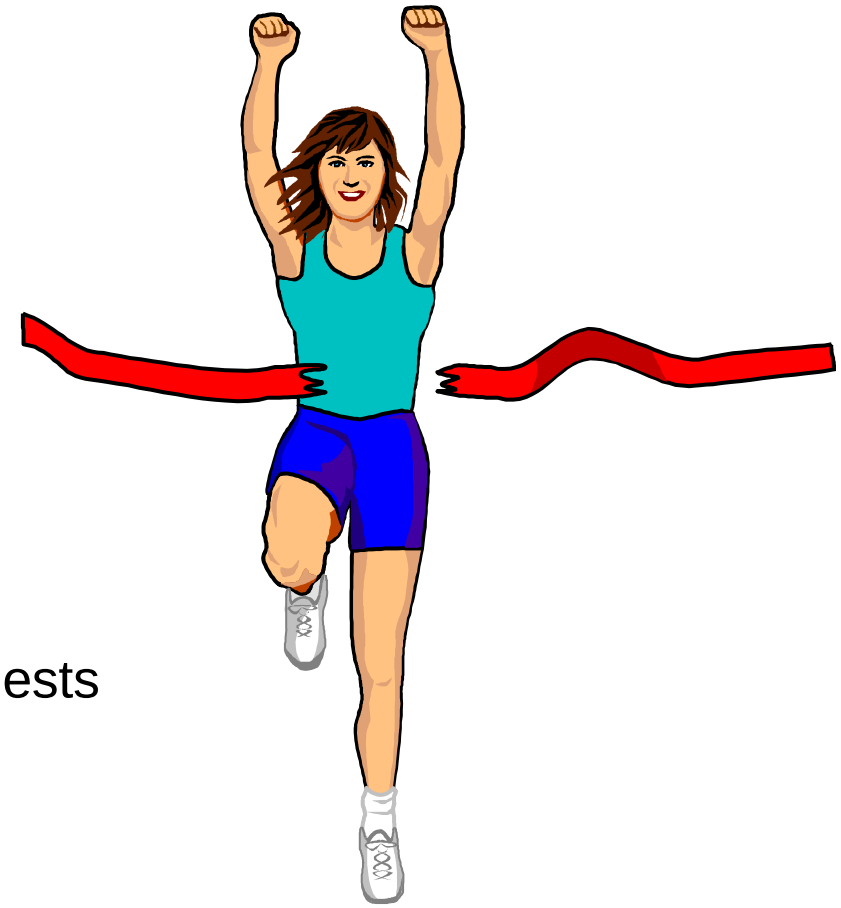
```
<channel name="fund" xmlns=http://www.ibm.com/xmlns/prod/CICS/channel>
  <container name="id" type="char" use="required"></container>
  <container name="name" type="char" use="optional"></container>
  <container name="rating" type="char" use="optional"></container>
  <container name="price" type="char" use="optional"></container>
</channel>
```

A container can have a language structure

```
<channel name="fund" xmlns=http://www.ibm.com/xmlns/prod/CICS/channel>
  <container name="fundAdmin" type="char" use="required">
    <structure location="//WSPOT01.CICSLAB.UTIL(FUNDADMN)"/>
  </container>
  <container name="fundHistory" type="char" use="optional">
    <structure location="//WSPOT01.CICSLAB.UTIL(FUNDHSRY)"/>
  </container>
</channel>
```

Summary

- **Parsing Options**
- **Places where we parse and generate XML in CICS**
  - Web services support
  - EXEC CICS TRANSFORM
  - RESTful Requests
  - ATOM
  - CICS Dynamic scripting
- **Common issues when using CICS's built-in parsing capability**
  - xsd:String
  - Occurring Items
  - Adding headers on Web service requests
  - xsd:any and xsd:anyType
  - Multiple Containers



References

- **Intro to XML:**
  - http://www.ibm.com/developerworks/edu/x-dw-xmlintro-i.html
  - http://www.w3.org/TR/xmlschema-0/
- **XML Schema**
  - http://www.w3.org/TR/xmlschema-1/
  - http://www.w3.org/TR/xmlschema-2/
  - http://www.w3.org/XML/
- **COBOL:**
  - http://www-01.ibm.com/software/awdtools/cobol/zos/about/
- **JAVA**
  - http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/
  - http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=/com.ibm.java.doc.user.lnx.60/user/xml/using_xml.html
  - http://java.sun.com/developer/technicalArticles/releases/j2se15/
- **Java XML/XSLT for z/OS Performance**
  - http://www-03.ibm.com/systems/z/os/zos/tools/xml/perform/performjava.html
- **C++ Parsing**
  - http://www-03.ibm.com/systems/z/os/zos/tools/xml/perform/index.html

References

- DB2 XML Parsing
  - http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp?topic=/com.ibm.db29.doc.xml/db2z_parsexml.htm
- DB2 for z/OS XML Guide – SC18-9858
- Using CICS with DB2 pureXML
  - http://www.ibm.com/developerworks/data/library/techarticle/dm-1004cicsdb2purexml/index.html
- Web Services in CICS
  - http://www.redbooks.ibm.com/abstracts/sg247126.html?Open
  - http://www.redbooks.ibm.com/redpieces/abstracts/sg247206.html