Session B05
An Introduction to XML

for Database Specialists

S. Malaika, db2xml@us.ibm.com

IBM Data Management Technical Conference, Anaheim, CA **September, 2002**

Agenda

- XML Motivation
- XML Overview
  - The basics: XML 1.0 and 1.1, DTDs, XML Schemas, Namespaces
  - The basic XML technologies: XML Parsers and APIs: DOM, SAX
- XML Standards
- XML Technologies
- More on XML
  - Transformation and Query:
    - XPath, XSLT, SQL/XML, XQuery
- XML Resources
- XML Summary

Why XML

- XML: a notation for data exchange between systems and applications that have not been introduced to each other.
- XML: enables the creation of precise descriptions for admissible content:
  - Encoding: Allowable characters and allowable character encodings - to support diverse platforms (operating environments and hardware).
    - Ways to discover the particular encoding
  - Schemas: Methods for defining application specific and general purpose content
    - Ways to discover, access and process the schemas
- The success of XML has resulted in the wide availability of :
  - General purpose software for processing XML, e.g., XML parsers, XML transformers
  - XML content - and hence the need for data managers to support XML

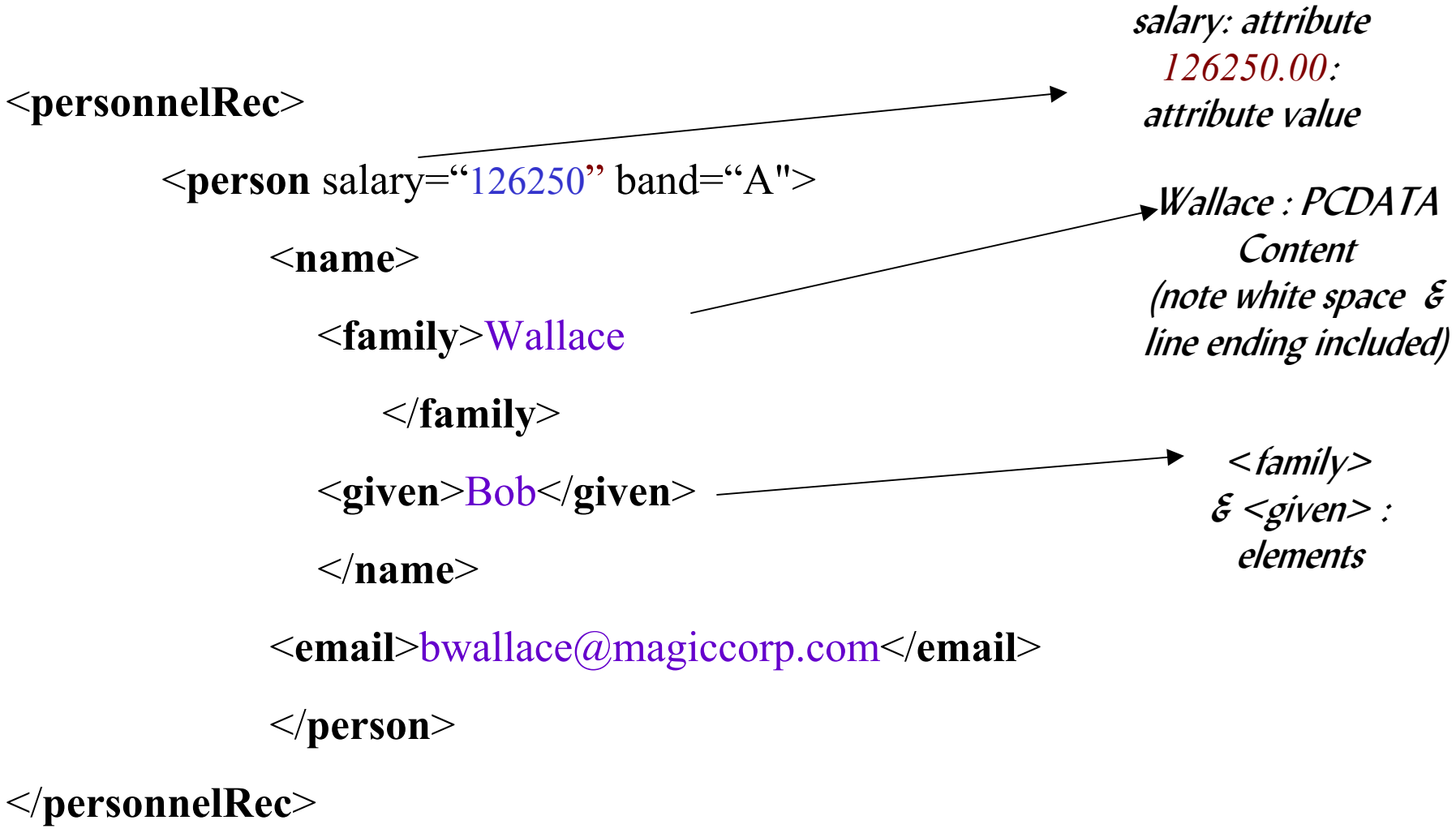
XML

- **XML:** e**X**tensible **M**arkup **L**anguage for specifying your own tagset (elements) in documents
  - Begin tag example: <tag>
  - End tag example: </tag>
  - XML documents must be **well formed**
    - Exactly one root element
    - All begin and end tags are present and properly nested (or /> used to skip the end tag for empty elements).
    - All attribute values are in quotes.
- **XML** is a simplified subset of **SGML**
  - Standard Generalized Markup Language
- The XML 1.0 Specification is at:
    - **http://www.w3.org/TR/REC-xml**

Valid XML

- With DTDs you can:
  - Specify the shapes of documents, e.g., element nesting and repetition
  - Set default values, or range checks
  - Substitute entities
- With XML Schemas you can:
  - Specify data types for element and attribute values
  - Define your own complex types
- A document that conforms to an XML schema or to a DTD is called a valid document
- All XML documents are well formed but only some are valid

Elements, Attributes and Character Data



Well Formed XML

1. Violates “One Root Element”

```
<family>Wallace</family>
<given>Bob</given>
```

2. Violates “Every Start Tag has a matching End Tag”

```
<name><family>Wallace</family>
    <given></name>
```

3. Violates “Tags are properly nested”

```
<name>
    <family>Wallace<given>Bob
    </family></given>
</name>
```

4. Well Formed XML

```
<name>
    <family>Wallace</family>
    <given>Bob</given>
</name>
```

XML Vocabularies

- An **XML grammar** (or **vocabulary**) is defined by:
  - An XML Document Type Definition (DTD) or an XML Schema.
- Some analogies:
  - DTDs and XML Schemas are analogous to DDL for relational tables
  - XML instance documents are analogous to rows in tables
- With DTDs or XML Schemas you can:
  - **validate** XML documents that you consume or produce or modify using a validating XML parser
  - **define your own vocabularies** to exchange documents within your company or between companies

Sample XML instance document:
personnel data

```
<?xml version="1.1" encoding="UTF-8" ?>          <!--XML declaration -->
<!DOCTYPE personnelRec SYSTEM "prml.dtd"> <!-- Doctype declaration -->
<!-- This is a comment -->
<personnelRec>                                   <!-- Root element-->
        <person salary="250000.00" band="D">
            <name><family>Terfel</family><given>Bryn</given></name>
            <email>terfel@roh.org</email>
            <dept>&d1</dept>                     <!-- entity reference -- >
        </person>
</personnelRec>
```

Processing XML with an XML parser

- XML processors such as parsers respond to XML processing instructions
  - Processing instructions are bounded by <? and ?>
  - A special processing instruction is the xml declaration at the start of an XML document
    - **XML version, e.g., 1.0 or 1.1 (in development at the w3c)**
    - **XML encoding, if absent: UTF-8 (Unicode) is assumed**
  - XML markup declarations are bounded by
    - <! and >
    - An example is the DOCTYPE declaration
  - XML comments are bounded by <!-- and -->
- Typically XML parsers work with XML in UTF-16
  - Documents will be converted prior to parser processing

Document Type Definition for Personnel Data

```
<?xml encoding="UTF-8"?>
<!ENTITY d1 “bass department">
<!ELEMENT personnelRec (person)+>
<!ELEMENT person (name, email*)>
    <!ATTLIST person salary CDATA #REQUIRED >
    <!ATTLIST person band (A|B|C|D|E|F) #REQUIRED>
    <!ATTLIST person active (true|false) "true" #IMPLIED >
<!ELEMENT name (family, given)>
<!ELEMENT family (#PCDATA)>
<!ELEMENT given (#PCDATA)>
<!ELEMENT email (#PCDATA)>
```

PRML.DTD

XML Schema

- In contrast with DTDs:
  - XML schemas are XML documents
  - There is no linkage mechanism from XML instance documents to XML schemas
- XML schemas can be used with DTDs
- You can enforce data type checking with XML schemas
- Constructs in schemas include:
  - Complex types
- There is a mapping defined between XML schema types and SQL data types (part of SQL/XML)

XML Schema for Personnel Data
(Part 1)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">
<xs:element name="email" type="xs:string"/>
<xs:element name="family" type="xs:string"/>
<xs:element name="given" type="xs:string"/>
<xs:complexType name="nameType">
        <xs:sequence>
                <xs:element ref="family"/><xs:element ref="given"/>
        </xs:sequence>
</xs:complexType>
```

XML Schema for Personnel Data
(Part 2)

```
<xs:complexType name="personType">
<xs:sequence>
        <xs:element name="name" type="nameType"/>
        <xs:element ref="email" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="salary" type="xs:string" use="required"/>
```

XML Schema for Personnel Data (Part3)

```
<xs:attribute name="band" use="required">
        <xs:simpleType><xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="A"/>
        <xs:enumeration value="B"/>
        <xs:enumeration value="C"/>
        <xs:enumeration value="D"/>
        <xs:enumeration value="E"/>
        <xs:enumeration value="F"/>
        </xs:restriction></xs:simpleType>
</xs:attribute>
```

XML Schema for Personnel Data
(Part 4)

```
<xs:attribute name="active" default="true">
        <xs:simpleType><xs:restriction base="xs:NMTOKEN">
        <xs:enumeration value="true"/><xs:enumeration value="false"/>
        </xs:restriction></xs:simpleType>
</xs:attribute>
</xs:complexType>          <!-- personType -->
<xs:element name="personnelRec">
        <xs:complexType><xs:sequence maxOccurs="unbounded">
                <xs:element name="person" type="personType"/>
                </xs:sequence></xs:complexType>
</xs:element></xs:schema>
```

Namespaces

- Naming conflicts occur in XML, e.g., when elements from different vocabularies are included in a single document
- Example: <name> in the fragment below:

```
<person>
        <name>Pavarotti</name>
        <dept><name>La Scala</name></dept>
</person>
```

- Namespaces are a two-part naming system: URI + local name
  - The URI qualifies element names
- The URI does not have to point to anything in particular

Namespaces

- The xmlns attribute associates element prefixes with URIs

```
<person xmlns:opsingers="http://www.operapeople.org/
     xmlns:oplocations="http://www.operaplaces.org/">
    <opsingers:name>Pavarotti</opsingers:name>
    <oplocations:name>La Scala</oplocations:name>
</person>
```

- Can define a default URI

```
<person xmlns ="http://www.operapeople.org/">
    <name>Pavarotti</name>
</person>
```

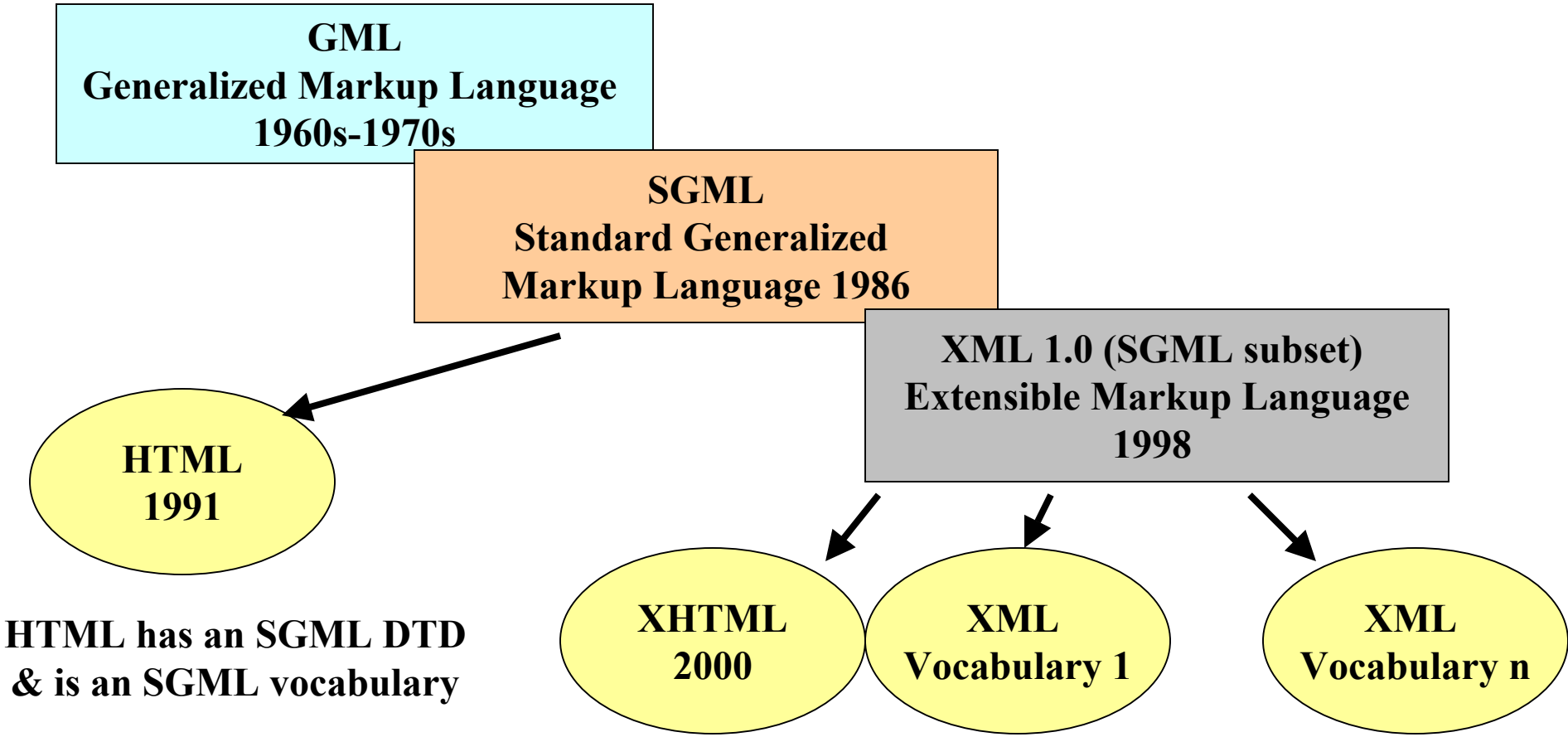
Processing XML with XML Parsers

- XML parsers support the DOM and SAX APIs
  - **DOM**: **Document Object Model**
    - The whole document is materialized in memory
    - Is flexible (Applications can navigate the XML tree many times)
  - **SAX: Simple API for XML**
    - Applications can navigate the XML document once only
- IBM's XML parsers and stylesheet processors for a variety of UNIX and Windows platforms can be downloaded from AlphaWorks
    - **http://www.alphaworks.ibm.com/tech/**
    - **http://www.ibm.com/developer/xml/**
  - IBM's XML parsers and stylesheet processors:
    - XML4C XML4J, Xalan-C and Xalan-J
- IBM's XML parser for OS390 (zSeries) can be downloaded or ordered from
    - **http://www.ibm.com/servers/eserver/zSeries/software/xml/**
- Source for XML parsers and stylesheet processors etc can be viewed at
    - **http://www.apache.org/**

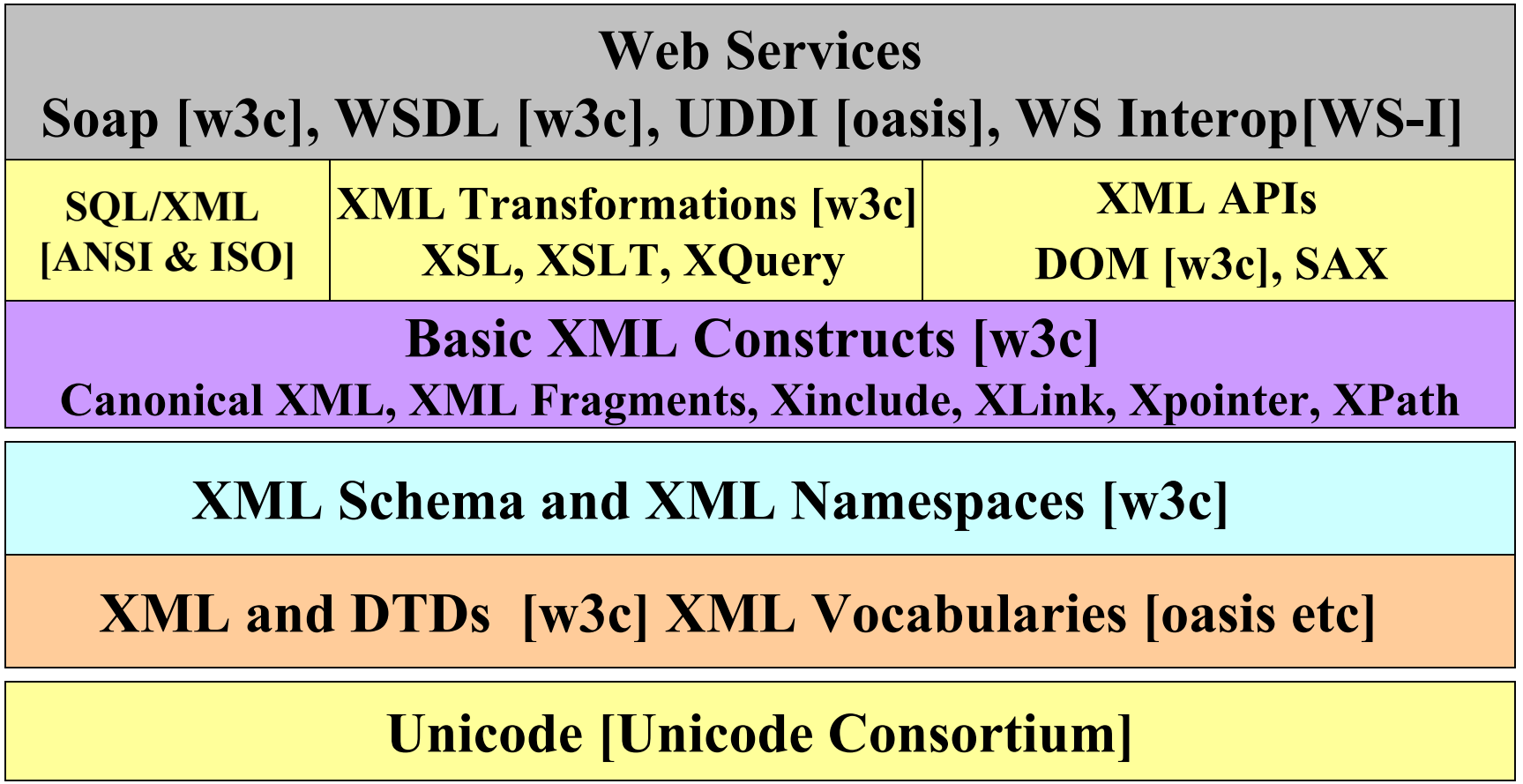
XML Design Considerations

- Use Unicode for XML wherever possible
- Elements or Attributes?
  - If in doubt: use elements.
- Avoid deeply nested structures
  - Take care with automated converters, e.g., from ASN.1 to XML
- Avoid designing large documents
  - Not all software can handle
  - Takes longer to parse
- Avoid duplicate element names unless you are using namespaces
- Avoid XML designs which model all content as name-value pairs
  - Analogous to designing databases with one table structure

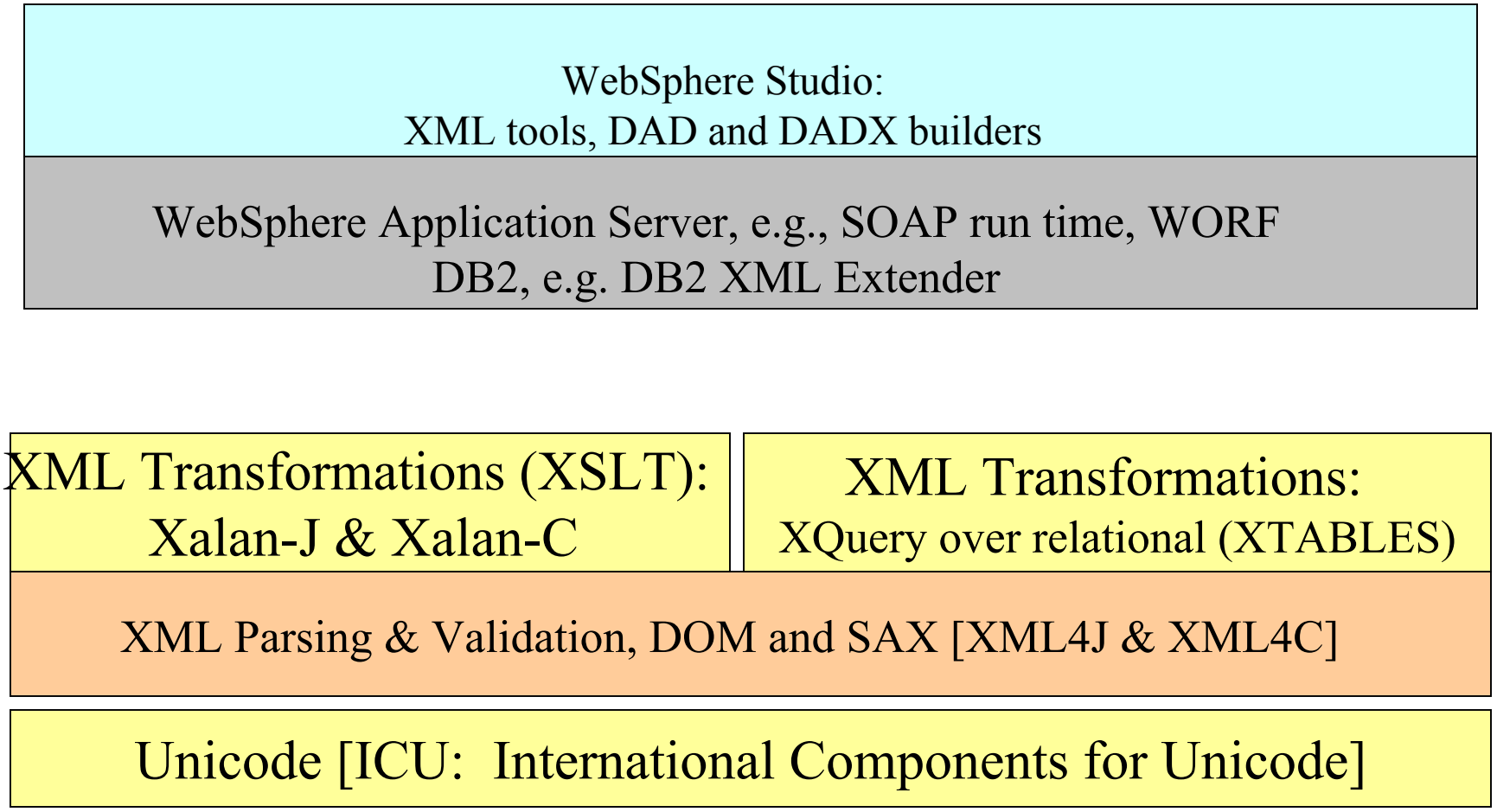
XML Time Line



Some XML Standards



Some XML Technologies from IBM



XML Prerequisites for Understanding DB2 XML and DB2 Web Services Support

- XML
  - Elements, Attributes, text nodes (PCDATA content)
- DTD and XML schemas
- Namespaces
- XPath
- XSLT

XPath

- Used to locate XML elements and attributes in a document
- Examples
  - /PersonnelRec/Person/Name/Family
  - /PersonnelRec/Person/@Salary
- -W3C Recommendation
- Used in XSLT, XQuery and DB2 XML Extender

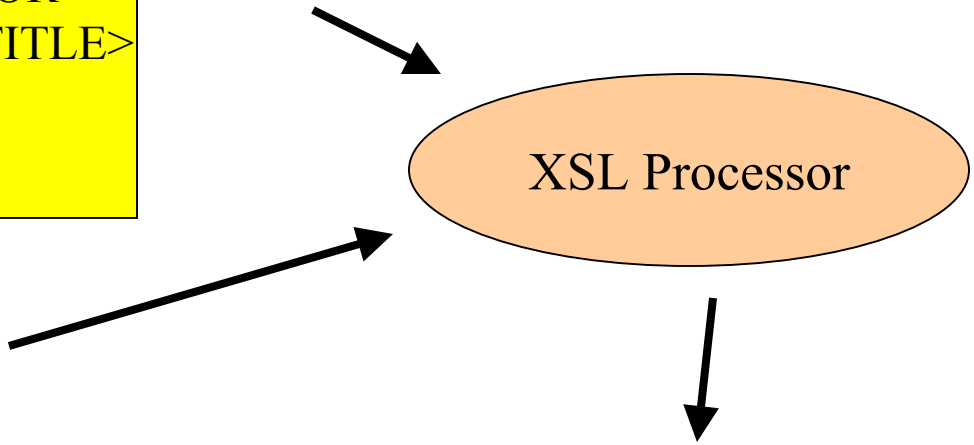
XSLT Example

eXtensible Style Sheet Language Transformation

Supported in DB2 V8.1 through the DB2 XML Extender

```
<BOOK >
        <AUTHOR>Neil Bradley</AUTHOR>
        <TITLE>The XML Companion</TITLE>
        <PRICE>$30.00</PRICE>
 </BOOK>
```

XML Input




```
<xsl:stylesheet>
<xsl:template match="TITLE">
<H1>
<xsl:apply-templates/>
</H1>
</xsl:template>
<xsl:template match="AUTHOR"/>
<xsl:template match="PRICE"/>
</xsl:stylesheet>
```

Stylesheet Input

```
<H1>
The XML Companion
</H1>
```

HTML Output

SQL/XML Support

Set of SQL extensions known as SQL/XML (or SQLX)

These functions are available in DB2 V8.1 (documented in the SQL Referemce):

- **XMLATTRIBUTES: Creates an XML Attribute**
- **XMLELEMENT: Creates an XML Element**
- **XMLAGG: Produces a forest of elements from a collection of elements.**
- **XML2CLOB: Return result as CLOB**

Example:

- **SELECT e.id,XML2CLOB(XMLELEMENT(NAME "emp",e.fname ||''||e.lname)) AS "result" FROM employee ;**

Result:

1001, <emp> John Smith</emp>

1206 , <emp>James Martin</emp>

SQL/XML Support

Example:

```
SELECT XMLELEMENT ( NAME "Department",
XMLATTRIBUTES ( e.dept AS "name" ),
XMLAGG( XMLELEMENT( NAME "emp", e.lname ))
) AS "dept_list"
FROM employees e GROUP BY dept
```

Result:

```
<Department name="Accounting">
    <emp>Yates</emp><emp>Smith</emp>
</Department>
<Department name="Shipping">
    <emp>Oppenheimer</emp><emp>Martin</emp>
</Department>
```

XQuery

A Query Language for XML

- **Input: XML documents**
- **Output: A sequence of XML related items**

Currently being defined at the W3C

- **FLWR Format: for, let, where, return**

DB2 XTABLES prototype

- **XQuery support over an XML view of relational data**

Example (from XQuery Use Cases):

For each item that has received a bid, list the item number, the highest bid, and the name of the highest bidder, ordered by item number.

XQuery

Example:

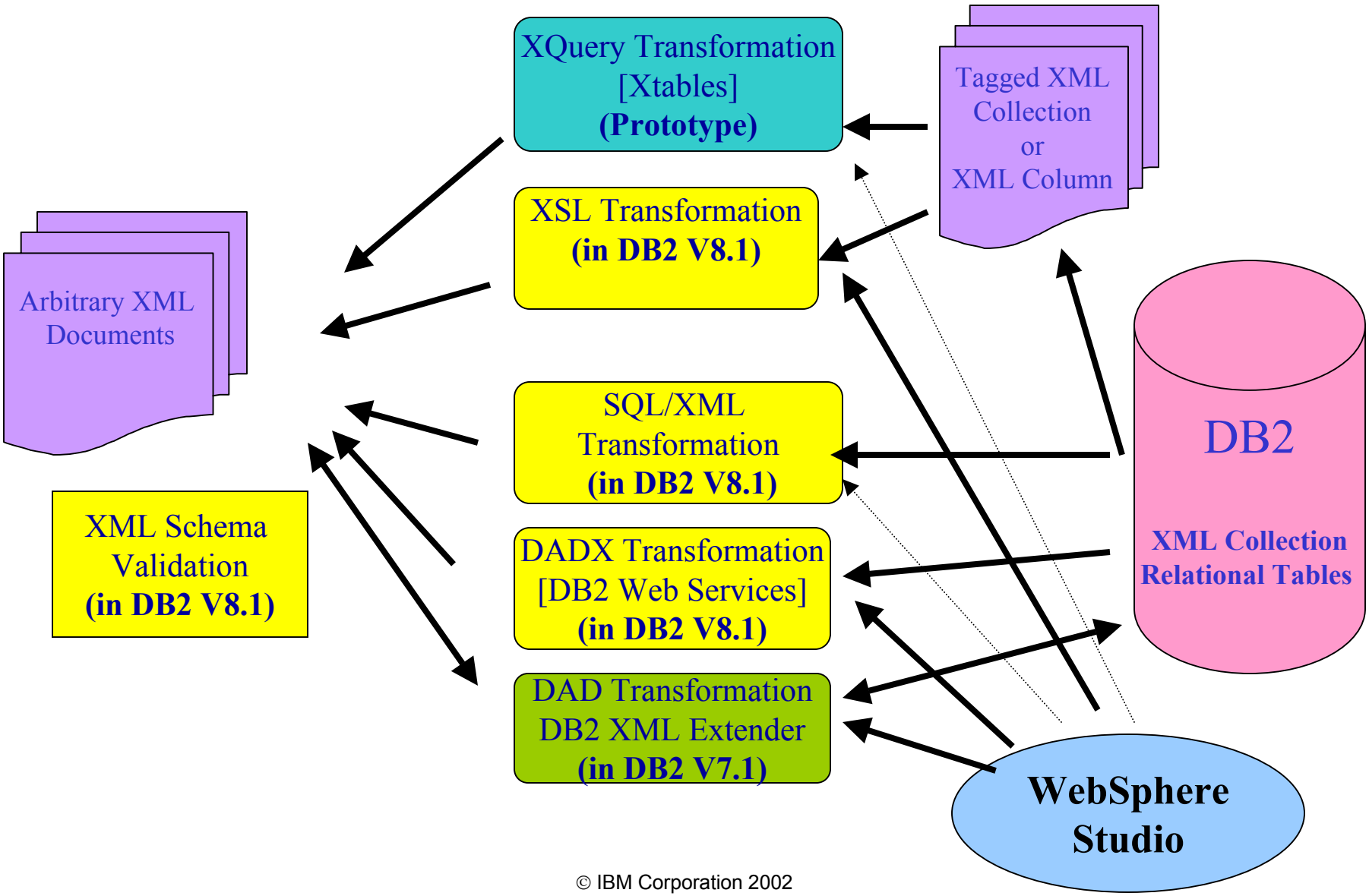
```
<result> { for $highbid in document("bids.xml")//bid_tuple, $user
    in  document("users.xml")//user_tuple
where $user/userid = $highbid/userid and
   $highbid/bid =
      max(for $x in
          document("bids.xml")//bid_tuple[itemno=$highbid/itemno]/bid
       return decimal($x))
return
<high_bid> { $highbid/itemno } { $highbid/bid } <bidder>{
    $user/name/text() }</bidder> </high_bid>
 sortby(itemno) } </result>
```

XQuery

Result:

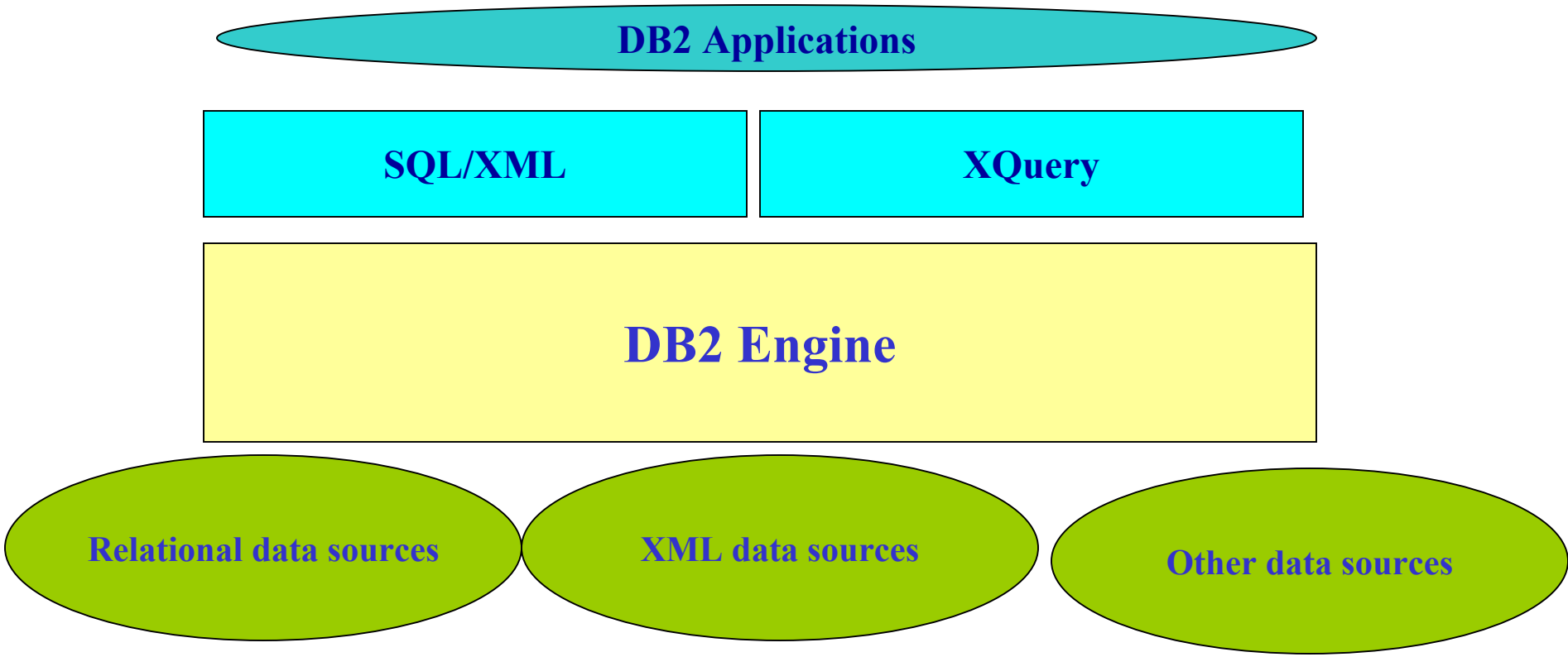
```
<result>
    <high_bid>
        <itemno>1001</itemno>
        <bid>55</bid>
        <bidder>Mary Doe</bidder>
    </high_bid>
    <high_bid>
        <itemno>1002</itemno>
        <bid>1200</bid>
        <bidder>Mary Doe</bidder>
    </high_bid>
</result>
```

XML Transformations in DB2 through SQL



SQL and XML Directions Summary

- **XML data type and XML Extender further integrated within DB2**
- **SQL/XML: Relational programming interfaces adapted for XMLand in the process of standardization (first available in DB2 V8.1)**
- **XQuery: XML programming interfaces (in the process of standardization)**



XML Resources: General

- Robin Cover pages for XML and SGML
  - http://www.oasis-open.org/cover/sgml-xml.html
- Standards Groups and Consortiums:
  - Unicode http://www.unicode.org/
  - The W3C http://www.w3.org/
  - Oasis http://www.oasis-open.org/
  - xml.org: http://www.xml.org
  - Web Services Interoperability: http://www.ws-i.org/
- The Annotated XML Specification
  - http://www.xml.com/pub/a/axml/axmlintro.html
- xml-dev mailing list and archives
  - http://www.xml.org/xml/xmldev.shtml

XML Resources: From IBM

- International Components for Unicode (ICU)
  - http://oss.software.ibm.com/icu/
- IBM XML parsers XML4C, XML4J, Xalan-C and Xalan-J (LotusXSL)
  - http://www.alphaworks.ibm.com/
- IBM XML parsers & ICU for OS390 (zSeries)
  - http://www.ibm.com/servers/eserver/zSeries/software/xml/
- IBM Developerworks for XML
  - http://www.ibm.com/developerworks/xml/
- IBM Developerworks for Web Services
  - http://www.ibm.com/developerworks/webservices/
- WebSphere Studio: http://www.ibm.com/software/ad/studioappdev/
- DB2 XML Extender
  - http://www.ibm.com/software/data/db2/extenders/xmlext/
- DB2 XML Extender Hints and Tips
  - http://www.ibm.com/software/data/db2/extenders/xmlext/support.htm
- DB2 Web Services http://www7b.boulder.ibm.com/dmdd/zones/webservices/
- Xperanto
  - http://www.ibm.com/software/data/developer/demos/xperanto/

XML Resources: IBM Papers

- Red books and red papers
  - http://www.redbooks.ibm.com/
    - Integrating XML with DB2 XML Extender and DB2 Text Extender SG24-6130
    - DB2 for OS/390 and z/OS Powering the World's e-business Solutions SG24-6257 (Chapter on XML Extender)
    - DB2 XML Extender Hints and Tips (red paper)
      - http://www.redbooks.ibm.com/redpapers/pdfs/redp0135.pdf.
- DB2 MQSeries and XML Papers
  - http://www7b.boulder.ibm.com/dmdd/library/techarticle/wolfson/0108wolfson.html
  - http://www7b.boulder.ibm.com/dmdd/library/techarticle/wolfson/0201wolfson.html
  - http://www.ibm.com/software/data/db2/extenders/xmlext/docs/v72wrk/dxxmq.htm
  - http://www.ibm.com/software/data/db2/extenders/xmlext/docs/v72wrk/dxxrnfp4.htm#Header_10

XML Resources: IBM Papers and Downloads

Download DB2 Web Services V7.2

http://www7b.software.ibm.com/dmdd/zones/webservices/worf/

(also available in DB2 V8.1 (Windows & UNIX) and in WebSphere Studio)

- DB2 and XML Web Services Papers

  http://www7b.software.ibm.com/dmdd/zones/webservices/

  – DB2 and Web Services: The Big Picture
    http://www7b.boulder.ibm.com/dmdd/zones/webservices/bigpicture.html
  – Running DB2® Web Services on WebSphere® Application Server Advanced Edition 4.0 by Reto Preisig
  – http://www7b.boulder.ibm.com/dmdd/library/techarticle/preisig/0108preisig.html

DAD Checker

http://www.ibm.com/software/data/db2/extenders/xmlext/download/beta/dadcheck_rn.html

(available in DB2 V8.1)

XML Summary

- There are many XML based interfaces and technologies available or in development including:
  - Unicode: Used for encoding XML documents
  - Vocabulary definition & validation: XML, DTDs, XML Schemas, Namespaces
  - XML document APIs: DOM, SAX
  - Transformation and Query: XPath, XML Stylesheets, XQuery\
  - SQL/XML: XML extensions for the SQL language
  - Document fragment management and composition: Xinclude, Xlink, Xpointer
  - Web Services: SOAP, WSDL, UDDI
- There are many industry consortia and standards bodies defining standards based on XML and Web Services interfaces:
  - W3C, Oasis, WS-I, Java JSRs, ANSI etc

XML and DB2 Summary

- The following are available in DB2:
  - There are multiple options:
    - for storing XML in DB2: XML Collection (relational tables), XML Column (a single column), Clobs
    - for manipulating XML in DB2: extract, update, import, export UDFs, SQL extensions (SQL/XML) for tagging, document shredding and composition stored procedures
  - There is support in DB2
    - for managing XML document encodings in conjunction with DB2 database code pages which may be different
    - for integrating DB2 data with XML in file systems and XML in MQSeries queues
    - for DB2 as a Web Service Provider (through DADX)
- WebSphere Studio provides support for building DB2 XML (DAD) and Web Services (DADX) applications

XML and DB2 Directions Summary

- DB2 directions are:
  - Increased integration
    - of XML storage and indexing methods in DB2
    - of XML APIs and technologies in DB2
  - Increased support
    - of XML APIs for use in DB2 applications
    - of Web Services for use in DB2 applications or to access DB2 data or other Web Services
  - More tools to develop and maintain DB2 XML and Web Services:
    - applications
    - meta data