WebSphere software

IBM

e business software

IBM WebSphere Studio Enterprise Developer V5.0

XML Enablement and z/OS Web Services

Mark Evans
WebSphere Studio Enterprise Developer and VAGen Development
evansm@us.ibm.com

IBM Software Group

Agenda

- Brief Introduction to Web Services
- Why a need for z/OS XML Enablement?
- z/OS XML Enablement
- Benefits
- Usage Scenarios
- A look at the XML Enablement tool
- Prerequisites
- Using in a Web Service
- Summary

Web Service Definition

- Self-contained, self-describing, modular applications that can be published, located, and invoked over a network, generally, the Web.
  - Communicate via XML messages
- A web service is composed of:
  - An interface that defines methods (portType)
  - An implementation that will contain the actual business logic and functionality

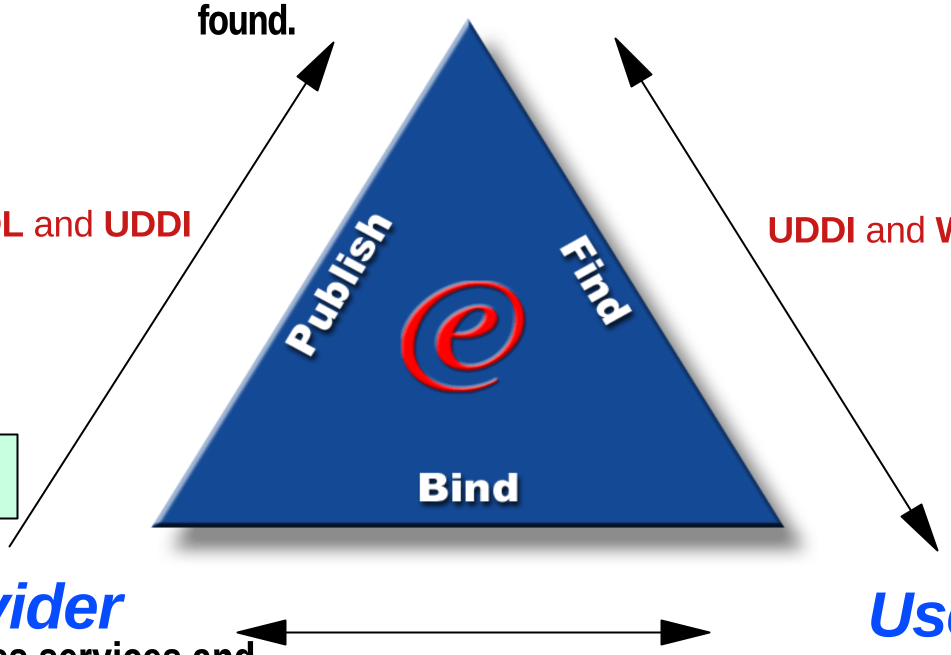
Examples

- Business information with rich content
  - weather reports
  - stock quotes
  - airline schedules
  - credit check
  - news feed
- Transactional Web Services for B2B, B2C
  - airline reservation
  - rental car agreement
  - supply chain mgmt
- Business process externalization
  - business linkage at workflow level
  - complete integration at process level

Web Services Participants and their Roles

Broker

- Maintains Web Services Registry where services are published and found.



WSDL and **UDDI**

1

UDDI and **WSDL**

2

Provider

- **Provides e-business services and PUBLISHES availability of these services to Web Services Registry.**

SOAP

3

User

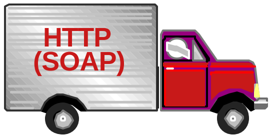
- **FINDS required services in Web Services Registry, BINDS to those services in applications.**

Makes use of XML messaging via Simple Object Access Protocol (SOAP)

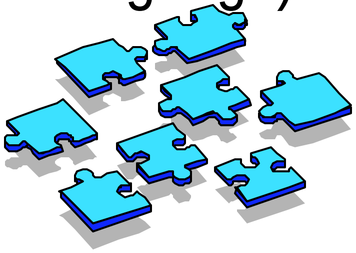
WebSphere software

What is WSDL?

- Web Services Description Language
- WSDL is an XML based vocabulary for defining a Web Service:
  - interfaces
    - operation types (i.e. one-way, request-response, notification)
    - messages defining a Web Service interface
    - definition of data types (XML Schema)
  - access protocol (i.e. SOAP over HTTP)
  - contact endpoints (i.e. Web Service URL and URNs[1])



- A Web Service URL returning WSDL makes Web Services **self-describing**
- Similar in purpose to IDL (Interface Definition Language)
  - From a WSDL file, wizards can generate:
    - proxy classes for calling Web Service
    - skeleton classes to implement a Web Service



What is SOAP?

- Simple Object Access Protocol
- SOAP is an XML based protocol for communication between two remote applications:
  - is based on RPC messaging
  - is language independent (**de-couples** interface from implementation)
  - represents remote procedure calls and responses
- A SOAP message consists of:
  - envelope
    - wraps the message itself
    - defines rules for decoding the message
  - message
    - request
      - method to invoke on a remote object and parameters
    - response
      - result of running the method and exceptions

Problems Solved by Web Services

- Allows heterogeneous systems to communicate with each other
  - i.e. A windows client could access a web service running on a main frame
- Customers can find services based upon the service type
  - Similar to finding a list of landscapers in the phone book
- Allows different organizations to communicate
  - Enables applications to talk with each other

XML Enablement for z/OS

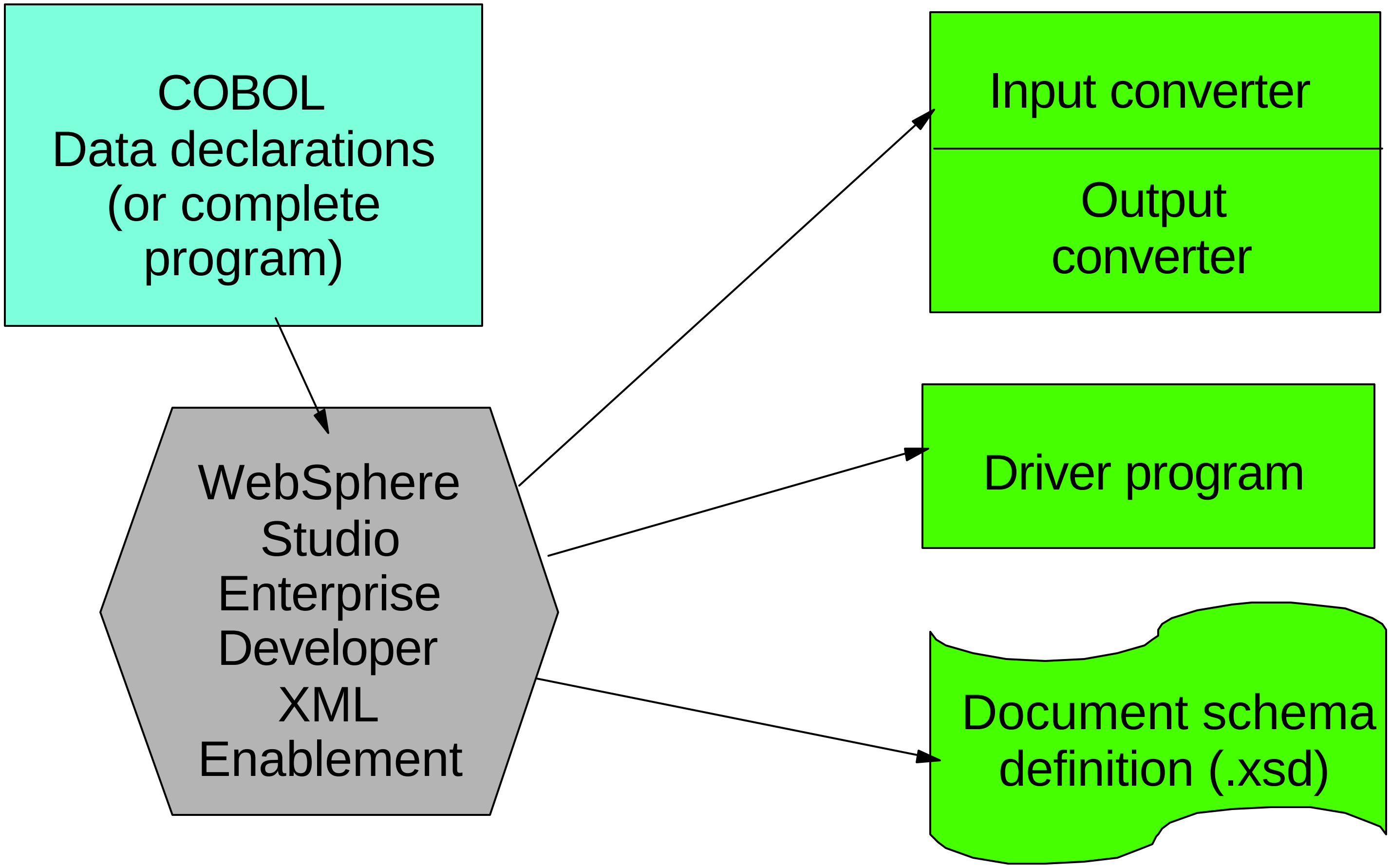
What is z/OS XML Enablement?

Enables COBOL-based applications to consume and produce XML messages

- Leverages XML parsing capabilities of IBM Enterprise COBOL V3.1
- Creates COBOL converter programs
  - Inbound to convert XML messages into native COBOL data
  - Outbound to convert native COBOL data into XML messages
- Creates template COBOL driver program
  - Illustrates the invocation of converters
  - Illustrates the interaction with existing application
  - Needs to be updated before run
- Enables communication with XML based systems

XML Enablement

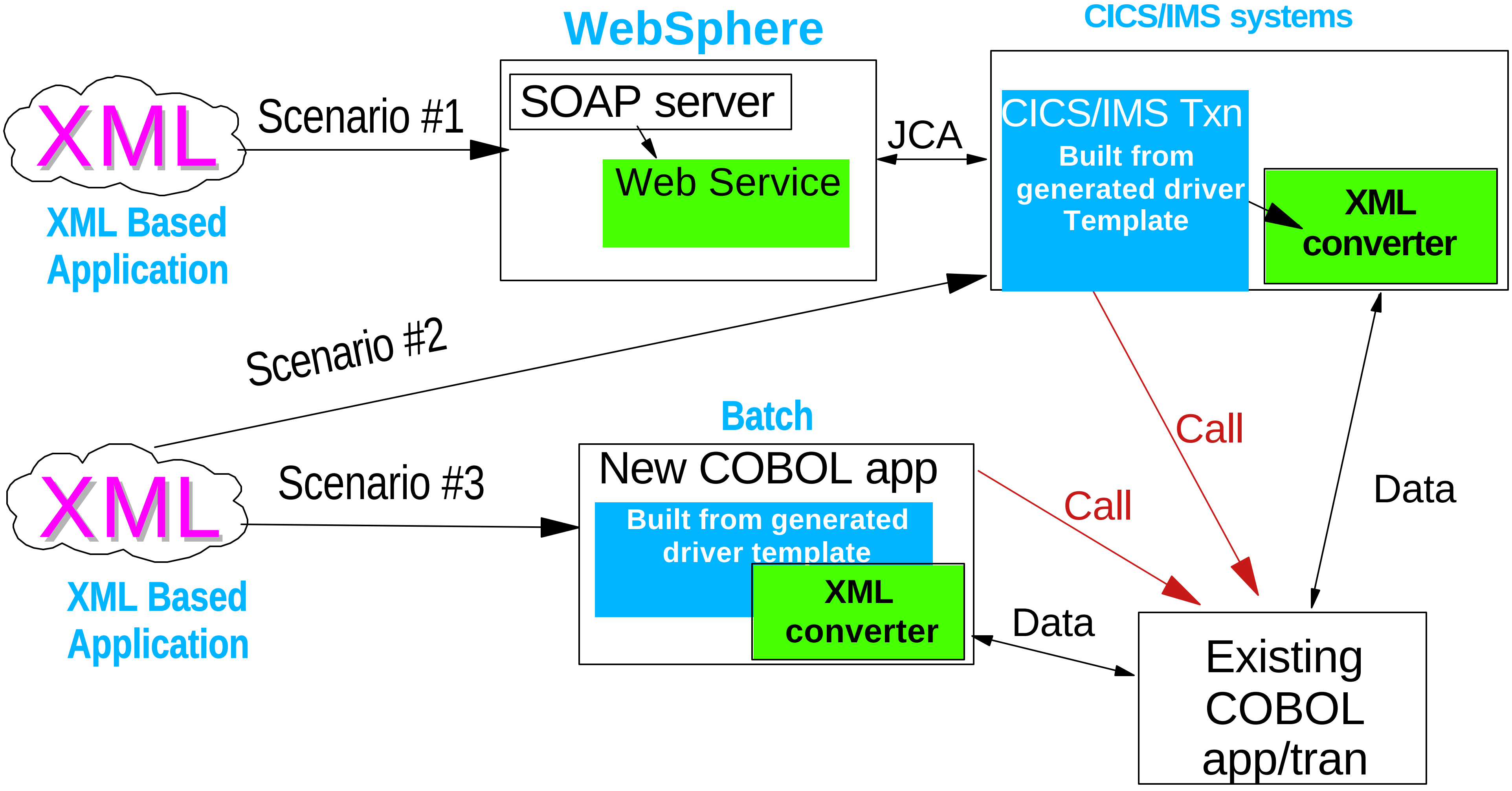
- Enables COBOL-based applications to consume and produce XML messages
  - Original COBOL program unchanged



Benefits of XML Enablement

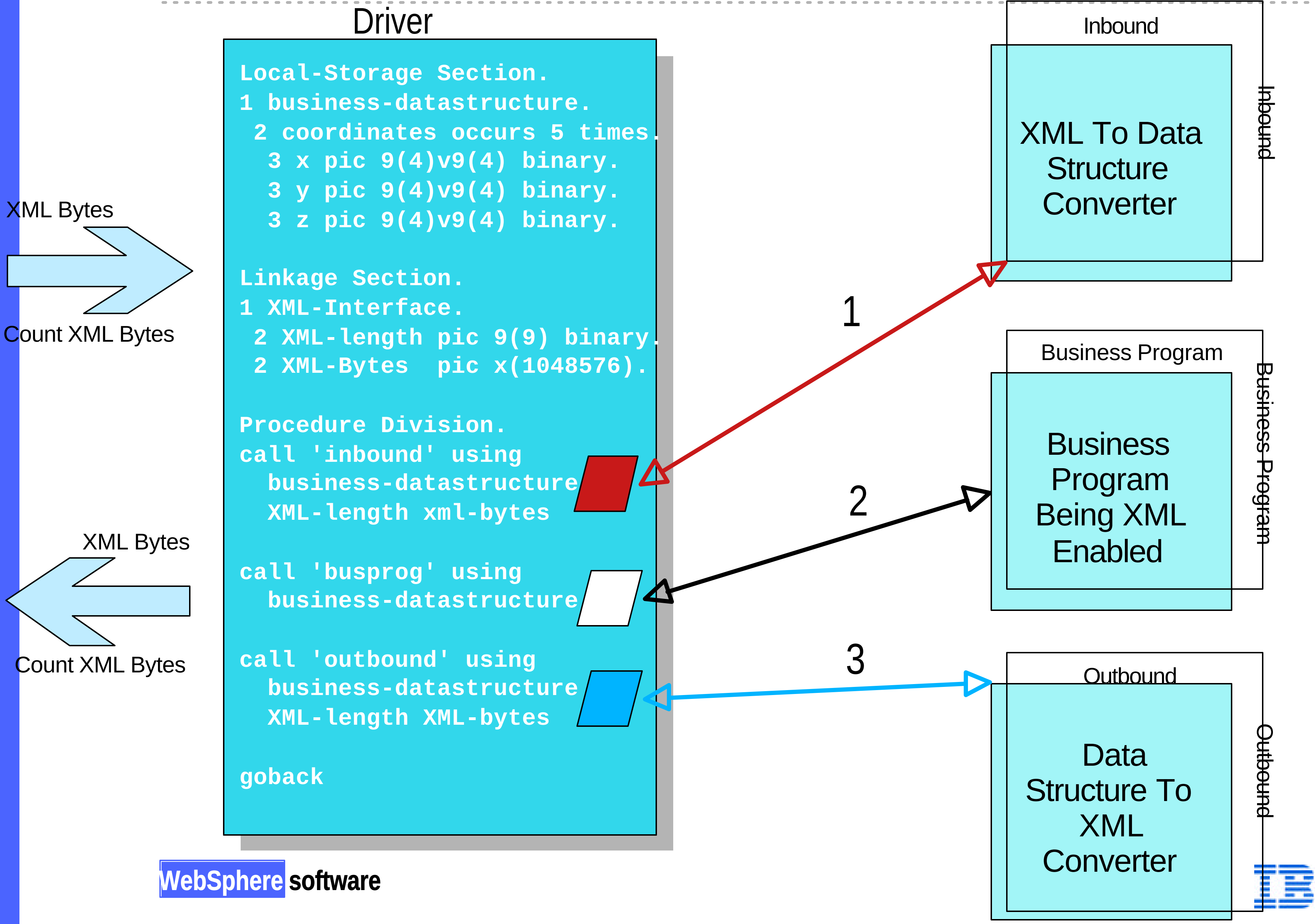
- **Enterprise Modernization:**
  - ►Easy to "reface" existing COBOL applications to support XML messages
- **Programmer Productivity:**
  - ►Converter programs are generated to easily convert between XML and COBOL datatypes
  - ►Template program generated which illustrates how converter programs are used with existing COBOL
  - ►Exploits customers' existing assets/skills/literacy
- **Performance**
  - ►XML parsing/conversions run on z/OS
- **Eliminates the requirement of having a rigid, binary interface for legacy COBOL and PLI programs.**
- **Supports multiple runtime scenarios**
  - ►Including web services

XML Enablement - Runtime Scenarios



z/OS

Using the XML Converters



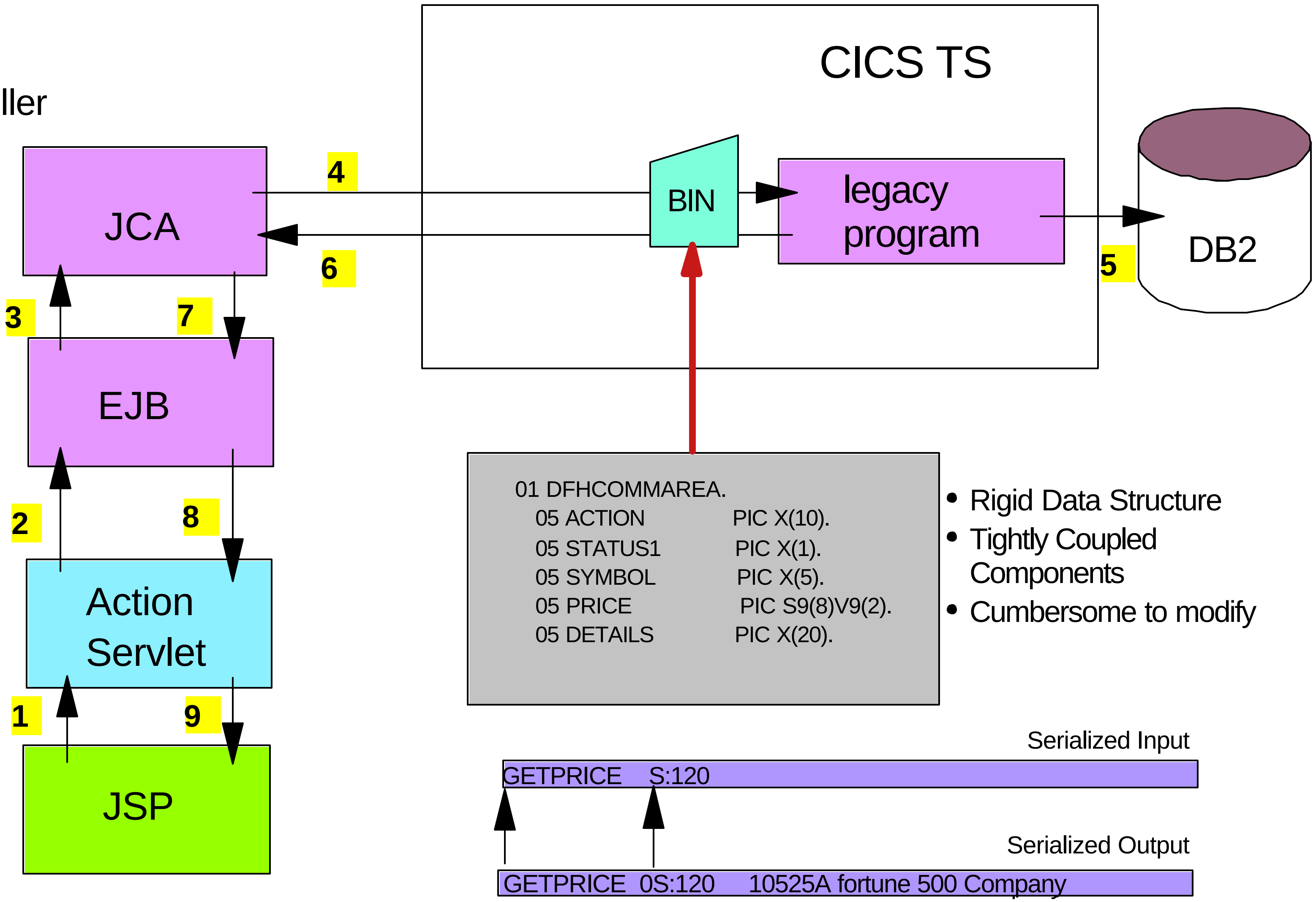
```
Local-Storage Section.
1 business-datastructure.
 2 coordinates occurs 5 times.
  3 x pic 9(4)v9(4) binary.
  3 y pic 9(4)v9(4) binary.
  3 z pic 9(4)v9(4) binary.

Linkage Section.
1 XML-Interface.
 2 XML-length pic 9(9) binary.
 2 XML-Bytes  pic x(1048576).

Procedure Division.
call 'inbound' using
  business-datastructure
  XML-length xml-bytes

call 'busprog' using
  business-datastructure

call 'outbound' using
  business-datastructure
  XML-length XML-bytes

goback
```

Without z/OS XML Enablement - Java Enabled CICS

🟪 Model

🟩 View

🟦 Controller



```
01 DFHCOMMAREA.
   05 ACTION           PIC X(10).
   05 STATUS1          PIC X(1).
   05 SYMBOL           PIC X(5).
   05 PRICE            PIC S9(8)V9(2).
   05 DETAILS          PIC X(20).
```

- Rigid Data Structure
- Tightly Coupled Components
- Cumbersome to modify

Serialized Input

`GETPRICE    S:120`

Serialized Output

`GETPRICE  0S:120     10525A fortune 500 Company`

When input to legacy program is XML

```
<?xml version="1.0" encoding="UTF-8"?>
<DFHCOMMAREA>
      <action>GETPRICE</action>
      <status1>0</status1>
      <symbol>s:100</symbol>
      <price>88.25</price>
      <details>A Fortune 500
Company</details>
</DFHCOMMAREA>
```

- Self Describing Data (loosely coupled)

- De-couples Caller (e.g. EJB and JCA for legacy CICS txn)
  - ▶ These components are just dealing with a "string buffer"

- Easily enhance...client sends additional fields and CICS transaction can process or ignore

General Limitations

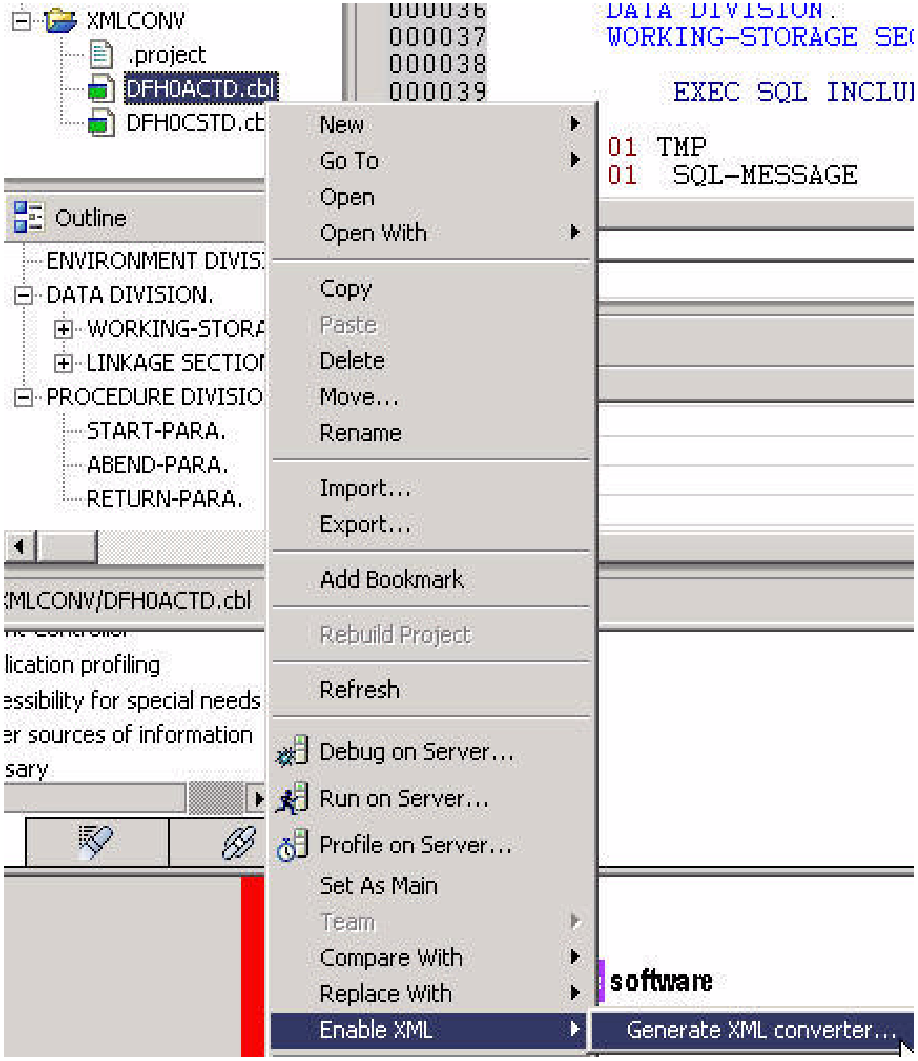
- Workbench
  - MVS Project cannot be source and target (must use local project)
  - Copy books must be fully expanded
- z/OS Runtime
  - Usage "COMP-X" not supported
  - Error handling via Language Environment exceptions
  - Mapping of XML Element attributes not supported
  - REDEFINING items are ignored
- Inbound message processing
  - Occurs-Depending-On (ODO) is supported
    - No validation that group repetitions don't exceed **depending on** variable
  - Entire XML message must be scanned
- Outbound message generation
  - Complex Occurs-Depending-On (ODO) not supported

Early Availability Limitations

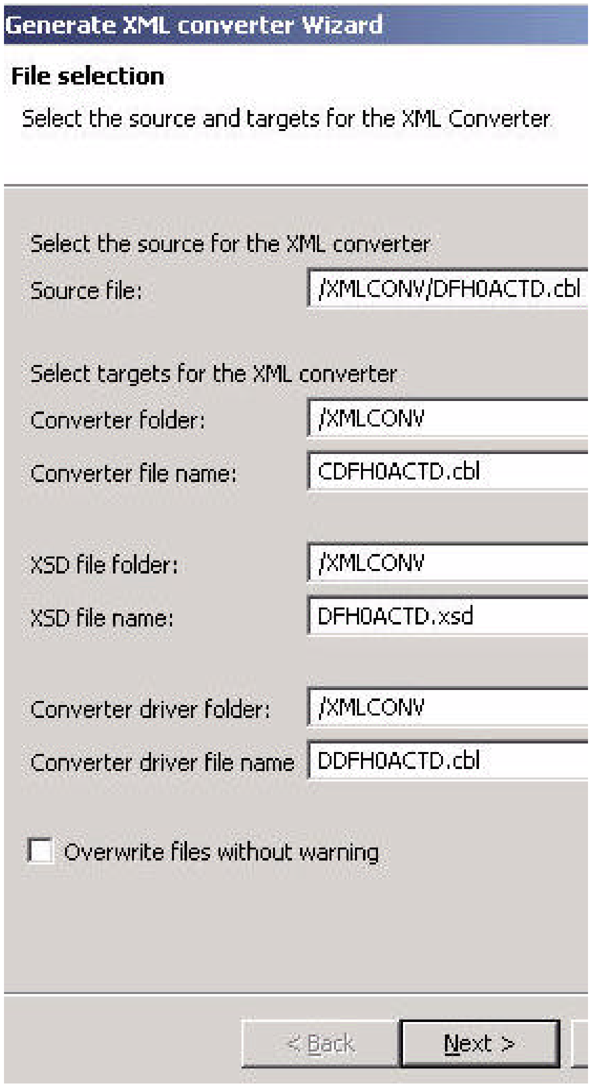
- Workbench
  - No online help
    - ***XML for the Enterprise*** white paper
- Inbound message processing
  - Unicode UTF-16 is not supported
- Outbound message generation
  - Simple Occurs-Depending-On (ODO) not supported
  - Trailing/leading blanks in character content not removed
  - Trailing/leading zeroes in numeric content not removed
  - **<**, **>**, **'**, **"**, **&** not allowed in character content

All of these removed at General Availability

Using the Generate XML Converter Wizard



```
000036          DATA DIVISION.
000037          WORKING-STORAGE SE
000038
000039              EXEC SQL INCLU
                01 TMP
                01  SQL-MESSAGE
```



Generate XML converter Wizard

File selection

Select the source and targets for the XML Converter

Select the source for the XML converter

Source file: /XMLCONV/DFH0ACTD.cbl

Select targets for the XML converter

Converter folder: /XMLCONV

Converter file name: CDFH0ACTD.cbl

XSD file folder: /XMLCONV

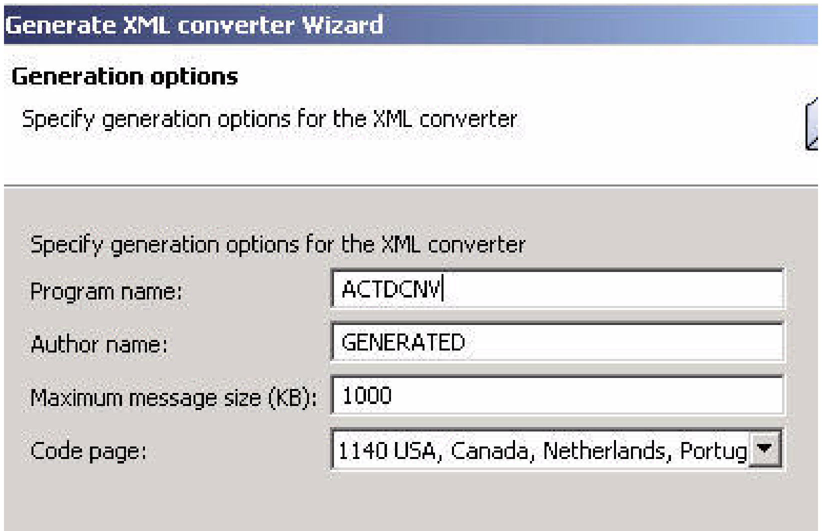
XSD file name: DFH0ACTD.xsd

Converter driver folder: /XMLCONV

Converter driver file name DDFH0ACTD.cbl

☐ Overwrite files without warning

Using the Generate XML Converter Wizard ...



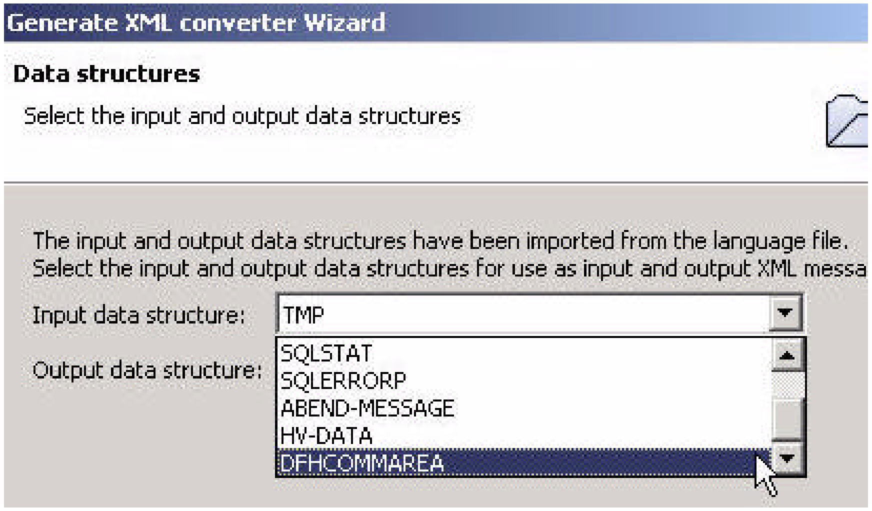
Generate XML converter Wizard

Generation options

Specify generation options for the XML converter

Specify generation options for the XML converter

Program name:	ACTDCNV
Author name:	GENERATED
Maximum message size (KB):	1000
Code page:	1140 USA, Canada, Netherlands, Portug



Generate XML converter Wizard

Data structures

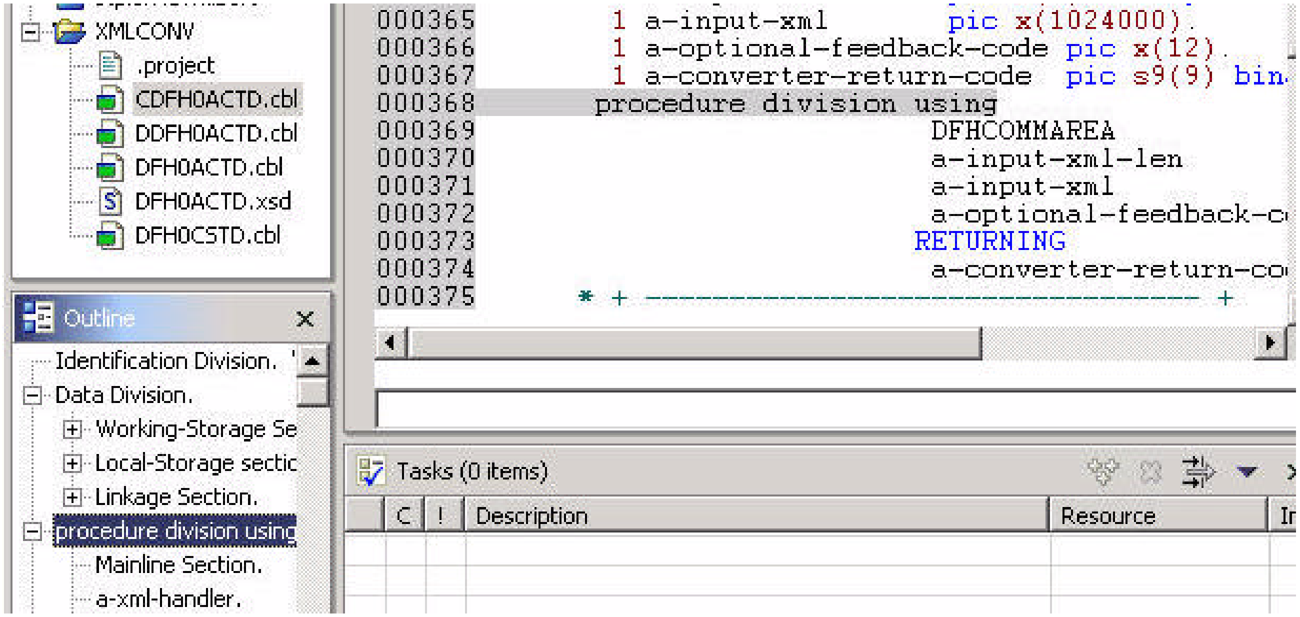
Select the input and output data structures

The input and output data structures have been imported from the language file.
Select the input and output data structures for use as input and output XML messa

Input data structure: TMP

Output data structure:

- SQLSTAT
- SQLERRORP
- ABEND-MESSAGE
- HV-DATA
- DFHCOMMAREA



```
000365           1 a-input-xml          pic x(1024000).
000366           1 a-optional-feedback-code pic x(12).
000367           1 a-converter-return-code  pic s9(9) bin
000368       procedure division using
000369                            DFHCOMMAREA
000370                            a-input-xml-len
000371                            a-input-xml
000372                            a-optional-feedback-c
000373                           RETURNING
000374                            a-converter-return-co
000375      * + ------------------------------------ +
```

Tasks (0 items)

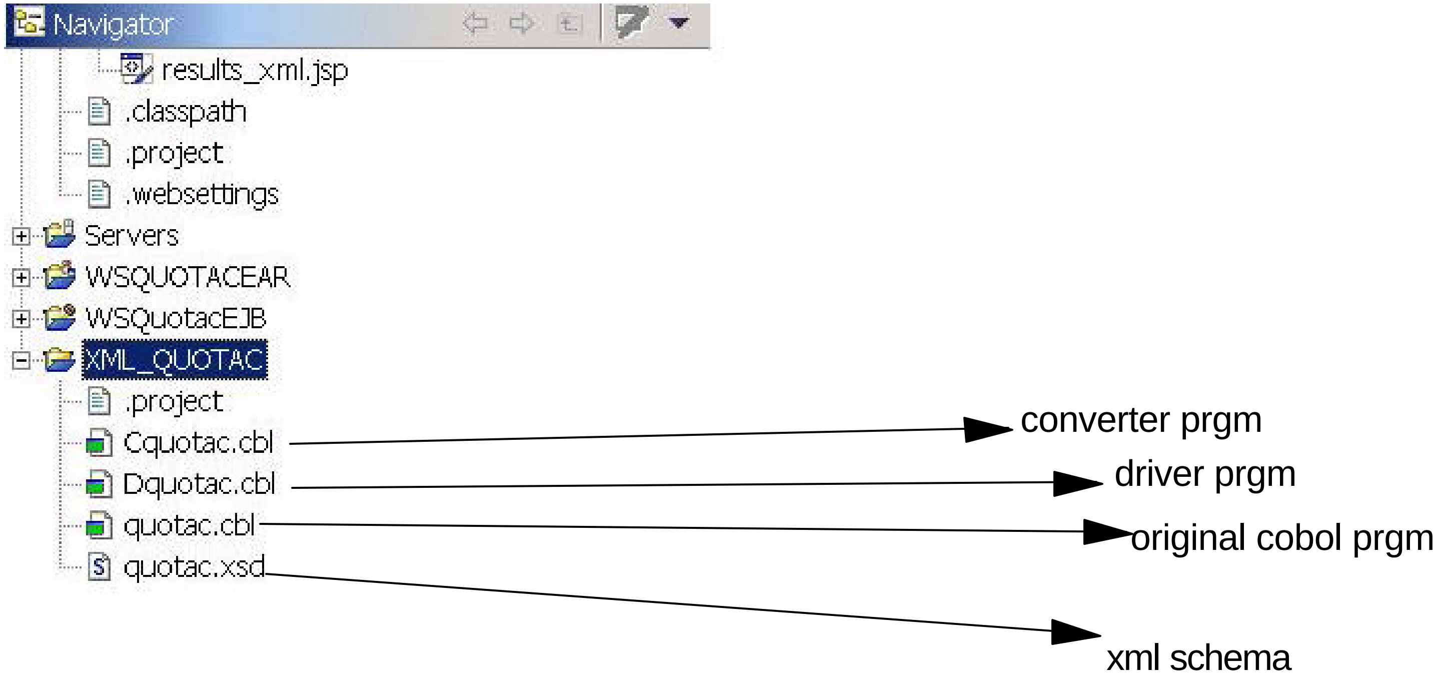
	C	!	Description	Resource	In

WebSphere software

IBM

WSED XML Enablement Wizard - Example output

- Generated files



Navigator

```
      results_xml.jsp
    .classpath
    .project
    .websettings
⊞ Servers
⊞ WSQUOTACEAR
⊞ WSQuotacEJB
⊟ XML_QUOTAC
    .project
    Cquotac.cbl
    Dquotac.cbl
    quotac.cbl
    quotac.xsd
```

converter prgm

driver prgm

original cobol prgm

xml schema

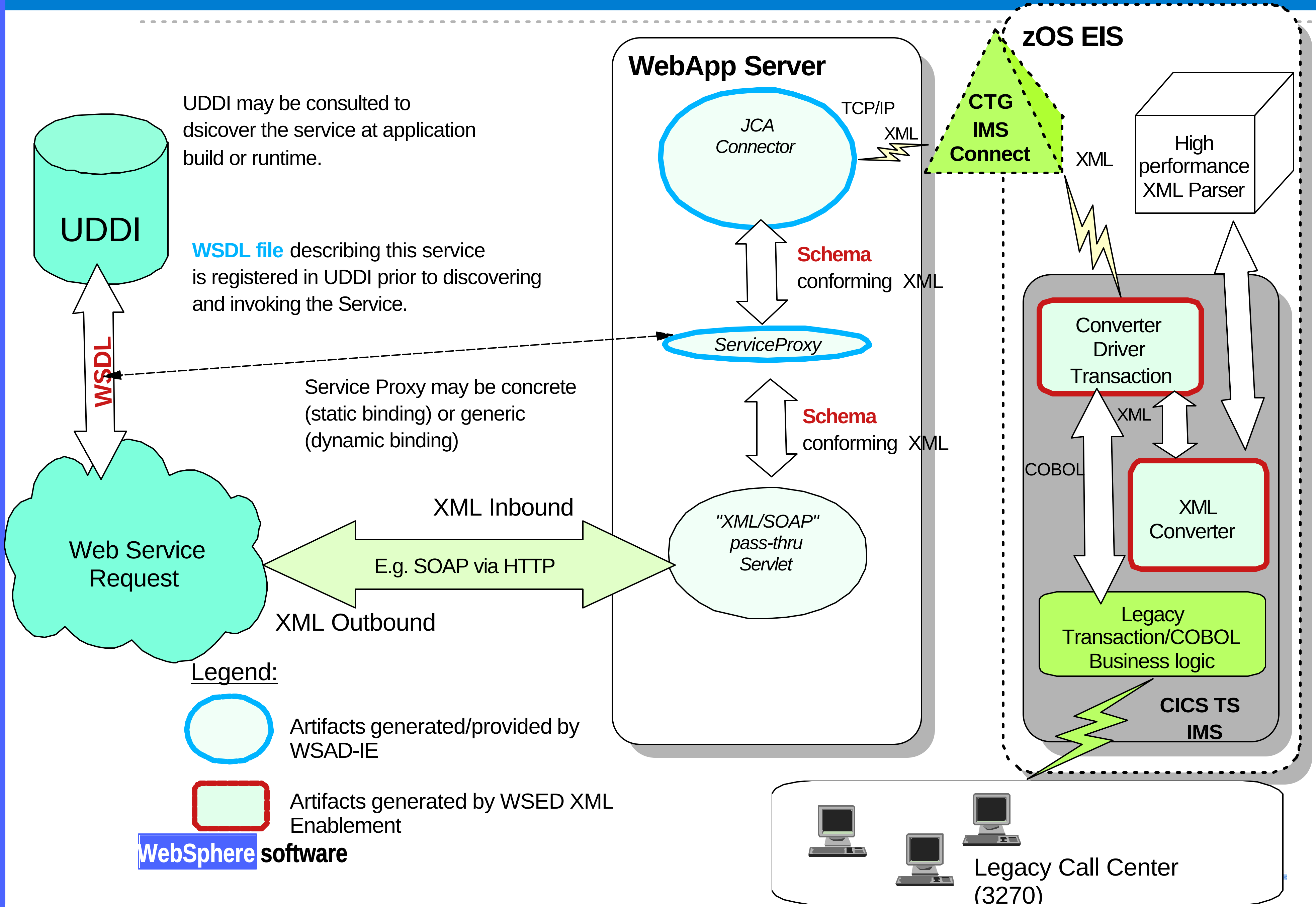
Creating a z/OS Web Service

- Legacy Program is now enabled to accept XML documents

- Can use Web services wizards within WebSphere Studio
  - From EJB or Javabean used for wrapper invocation of legacy code
  - From Javabean using generated XSD schema from XML Enablement tool

  - At General Availability, WSAD-IE z/OS Web Service wizards
    - Creates artifacts based on generated COBOL driver program from XML Enablement tool (or other COBOL programs)
      - WSDL
      - Service Proxy
      - JCA Connector code

XML Enablement Runtime Scenario and Web Services



UDDI

UDDI may be consulted to dsicover the service at application build or runtime.

WSDL

WSDL file describing this service is registered in UDDI prior to discovering and invoking the Service.

Web Service Request

Service Proxy may be concrete (static binding) or generic (dynamic binding)

XML Inbound

E.g. SOAP via HTTP

XML Outbound

WebApp Server

JCA Connector

TCP/IP

XML

Schema conforming XML

ServiceProxy

Schema conforming XML

"XML/SOAP" pass-thru Servlet

CTG IMS Connect

zOS EIS

XML

High performance XML Parser

Converter Driver Transaction

XML

COBOL

XML Converter

Legacy Transaction/COBOL Business logic

CICS TS IMS

Legacy Call Center (3270)

Legend:

○ Artifacts generated/provided by WSAD-IE

▢ Artifacts generated by WSED XML Enablement

XML Enablement Runtime Prerequisites

- Prerequisites
  - ►IBM Enterprise COBOL for z/OS and OS/390 Version 3 Release 1 (program number 5648-A25) or later
  - ►IBM Language Environment for OS/390 Version 2 Release 10 (program number 5647-A01) or z/OS V1.1 or later with PTF for APAR PQ65085 (Available Sept. 2002)
  - ►OS/390 R8/R9/R10 and z/OS V1R1 support for Unicode ™ is required for the XML converters generated by WSED General Availability release.OS/390 R8/R9/R10 and z/OS V1R1 support for Unicode ™ can be obtained free of charge at https://www6.software.ibm.com/dl/os390/unicodespt-p . This support is integrated into z/OS Version 1 Release 2 and later.
  - ►IBM WebSphere Studio Enterprise Developer (WSED) for Multiplatforms Version 5 Early Availability (EA) (program number 5724-B67) or later

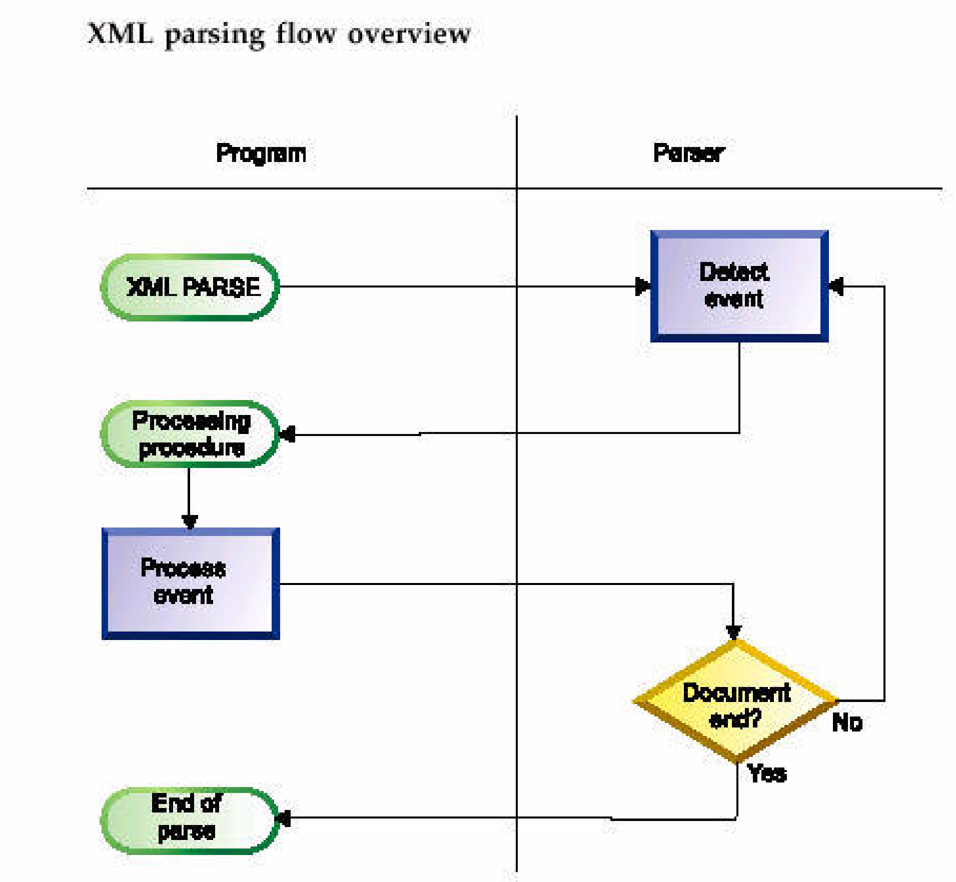
Summary

- Facilitate enterprise modernization by refacing existing COBOL applications to support XML messages
- Achieve significant productivity gains by utilizing the converter and driver template generators
- Gain performance benefits by running XML parsing and conversions on the z/OS systems
- Enable development of Web Services for z/OS applications

XML Enablement for z/OS

Backup Charts

XML Parsing Flow


XML parsing flow overview

COBOL Compiler Support for XML

- Introduced with IBM Enterprise COBOL for z/OS and OS/390 V3R1
- High-speed XML parser
  - Consumes inbound XML messages
  - Verifies well-formedness
  - Transforms contents into COBOL data structures
  - Supports XML documents encoded in Unicode UTF-16, EBCDIC, ASCII
- New **XML PARSE** statement
  - Begins XML parse
  - Identifies document to be processed
  - Identifies processing procedure
- Processing procedure
  - Controls the parse
  - Receives and processes XML events
  - Handles exceptions

XML Document Defined in Working-Storage

```
*****************************************************************
*XML document,encoded as initial values of data-items.*
*****************************************************************
1 xml-document.
   2 pic x(39)value '<?xml version=“1.0 ”encoding=“ibm-1140 ”'.
   2 pic x(19)value 'standalone=“yes ”?>'.
   2 pic x(39)value '<!--This document is just an example-->'.
   2 pic x(10)value '<sandwich>'.
   2 pic x(35)value '<bread type=“baker&apos;s best ”/>'.
   2 pic x(41)value '<?spread please use real mayonnaise ?>'.
   2 pic x(31)value '<meat>Ham &amp;turkey</meat>'.
   2 pic x(40)value '<filling>Cheese,lettuce,tomato,etc.'.
   2 pic x(10)value '</filling>'.
   2 pic x(35)value '<![CDATA [We should add a <relish>'.
   2 pic x(22)value 'element in future!]]>'.
   2 pic x(31)value '<listprice>$4.99 </listprice>'.
   2 pic x(27)value '<discount>0.10</discount>'.
   2 pic x(11)value '</sandwich>'.
1 xml-document-length computational pic 999.
```

Data Definitions for XML Content

```
*******************************************************************
*Sample data definitions for processing numeric XML content.*
*******************************************************************
1 current-element pic x(30).
1 xfr-ed pic x(9)justified.
1 xfr-ed-1 redefines xfr-ed pic 999999.99.
1 list-price computational pic 9v99 value 0.
1 discount computational pic 9v99 value 0.
1 display-price pic $$9.99.
```

Parsing XML Documents

```
XML PARSE xml-document
      PROCESSING PROCEDURE
xmlevent-handler
   ON EXCEPTION
      DISPLAY 'XML document error'
XML-ERROR
      STOP RUN
   NOT ON EXCEPTION
      DISPLAY 'XML document was
succesfully parsed.'
END-XML
```

XML Processing Procedure

```
xmlevent-handler section.
   evaluate XML-EVENT
*==>Order XML events most frequent first
      when 'START-OF-ELEMENT'
         display 'Start elementtag:<'XML-TEXT '>'
         move XML-TEXT to current-element
      when 'CONTENT-CHARACTERS'
         display 'Content characters:<'XML-TEXT '>'
*==>Transform XML content to operational COBOL data item...
         evaluate current-element
            when 'listprice'
*==>Using function NUMVAL-C...
            compute list-price =function numval-c(XML-TEXT)
            when 'discount'
*==>Using de-editing of a numeric edited item...
            move XML-TEXT to xfr-ed
            move xfr-ed-1 to discount
         end-evaluate
      when 'END-OF-ELEMENT'
      ....
```