



XML in DB2 9 for z/OS

August 2007

DB2 track session 4

Information Management software



What is XML?

XML

- ▶ eXtensible Markup Language
- ▶ Self-describing data structures
- ▶ XML tags describe each element and their attributes

Benefits

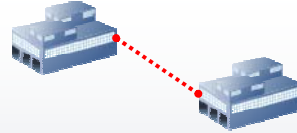
- ▶ Flexible
 - No fixed format or syntax
 - Structures can be easily changed
- ▶ Platform Independent
 - Not tied to any platform, operating system, language or software vendor
 - XML can be easily exchanged
- ▶ Fully Unicode compliant

```
<? xml version="1.0" ?>  
<purchaseOrder id="12345" >  
  <customer id="A6789">  
    <name>John Smith Co</name>  
    <address>  
      <street>1234 W. Main</street>  
      <city>Toledo</city>  
      <state>OH</state>  
      <zip>95141</zip>  
    </address>  
  </customer>  
  ...
```

XML Solves Business Problems Today

- **Business to Business Integration**

- ▶ Platform independent transport mechanism
Purchase order triggers transactions flowing over a service oriented architecture



- **Forms and Document Processing**

- ▶ Government and legal industry require digital signature
Tax forms require signature & change year to year



- ▶ Documents often contain sub-documents
Literary materials contain books, chapters, and sub-chapters

- **Business Insight**

- ▶ Universal representation from multiple sources
Claims adjustor reviews damage estimates from multiple garages with consideration of original format



3

XML is the Language of Business

- **Banking and Financial Markets**

- ▶ **IFX** - Interactive Financial Exchange – Trades, banking, consumer transactions, etc.
- ▶ **MISMO** - Mortgages

- **Insurance**

- ▶ **ACORD** – Policy management, underwriting, indemnity, claims, etc.
<http://www.acord.org>

- **Healthcare**

- ▶ **HL7** – Patient Management – Diagnosis, treatments, prescriptions, etc. <http://www.hl7.org>

- **Retail**





- ▶ **IXRetail** – Inventory, customer transaction and employee management <http://www.nrf-arts.org/>

... and hundreds more!

4

XML Data Needs Relational Maturity

Complementing XML Processing

- **XML Data Needs Protection**
 - ▶ Backup and recovery features to ensure continuity
 - ▶ Data is protected using database security
- **Simplified XML Data Access**
 - ▶ Centrally store and access difficult to retrieve data
 - ▶ SQL or XQuery can be used to retrieve data
 - ▶ Join XML data with it's related relational data
- **Search Speed**
 - ▶ Search documents quickly and efficiently using proven search optimization engine of mature database
- **Optimize Existing Investments**
 - ▶ Use existing technology infrastructure and skills to store and manage both relational and XML

5

Yesterday's Solutions

- **Not Persisting XML**
 - ▶ No cost effective solution
- **XML in File Systems**
 - ▶ No elegant solution for backup, recover, search, retrieval
- **XML Proprietary Databases**
 - ▶ Difficult to integrate with traditional relational data
- **Relational Model Accommodations**
 - ▶ Shredding
 - ▶ Large Objects

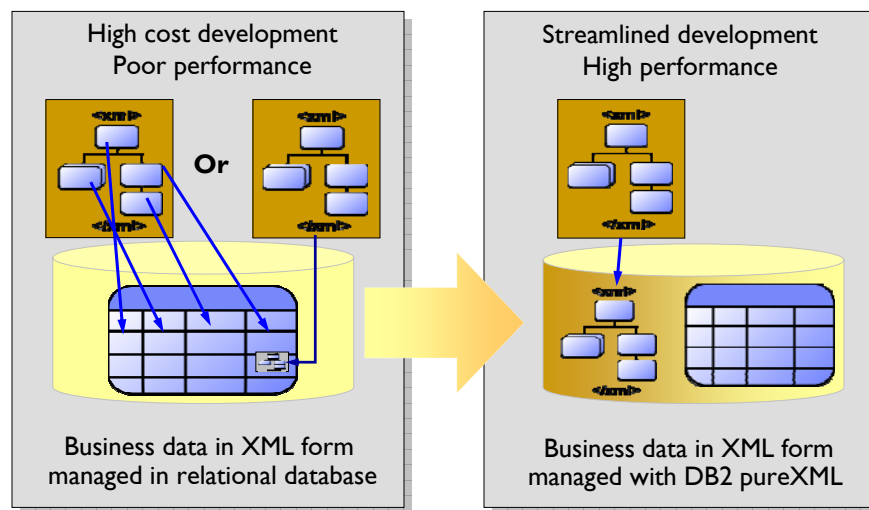
6

Problems of Relational Model XML

- CLOB storage:
 - Query evaluation & sub-document level access requires costly XML Parsing – too slow !
- Shredding:
 - Mapping from XML to relational often too complex
 - Often requires dozens or hundreds of tables
 - Complex multi-way joins to reconstruct documents
 - XML schema changes break the mapping
 - no schema flexibility !
 - For example: Change element from single- to multi-occurrence requires normalization of relational schema & data

7

DB2 9: A New Generation Hybrid Data Server



8

DB2 9 – A Pure XML, Relational Hybrid

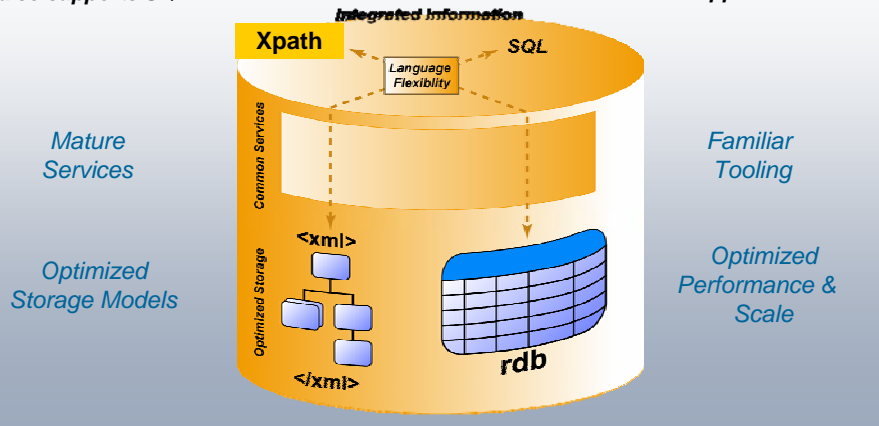
XML Developer
 "I see a sophisticated XML repository that also supports SQL."



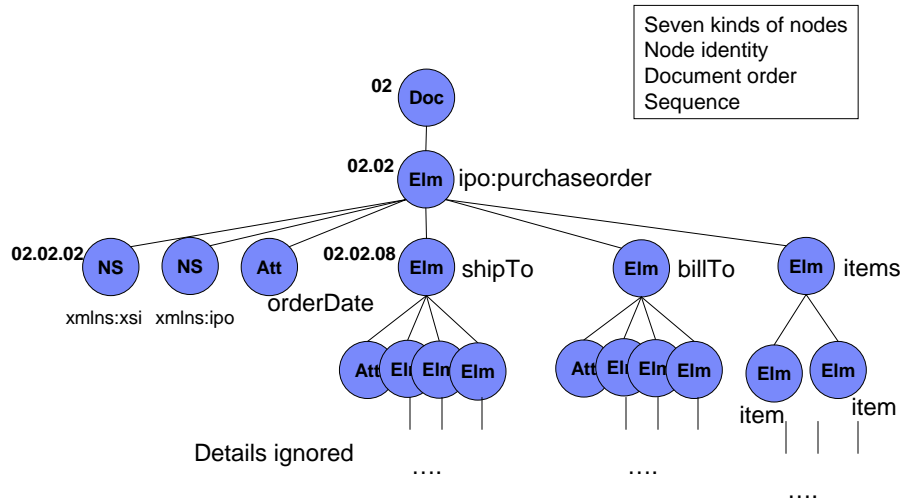
Familiar Programming Models



SQL Developer
 "I see a sophisticated RDBMS that also supports XML."

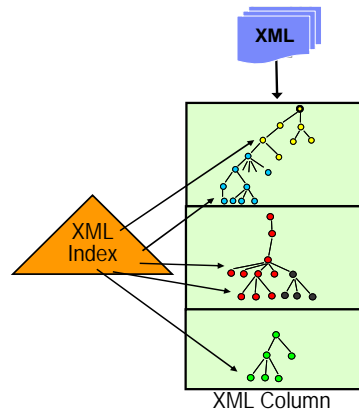


XML Data Model (XDM)



DB2 pureXML Advantages

DB2's hierarchical storage:
XML type as XML



- Directly store XML, no decomp/comp, normalize/de-normalize
- Eliminates database schema evolution bottleneck
- Declarative language, reduce complexity, dramatically improve application development productivity
- Native processing, high performance
- Unparalleled reliability, availability, scalability

Up to 10 times

11

Example: Tax Forms

- Application
 - ▶ Processing & validating tax returns, payments, refunds
 - ▶ Corporate Tax, Personal Income Tax (PIT), Sales Tax
- Objectives
 - ▶ Move Tax processing off legacy systems
 - Move to a more flexible, automated, extensible framework
 - Reduce cost & labor for implementing tax form changes
 - ▶ Increase performance. Improve straight-through processing from filing to refund/payment
- Typical current environment
 - ▶ Processing using manual and/or legacy systems
- This is an example of usage for Online Forms processing in general

12

Tax Forms

- Usually hundreds-thousands of different tax forms
 - **Schema Diversity**
 - Typically not every field in a form is used
 - **Sparse Data**
 - Many forms change every year
 - **Schema Evolution**
- **A case for XML !**

The image shows a detailed view of a tax form, specifically the 'Resident Income Tax Return' (IT-201) for the year 2000. The form is filled out with various data points, including personal information, income sources, and tax calculations. It features a grid-like structure for reporting multiple income items and a series of checkboxes for filing status and other tax-related options. The form is titled 'New York State Department of Taxation and Finance' and includes a barcode and the year '2000' in several places.

Typical Current Usage: Relational Database

- Solution 1: Each form has a different set of fields (schema)
 - Thousands of Tables ... i.e. one per form ?
- Considered not feasible
 - ▶ Too many tables to maintain
 - ▶ Relational schema would deteriorate over time
 - ▶ Not sufficiently flexible and extensible
- Solution 2: Single table whose rows can store *any* form
 - ▶ 100s of generic columns ... Ouch!

This block contains two overlapping images of tax forms. The top form is the 'Resident Income Tax Return' (IT-201) for 2000, which is partially obscured by the bottom form. The bottom form is the 'General Business Corporation Franchise Tax Return' (CT-3) for 2000. Both forms are complex, multi-sectioned documents with various fields for reporting financial and tax information.

Generic columns → XML

Current relational storage, inefficient, anonymous columns, requires complex mappings in the application

col1	col2	col3	col4	col5	...	col1000
134	NULL	11/23/05	NULL	NULL		NULL
NULL	276	NULL	NULL	Yes	...	NULL
12	NULL	NULL	99.99	NULL	...	NULL
NULL	NULL	NULL	...	NULL	...	NULL

New XML format:

```
<form>
  <wages>134</wages>
  <date>11/23/05</date>
</form>
```

XML: Avoids sparsity. Proper data labeling. 2 columns, not 1000. Transformable. Extensible. Simplifies mapping.

DB2 9 – Summary of pureXML Support

- XML as a native data type
- Pure XML storage and indexing
- SQL/XML and XPath support
- Integration with traditional relational data
- XML Schema Repository
- Schema validation
- Application Support (Java, C/C++, .NET, PHP, COBOL, PL/1 etc.)
- Visual Tooling, Control Center Enhancements
- DB2 Utilities: Load, Unload, Reorg, etc.
- ...and more

DB2 9

Secure and Resilient Infrastructure for a New Breed of Agile Applications

Interface Overview



- **Data Definition**
create table dept(deptID int, deptdoc xml);
- **Insert**
insert into dept(deptID, deptdoc) values (?,?)
- **Index**
create index xmlindex1 on dept(deptdoc)
generate key using xmlpattern '/dept/name' as sql varchar(30);
- **Retrieve**
select deptdoc from dept where deptID = ?
- **Query**
select deptID, xmlquery('/dept/name' passing deptdoc)
from dept where deptID <> "PR27";

17

SQL/XML – work in both worlds

- Full power of SQL to address structured fields
- Join XML documents and tables
- SQL and XML Predicates
- Create XML from structured fields
- Materialize tables from XML documents



```
select d.deptID , u.headcount, xmlquery('$deptdoc/dept/name' passing d.deptdoc as
"deptdoc")
  from dept d, unit u
 where d.deptID = u.unitID and u.headcount > 200
    and xmlexists('$deptdoc/dept[@bldg = $b]' passing d.deptdoc as "deptdoc",
    u.bldg as "b")
    and xmlexists('$deptdoc/dept/employee/name' passing d.deptdoc as "deptdoc")
```

18

SQL/XML – XML Document Publishing

FIRSTNAME	LASTNAME	DEPARTMENT
SEAN	LEE	A00
MICHAEL	JOHNSON	B01
VINCENZO	BARELLI	A00



```

SELECT
  XMLELEMENT (NAME "Department",
    XMLATTRIBUTES (e.department AS "name" ),
    XMLAGG ( XMLELEMENT(NAME "emp", e.firstname)) AS "department_list"
  FROM employee e GROUP BY e.department;

```



```

department_list
<Department name="A00">
  <emp>VINCENZO </emp>
  <emp>SEAN</emp>
</Department>
<Department name="B01">
  <emp>MICHAEL</emp>
</Department>

```

19

API Support

- XML type is supported in
 - Java (JDBC, SQLJ), ODBC,
 - C/C++, COBOL, PL/I, Assembly
 - .NET
- Applications use:
 - XML as CLOB(n)
 - XML as DBCLOB(n)
 - XML as BLOB(n)
 - All character or binary string types are supported
- XMLParse and XMLSerialize apply (implicitly or explicitly)

20

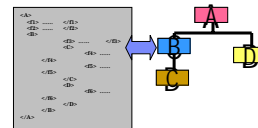
Utilities

- Enhanced to handle new XML type, XML tablespaces, and XML indexes
- CHECK DATA
- CHECK INDEX
- COPY INDEX
- COPY TABLESPACE
- COPYTOCOPY
- LISTDEF
- LOAD
- MERGECOPY
- QUIESCE TABLESPACESET
- REAL TIME STATISTICS
- REBUILD INDEX
- RECOVER INDEX
- RECOVER TABLESPACE
- REORG INDEX
- REORG TABLESPACE
- REPORT TABLESPACESET
- UNLOAD
- Basic RUNSTATS

21

IMS V9 XML Database Storage and Retrieval

- Enables global standard for on demand solutions, providing universal, transparent information interchange among internal business processes, suppliers, partners, customers
 - ▶ Implementation is native IMS
 - ▶ Converts existing IMS data to XML
 - ▶ Decomposes XML data for use by non-XML enabled applications
 - ▶ Identical data descriptions for distributed and host environments
 - ▶ DL/I Model Utility generates XML schema used at runtime
 - ▶ Introduces way to view/map IMS hierarchical data to *native* XML documents, aligning IMS Database Definitions (DBD) with XML Schema
 - ▶ Retrieval/storage of IMS Records as XML documents without change to existing IMS DBs

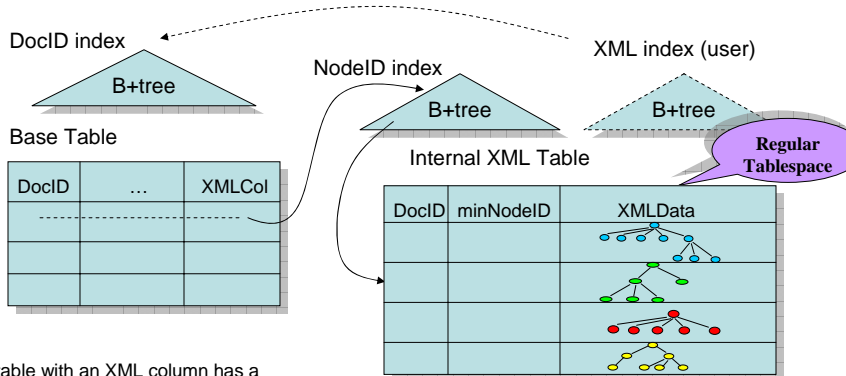


22

Native XML Business Values for Clients

- Reduce time to market
 - ▶ Improved development productivity
- Improve resilience of DBs to rapid requirement changes
- Improve system performance
- Improve system quality & manageability
 - ▶ Reduce complexity
- Flexible, hierarchical structures
- No need for decompose / compose, normalization / denormalization
- High-level declarative languages
- High performance
- Unparalleled reliability, availability, scalability

XML Storage



A table with an XML column has a DocID column, used to link from the base table to the XML table. A DocID index is used for getting to base table rows from XML indexes.

Each XMLData column is a VARBINARY, containing a subtree or a sequence of subtrees, with context path. Rows in XML table are freely movable, linked with a NodeID index.

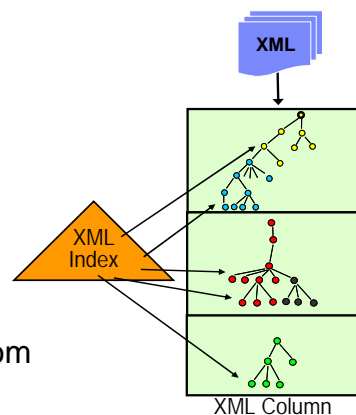
SQL/XML in DB2 9

- Complete SQL/XML constructor functions
 - ▶ Binary data type support
 - ▶ New options for XMLELEMENT and XMLFOREST
 - ▶ XMLPI, XMLCOMMENT, XMLTEXT, XMLDOCUMENT
 - ▶ Allow all the types including XML columns and XMLQUERY result to be used in XML constructors
- Built-in XML type, native storage of XQuery Data Model
- XPath value indexes
- XMLPARSE, XMLSERIALIZE, XMLCAST
- Other SQL/XML functions with XPath
 - ▶ XMLEXISTS, XMLQUERY, XMLTABLE
- Schema repository, Validation UDF
- DRDA (distributed support) and application interfaces
- Utilities

25

What You Can Do with pureXML

- Create tables with XML columns
- Insert XML data, optionally validated against schemas
- Create indexes on XML data
- Efficiently search XML data
- Extract XML data
- Decompose XML data into relational data
- Construct XML documents from relational and XML data
- All the utilities and tools support for XML



26

What can you do with SQL/XML? (1)

```
CREATE TABLE PurchaseOrders (  
  ponumber  varchar(10) not null,  
  podate    date not null,  
  status    char(1),  
  XMLpo    xml);
```

```
CREATE VIEW ValidPurchaseOrders as  
  SELECT ponumber, podate, XMLpo  
     FROM PurchaseOrders  
     WHERE status = 'A';
```

```
ALTER TABLE PurchaseOrders  
  ADD revisedXMLpo xml;
```

27

What can you do with SQL/XML? (2)

```
EXEC SQL BEGIN DECLARE SECTION;  
  SQL TYPE IS XML AS CLOB(1M) xmlPo;  
EXEC SQL END DECLARE SECTION;  
INSERT INTO PurchaseOrders VALUES ('200300001',  
  CURRENT DATE, 'A', :xmlPo);  
INSERT into PurchaseOrders VALUES( '200300001',  
  CURRENT DATE, 'A',  
  DSN_XMLValidate(:xmlPo,myPOSchema));  
UPDATE PurchaseOrders SET XMLpo = XMPpo_backup  
  WHERE ponumber = '12345';  
DELETE FROM PurchaseOrders WHERE ponumber =  
  '12345';  
LOAD into PurchaseOrders ...
```

28

What can you do with SQL/XML? (3)

```

SELECT XMLpo INTO :xmlPo
  FROM PurchaseOrders
  WHERE ponumber = '200300001';
CREATE INDEX ON PurchaseOrders(XMLPO) Generate
  Keys Using XMLPATTERN '//items/item/desc' as SQL
  VARCHAR(100);
SELECT XMLQUERY('//items/item/quantity' PASSING
  XMLpo) FROM PurchaseOrders;
SELECT XMLPO
  FROM PurchaseOrders
  WHERE XMLEXISTS('//items/item[desc = "Shoe"]'
  PASSING XMLpo);

```

29

What can you do with SQL/XML? (4)

```

SELECT TX.*
  FROM PurchaseOrders PO,
       XMLTable ('//item'
        PASSING PO.XMLpo
        COLUMNS
          "Part #"          CHAR(6)          PATH '@partnum',
          "Product Name"   CHAR(20)         PATH 'productName',
          "Quantity"       INTEGER          PATH 'quantity',
          "US Price"       DECIMAL(9,2)     PATH 'USPrice',
          "Ship Date"      DATE              PATH 'shipDate',
          "Comment"        CHAR(80)         PATH 'comment'
        WITH ORDINALITY "Seqno") AS TX
  WHERE PO.ponumber = '200300001';

```

30

What can you do with SQL/XML? (5)

```
<?xml version="1.0"?>
<purchaseOrder orderDate="1999-10-20">
  <shipTo country="US">
    <name>Alice Smith</name>
    . . .
  </shipTo>
  <billTo country="US">
    <name>Robert Smith</name>
    . . .
  </billTo>
  <comment>Hurry, my lawn is going wild!</comment>
  <items>
    <item partNum="872-AA">
      <productName>Lawnmower</productName>
      <quantity>1</quantity>
      <USPrice>148.95</USPrice>
      <comment>Confirm this is electric</comment>
    </item>
    <item partNum="926-AA">
      <productName>Baby Monitor</productName>
      <quantity>1</quantity>
      <USPrice>39.98</USPrice>
      <shipDate>2003-05-21</shipDate>
    </item>
  </items>
</purchaseOrder>
```

Seqno	Part#	Product Name	Quantity	US Price	Ship Date	Comment
1	872-AA	Lawnmower	1	148.95	null	Confirm this is electric
2	926-AA	Baby Monitor	1	39.98	'2003-05-21'	null

31

Registering XML Schemas

- XML Schema Repository (XSR): DB2 supplied user tables
- How to identify a schema?
 - ▶ External names
 - target namespace (e.g., "http://www.w3.org/2001/XMLSchema")
 - schema location (e.g., "http://www.ibm.com/schemas/sample.xsd")
 - used in XML documents and registration
 - ▶ SQL identifier
 - unique identifier in DB, e.g., PURCHSYS."mySampleSchema"
 - used to reference schemas in DDL statements
- Where are schemas used?
 - ▶ DSN_XMLValidate in SQL
 - ▶ Decomposition

32

Annotated Schema-based Decomposition

- A new way of decomposition of XML into relational tables – Shred2.
- Link to XSR (XML Schema Repository).
- An outside engine solution – can decompose into multiple tables.

33

Efficient XPath Support

- Used in XMLEXISTS, XMLQUERY, XMLTABLE
- XPath 1.0 + -
 - ▶ XPath 1.0 constructs in XPath 2.0 semantics
 - ▶ + more data types: xs:boolean, xs:integer, xs:decimal, xs:double, xs:string
 - ▶ + namespace declaration from XQuery prolog
 - ▶ - Axes: only 5 forward axes & parent axis are supported.
- All stored XML data are untyped in V9.
 - ▶ Explicit type casting may be needed in some cases

34

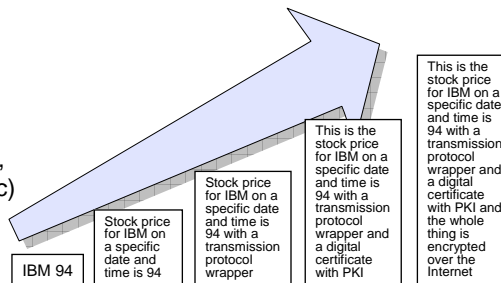
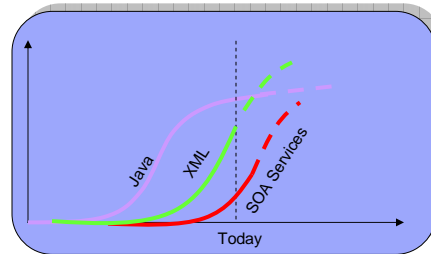
August 2007 announcement - z/OS XML System Services*

1. NEW! z/OS XML System Services is enabled to take advantage of zAAPs. Statement of Direction, at a future date:
2. IBM is intends to enable the z/OS XML to take additional advantage of zIIPs. (i.e. 100% zIIP redirect, greater than the current (about half) for DRDA)
3. IBM also intends to extend and expand the use of z/OS XML System Services with additional future enhancements:
 - IBM intends to enhance the XML Toolkit for z/OS so eligible workloads use z/OS XML. This allows eligible XML Toolkit processing to exploit zAAP.
 - IBM intends to add validating parsing to the z/OS XML component. This extends zAAP and zIIP exploitation to include XML validating parsing workload as well.



Significance of XML

- Since XML can be shared, XML will be a major workload of the future
 - ▶ XML plays a vital role in the development and adoption of SOA (Service-Oriented Architecture) solutions because in many cases, the messages flowing between the SOA services are XML.
- XML is a powerful but costly language (on all platforms)
 - ▶ Text oriented (not binary)
 - ▶ Additional processing (parsing, validation, canonicalization, etc)
 - ▶ Transformations and mapping
 - ▶ Still evolving



XML – a powerful but costly language

- Example: look up IBM stock price

DATA PASSED: IBM 94

5 chars
Traditional

```
SAMPLE CODE:
<SOAP-ENV:Envelope
xmlns:SOAP-ENV=
"http://www.w3.org/2001/06/soap-envelope"
SOAP-ENV:encodingStyle=
"http://www.w3.org/2001/06/soap-encoding">
```

272 chars
with XML
and SOAP

```
<SOAP-ENV:Body>
<Signature Id="MyFirstSignature" xmlns="http://www.w3.org/2000/09/xmldsig#">
<SignedInfo>
<CanonicalizationMethod Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026/">
<SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
<Reference URI="http://www.w3.org/TR/2000/REC-xhtml1-20000126/">
<Transforms>
<Transform Algorithm="http://www.w3.org/TR/2000/CR-xml-c14n-20001026/">
</Transforms>
<DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
<DigestValue>YWJjZGxma3NqamRlZmZnaGlvcztkbGZramFzZGw7Cg==</DigestValue>
</Reference>
</SignedInfo>
<SignatureValue>
YWJjZGxma3NqamRlZmZnaGlvcztkbGZramFzZGw7Cg==
</SignatureValue>
<KeyInfo>
<KeyValue>
<DSAKeyValue>
<P>...</P><Q>...</Q><G>...</G><Y>...</Y>
</DSAKeyValue>
</KeyValue>
</KeyInfo>
</Signature>
```

Over 919
chars
to add
digital
signature
only

And many
more
chars
to add
encryption

37

What is z/OS XML System Services?

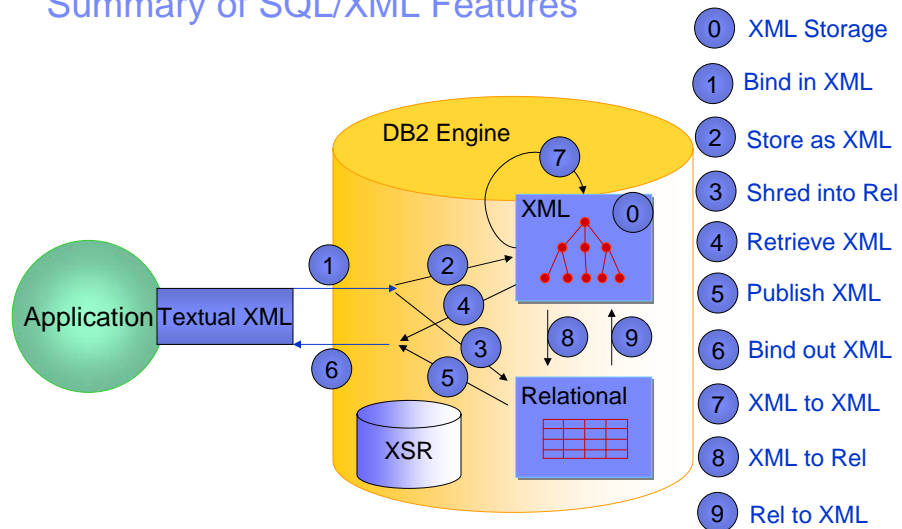
- An XML parser that is an integrated component of the z/OS base (1.8)
 - High performance (short pathlength)
 - Supports unique z/OS environments where minimum overhead is key
 - SRB and TCB modes
 - Cross-memory mode - No Language Environment® dependencies
 - Non-validating parser with well-formedness checking
 - No XML generation or XPath or XSLT processing capability
 - Assembler interface (V1.8), C/C++ interface (V1.9)
 - Available on z/OS V1.7 via SPE
- Simple call model that avoids event-driven interface overhead
- Ability to handle very large documents
- XML documents parsed to a form readily usable by the invoking app
- Intended for z/OS system environments, middleware, and applications that need to handle XML very efficiently
- DB2 9 for z/OS first IBM exploiter (via Assembler interface)

38

z/OS XML System Services is enabled to take advantage of zAAPs

- Middleware and applications requesting z/OS XML System Services will have this z/OS XML System Services parsing eligible to execute on the zAAP.
 - Specifically, all z/OS XML System Services parsing executing in TCB mode will be eligible for the zAAP.
 - ▶ Example: DB2 9 SQL/XML processing via local connection
 - DB2 9 utilizes z/OS XML System Services for a portion of its SQL/ XML.
 - ▶ Example: DB2 9 SQL/XML processing via local connection - executing in TCB mode
- 1) Applications (queries) running locally on z/OS: When DB2 9 inserts or updates XML data, the data has to be parsed and therefore DB2 invokes z/OS XML System Services (and zAAP, when present)
 - 2) Utilities: When XML data is loaded into tables, then the XML data needs to be parsed and therefore DB2 9 invokes z/OS XML System Services (and zAAP, when present)
 - ▶ How much DB2 9 work is eligible for the zAAP will depend on amount of XML data being processed.

Summary of SQL/XML Features



Recap

- *XML is a critical element of the infrastructure and can solve significant business challenges.*
- *Storage and management of XML data represents a rapidly growing opportunity*
- *pureXML technology in DB2 9 has unique capabilities to help unleash the business value of XML data*

DB2 9 pureXML -unique benefits:

- *Lower development costs*
- *Greater Business Agility*
- *Improved Business Insight*

More resources on XML

- XML Guide, SC18-9858
 - ▶ <http://publib.boulder.ibm.com/epubs/pdf/dsnxgk10.pdf>
- Powering SOA with IBM Data Servers, SG24-7259
 - ▶ <http://www.redbooks.ibm.com/abstracts/SG247259.html?Open>
- DB2 9 for z/OS Technical Overview, SG24-7330
- DB2 9 for z/OS Performance Topics, SG24-7473

