C58

XML Capabilities in an iSeries Environment

Leonardo Llames

IBM Advanced Technical Support

Rochester, MN

IBM @server
iSeries
Technical
Conference

Las Vegas, NV **June 17 – 21, 2002**

Agenda

Define XML

- What is XML
- Its Origin

XML's Value

- Business Value
- Technical Value

Look at XML & Related Components

XML on iSeries

Demos

Notes:

We're going to start out with a section on what is XML and where did it come from...

With that background, we can move into a discussion about what is XML good for and why would anybody want to use it.

After that discussion, we'll dig a little deeper into XML and some of the related components required for an XML implementation

Having a bigger picture of the XML world, we'll then explore how the iSeries supports an XML implementation.

Finally, we'll close on a discussion of where XML on the iSeries will be going in the future.

Objectives

Understand Basics of XML

Understand Value of XML in Business Applications

Understand XML Related Components

XML on iSeries

What is XML

XML == eXtensible Markup Language

Heritage

- a subset of SGML (Standard Generalized Markup Language)
  - based on GML (Generalized Markup Language)
  - intended for document processing and publishing
- a sibling to HTML (HyperText Markup Language)
  - used to format information for the Web

XML describes and delivers data

- goes beyond the limits of HTML
- developer can define own markup language
- separates the content from the presentation (formatting)
- allows precise declaration of content
- allows a standardized strategy to define content of documents & databases

Notes:

XML is an abbreviation for eXtensible Markup Language.

XML was created from SGML - SGML is a document language, intended for creating and managing large and complex documents. SGML is used by large corporations and the government to maintain their documents.

XML, like HTML, is intended for information delivery on the WEB. HTML is a very popular means of delivering information to the Web and HTML is simple to learn and to use. However it is becoming inadequate for the volume and variety of information available on the web today.

One of XML's objectives is to be a vehicle for delivering information on the web, enhancing and more often replacing the HTML text available today. HTML and XML are compared later in this presentation.

The real power of XML, (like SGML) is the ability to create tags that have **meaning** to each particular application, unlike HTML which has a fixed tag set.

XML allows the data to retain its meaning and is not concerned with the presentation of the data like HTML. XML has associated components that support the HTML type of presentation (these are discussed later).

XML's real power is:

- **retaining the meaning of data outside the structure**
- **being both platform and vendor neutral**
- **defined as a representation for universal data exchange**
- **extensibility -->> create new dialects and add to existing dialects and documents**

W3C - World Wide Web Consortium

W3C

- organization to establish specifications for Web technologies ensuring the highest degree of utility and interoperability
- a number of companies (Adobe, HP, IBM, MS, Netscape, Sun, ...) represent a variety of requirements and perspectives
- create and review specifications for XML and related components
- the W3C web page for a complete set of reference material is http://www.w3.org

Notes:

The World Wide Web Consortium has the task of both defining and validating the specifications for XML and XML related technologies. It is composed of companies in a variety of industries including:

- IBM & Microsoft (IT)
- Oracle (Database)
- Netscape (Internet)
- Adobe (Publishing)

to ensure their interests are considered in the evolution of these specifications.

XML vs HTML

XML

- Uses <tag> & </tag> style
- Tagged markup for information
  - Focused on data structure
  - Data retains meaning
- Extensible - can define new tags
- Descriptive markup
  - specific search criteria
- Stringent syntax
  - end tags required
    - <paper color="red"></paper>
    - or, <paper color="red"/>
  - element nesting enforced
  - case-sensitive elements
- *Associated components*
- Highly reusable documents
- Requires newer browser
  - Internet Explorer 5.0+
  - Netscape 4.0+

HTML

- Uses <tag> & </tag> style
- Tagged markup for text
  - Focused on presentation
  - Data is text (limited reuse)
- (Relatively) Fixed set of tags
- Document tagging
  - too many hits
- Loose syntax
  - end tags assumed
  - nesting errors effect display
- Simple and complete
- Single use -- for Web
- Works with any browser

Notes:

This chart show the difference between XML and HTML.

- Both XML & HTML are tagged markup languages
  - however XML retains the meaning of the information while HTML focuses solely on the presentation
  - XML data retains its meaning so there is a wide opportunity for reuse of the information
- By having the ability to define new tags within XML - tags are defined that specifically describe the data between the tags
  - for example an XML tag maybe <book title> while in HTML it would be expressed as <H1>
- These descriptive tags allow specifying more discrete searches because you state both the tag name and the content (data) in the search criteria
  - for example if you wanted books written by George GoldFarb you could build a search looking for author name of George Goldfarb this would retrieve only books by that author
- There are a small set of rules for creating an XML document unlike HTML:
  - rules include having both a start and end tag with the same name.
  - tags are case sensitive and space sensitive they must be exact <BOOK> does not equal <book>
  - nesting of elements is possible and is enforced
- Unlike HTML which is usable by any browser - XML has associated components which we will talk about later
- Highly reusable - representing data as an XML document allows that document to be used in a variety of implementations (EDI environments, on the web, transformed to other dialects...)
- XML requires newer browsers to view an XML document - HTML works with any browser.

Looking at XML & HTML

```
<?xml version="1.0"?>
<catalog>
  <product category="equipment">
    <number >SE1357908</number>
    <name>Tennis Racket R13</name>
    <description>Tennis  Racket Pro R13
       </description>
     <price currency="USD">148.00</price>
    <brand>Wilson</brand>
    <color>granite</color>
    <weight unit="pounds" >1.25</weight>
    <availableQuantity>14</availableQuantity>
  </product>
<product category="clothing">
    <number >SW5782098</number>
    <name>Running Shorts</name>
    <description>Running Shorts - Nylon
       </description>
     <price currency="USD">37.00</price>
    <brand>Nike</brand>
    <color>midnight blue</color>
    <size>small</size>
    <availableQuantity>34</availableQuantity>
  </product>
</catalog>
```

```
<html>
<table>
     <tr>
          <td>SE1357908</td>
          <td>Tennis Racket R13</td>
          <td>Tennis Racket Pro R13</td>
          <td>148.00</td>
          <td>Wilson</td>
          <td>granite</td>
          <td>1.25</td>
          <td>14</td>
     </tr>
     <tr>
          <td>SW5782098</td>
          <td>Running Shorts</td>
          <td>Running Shorts - Nylon</td>
          <td>37.00</td>
          <td>Nike</td>
          <td>midnight blue</td>
          <td>small</td>
          <td>34</td>
     </tr>
</table>
</html>
```

Notes:

Each tag set with content within XML is considered an element and has specific meaning for the data associated with it.

For example <name>Tennis Racket R13</name> is considered an element. XML elements can be nested and are structured according to the specific needs of the developer and the data. This is very helpful for retaining the representation of that data in a database.

Later, I will talk about how to control the content of an XML document and how rules can be applied to a XML document.

HTML delivers a presentation style as specified by its tag set. Search engines processing XML documents using both the tag name and the specific search criteria can make more precise searches for information. Intelligent agents using this same concept will be more realistic and usable in the future.

The <?xml version="1.0"?> tag specifies the version of XML being used. "xml" is a reserved word, with a version number (1.0 is the current specification).

Each tag is composed with a start <tag> and an end tag </tag> with exactly the same name and case. The tag names are case sensitive - so <Product> and <product> are two different tags in XML.

Tags can be nested to more closely represent the data.

An empty tag (used for structure purposes) uses the <empty/> construct. An empty tag may have attributes
Example: <Price dollars="5.00"/> where the "/" denotes the end.

Well-formed XML

A small set of rules define the basic syntax for XML documents

- Required 1st line <?xml version="1.0"?>
- Tag with corresponding end tag <tag>data</tag>
- Tag attributes <tag attribute="x">data</tag>

  Shortcut, if no data <tag attribute="x"/>
- Proper nesting and outermost tag
- Example

```
<?xml version="1.0"?>
<goodemployees>
<employee>
    <fullname>Leonardo LLames</fullname>
    <ID>666666</ID>
    <hiredate status="permanent">01/01/2002</hiredate>
</employee>

. . . . . .
</goodemployees>
```

All XML documents must follow these basic syntax rules.

The first line is boilerplate.

For every tag, there should be a corresponding end tag.

Attributes can also be specified within a tag. Generally, there is a finite set of valid attribute values for a given tag. In the example, an attribute of "status" is used within the "hiredate" tag. The value of "status" in the example is "permanent". Since this is employee data, one can conclude that the other valid "status" values can be "temporary", "contractual", etc.

Note that if there is no data provided between the tag and the end tag (assuming that it is optional), the tag and end tag pair can be specified using the given syntax of <tag attribute="x"/>. In the example, if the "hiredate" value is optional and not provided, the statement <hiredate status="permanent">01/01/2001</hiredate> can be written as <hiredate status="permanent"/>

Proper element nesting is also enforced. Tag and end tag pairs cannot cross each others' boundaries. For instance, <fullname><ID>Leonardo LLames</fullname> 666666</ID> is obviously incorrect.

Valid XML Documents

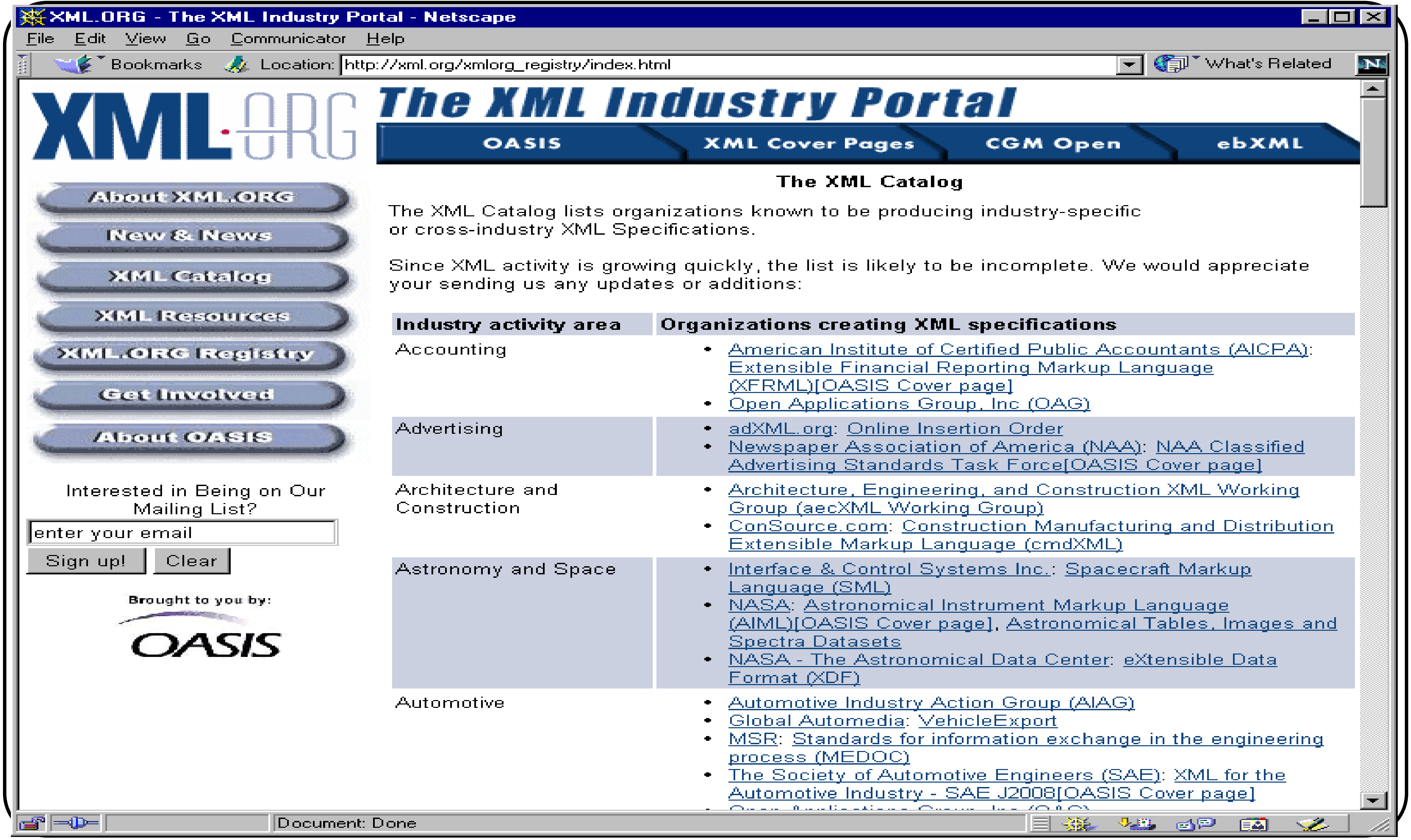
How do we specify:

- tag names that are allowed? Why "employee" and not "empl"?
- which tags can nest within other tags?
- required tags vs. optional tags?
- one occurrence, or any number of occurrences?
- default values of attributes?
- etc.?

A particular XML is called "vocabulary" and is expressed in either DTD or XML Schema.

A "valid" XML document conforms to the basic syntax rules (is well formed) AND complies with a specific DTD or XML Schema

Cross-Industry XML Vocabularies



The XML Catalog

The XML Catalog lists organizations known to be producing industry-specific or cross-industry XML Specifications.

Since XML activity is growing quickly, the list is likely to be incomplete. We would appreciate your sending us any updates or additions:

Industry activity area	Organizations creating XML specifications
Accounting	• American Institute of Certified Public Accountants (AICPA): Extensible Financial Reporting Markup Language (XFRML)[OASIS Cover page] • Open Applications Group, Inc (OAG)
Advertising	• adXML.org: Online Insertion Order • Newspaper Association of America (NAA): NAA Classified Advertising Standards Task Force[OASIS Cover page]
Architecture and Construction	• Architecture, Engineering, and Construction XML Working Group (aecXML Working Group) • ConSource.com: Construction Manufacturing and Distribution Extensible Markup Language (cmdXML)
Astronomy and Space	• Interface & Control Systems Inc.: Spacecraft Markup Language (SML) • NASA: Astronomical Instrument Markup Language (AIML)[OASIS Cover page], Astronomical Tables, Images and Spectra Datasets • NASA - The Astronomical Data Center: eXtensible Data Format (XDF)
Automotive	• Automotive Industry Action Group (AIAG) • Global Automedia: VehicleExport • MSR: Standards for information exchange in the engineering process (MEDOC) • The Society of Automotive Engineers (SAE): XML for the Automotive Industry - SAE J2008[OASIS Cover page]

Looking at XML & Friends

XML Enablers

XML represents data and information about that data

XML works with other related specifications (Enablers)

- DTD Document Type Description
  - defines valid XML Document syntax
  - well formed (w/o DTD) vs valid (DTD used)
- XML Schema
  - an improved XML definition and validation language
  - functional superset of DTDs
- XSL eXtensible Stylesheet Language
  - defines how to format and transform XML data streams
  - more robust than CSS Cascading Style Sheet
- XML Parsers
  - programs used to translate XML into computer usable tree structure
  - provides interface for applications to view & update XML documents
  - validating or non-validating, different programming languages, DOM or SAX

Notes:

XML focuses on representing data without losing the ability to describe that information.
But XML is not complete by itself, enablers are necessary.
.
The W3C have prepared written recommendations for the DTD, XML Schema, XSL and Parser and are no longer just "working specs".

Some of these are currently being used in products and tools today.

These enablers may be enhances/extended by other enablers that are more clearly defined or robust. However, the effort of creating the XML environment is not wasted.

DTD Document Type Description

Optional but should accompany a XML document

- ensures absolute data consistency

Defines the rules of the document

- which elements and attributes (if any) must be present
- structural relationship between the elements

Helps to validate data being received by an application

XML

```
<books>
<book type="fiction">
   <title>                </title>
   <author>           </author>
   <isbn>             </isbn>
   <reader-level>  </reader-level>
</book>
</books>
```

DTD

```
<!-- books.dtd                                    -->
<!-- books is the root of the document  -->
<!ELEMENT books (book+)>
<!ELEMENT book ( title, author+, isbn,
          reader-level? )>
<!ATTLIST book
     type (fiction | nonfiction) "fiction">
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT reader-level (#PCDATA)>
```

Notes:

A DTD is used to define a valid XML document.

A DTD is used to define
- valid vocabulary of element type names for use in tags of a document
- nesting rules and valid attributes
- allows validation by XML Parser

A DTD declares what constitutes valid markup for a specific document type and it declares the rules about the elements used (ie. relationships, occurrences, valid values...). A DTD can be as simple as the data represented in the XML document.

An XML document is validated by a parser having a DTD because the parser can apply constraints to the elements within the DTD.

DTD Document Type Description

```
<!-- comments   -->
```

```
<!ELEMENT books (book+)
```

(books contain book objects, in this case 1 or more)

```
<!ELEMENT book ( title, author+, isbn, reader-level? )
```

(book contains these child elements -

- title and isbn ("no indicator" means appears once)
- author - must be present (+ = one or more)
- reader level optional (? = once or not at all))

```
<!ATTLIST book
      type (fiction | nonfiction) "fiction">
```

Attributes for book element, default is fiction

```
<!ELEMENT title (#PCDATA)>
```

(#PCDATA - Parse Character Data - a reserved name - denotes character content)

```
<!ELEMENT author (#PCDATA)>

<!ELEMENT isbn (#PCDATA)>

<!ELEMENT reader-level (#PCDATA)>
```

Notes:

This example shows the basic syntax of a DTD for our book XML doc.

DTDs provide a specific set of rules to follow that we will not study here. But as an overview, notice after the book element in the list, each element has a specific nomenclature (a "+", a "?" or a blank):

- this nomenclature specifies the rules of the occurrence of an element within the document.
- "+" - the element must occur once and may be multiple
- "?" - the element may occur only once or not at all
- " " (blank) - the element must occur once
- "*" - the element may exist once, multiple or not at all
- "choice1|choice2" - the element must occur once and can only be "choice1" or "choice2"

Each element is listed individually to allow definition of its own characteristics. These characteristics can be #PCDATA (for Parse Character Data) as shown in our example for "title". Specific values or other keywords (like ANY which allows any data type, or EMPTY which has no value but can be a place holder for future use or indicate another level in the structure).

Along with #PCDATA can be sub elements which allow definition of specific valid values. For example, reader level may only allow values like beginner, intermediate or advanced as valid values.

XML Schema

W3C Recommendation

Functional superset of DTD's

Allow additional constraints to be defined

- strong type support, e.g., data and numeric element types
- min/max length
- min/max values
- pattern, enumeration, precision, etc.

Unlike DTDs, XML Schemas are well formed XML documents

- DTDs' #PCDATA can be **any** string of characters

XML Schema

Example: simpleType with min / max length

- Defines a new string type, called streetaddress that may contain 1 to 30 characters
- Uses facets minLength and maxLength to constrain the length

```
<simpleType name="streetaddress"
                      base="string">
     <minLength value="1"/>
     <maxLength value="30"/>
</simpleType>
```

XML Schema

The Schema

```
<schema>

  <element name="pOrder" type="POrder"/>

  <complexType name="POrder">
     <element name="shipTo" type="Address"/>
     <element name="billTo" type="Address"/>
     <attribute name="poDate" type="date"/>
  </complexType>

  <complexType name="Address">
     <element name="name" type="string"/>
     <element name="zip" type="integer"/>
  </complexType>

</schema>
```

An Instance

```
<pOrder poDate="1999-05-20">

   <shipTo>
      <name>Jo</name>
      <zip>95944</zip>
   </shipTo>

   <billTo>
      <name>Jo Sr.</name>
      <zip>20012</zip>
   </billTo>

</pOrder>
```

XSL eXtensible Stylesheet Language

Has two roles

- *transform* an XML document into another document or HTML
  - can create new formatting tags and properties
- *render* XML data for presentation
  - fonts, size, color, alignment
  - provide rules for ordering presentation

Value

- more formatting flexibility based on context or element position

Popular Stylesheet Engines (Java based)

- LotusXSL
- Xalan, open-source (Apache.org) version of LotusXSL

Notes:

XSL is being used to format and transform XML documents.

Transforming - Stylesheets can specify the rules for transforming an XML document into another XML Dialect or HTML. This allows a consistent form of the XML document to provide a base for a variety of other documents, which are used by a variety of devices and applications.

Formatting - different XSL files can be created for the same XML Document. The allows the user to select their personal favorite as a means to display the information. For example, one person may prefer to see information represented in a table which someone else may like an indented list. XSL takes XML far beyond HTML because of this particular feature.
The order of the information can be modified within the XSL Stylesheet as well, which makes the whole environment more user friendly.

XSL Example - formatting to Browser

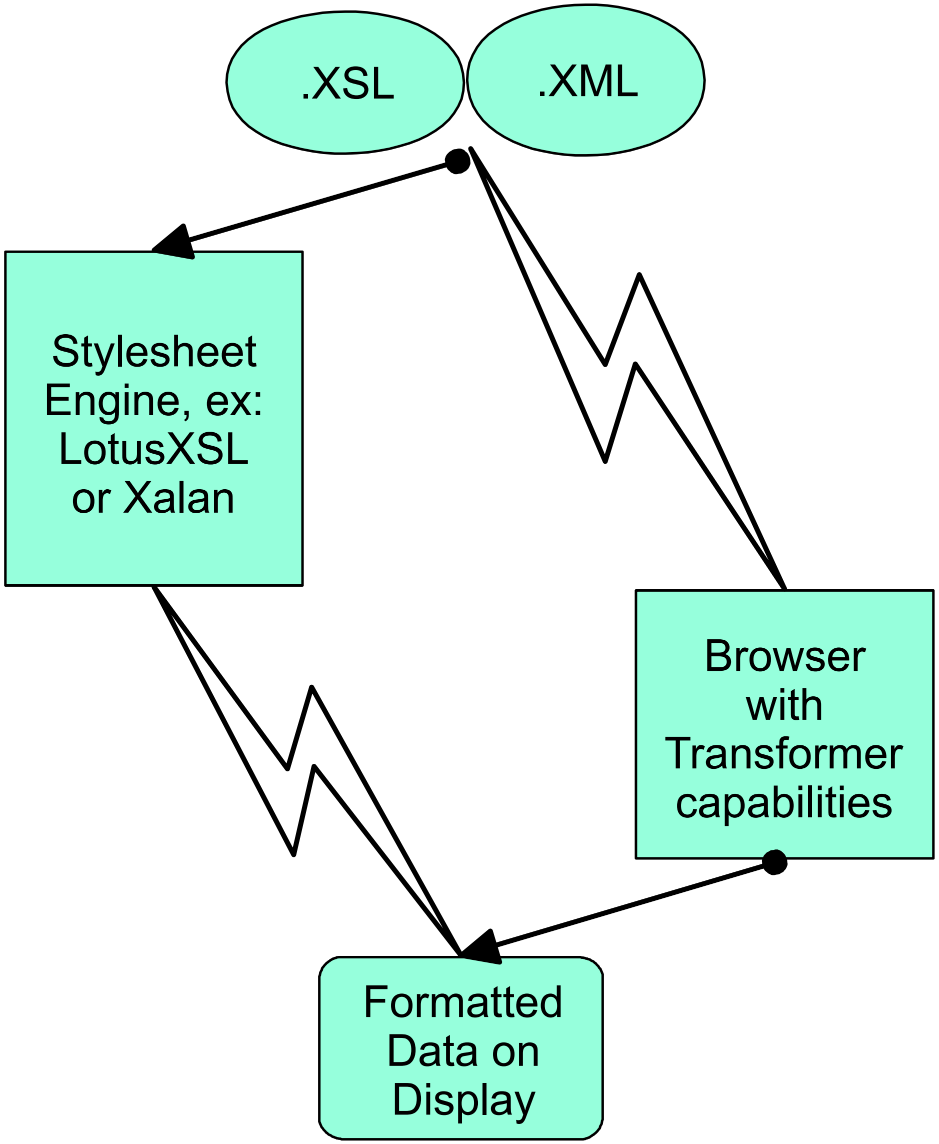
```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" indent="yes"/>

<xsl:template match="/">
     <xsl:apply-templates select="books/book"/>
</xsl:template>

<xsl:template match="books/book">
    <html>
      <head>   <title>Book Club List</title>   </head>
      <body>
        <h1><xsl:value-of select="title"/></h1>
        <xsl:apply-templates select="author"/>
      </body>
    </html>
</xsl:template>

<xsl:template match="author">
     <p><xsl:value-of select="."/></p>
</xsl:template>

</xsl:stylesheet>
```


.XSL
.XML
Stylesheet Engine, ex: LotusXSL or Xalan
Browser with Transformer capabilities
Formatted Data on Display

Notes:

This example shows how an XSL document might look. As you can see, it determines how the XML document is to be presented.

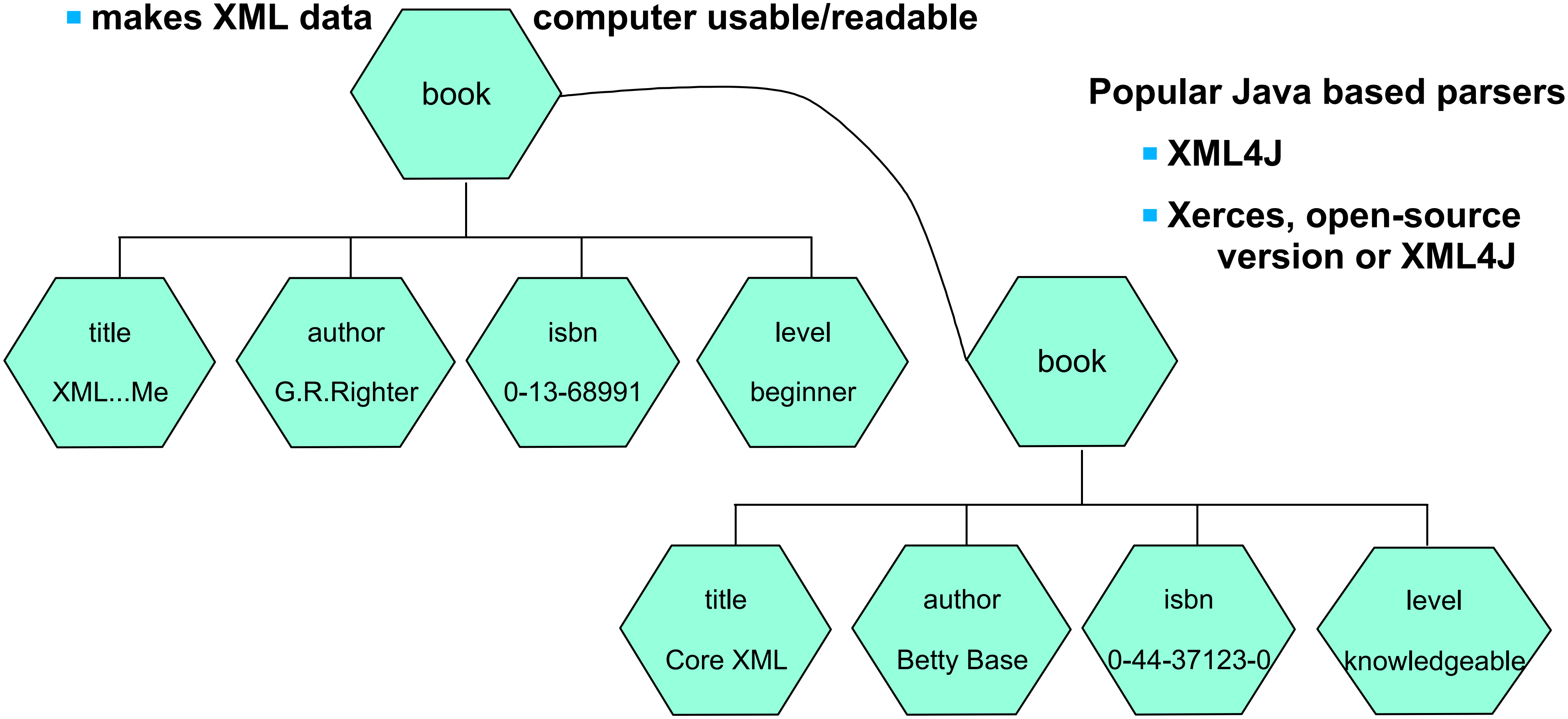
The developer uses a set of templates to match certain nodes in the XML document.

Rules can be included which order the presentation of sections - our presentation could be ordered by different elements.

XML Parsers

Well defined syntax and use of DTD (tag definition) enable reusable parsers

- **DOM (Document Object Model) - standard API parser support for viewing and manipulating XML documents.**
- **SAX (Simple API for XML) - event driven. Instead of getting a complete DOM tree, get notifications of arrival at each element.**
- **makes XML data computer usable/readable**



Popular Java based parsers

- **XML4J**
- **Xerces, open-source version or XML4J**

Notes:

IBM has an excellent and very popular XML parser (XML4J for Java and XML4C for C++) available free on the alphaWorks web site. The alphaWorks web site is shown later in the presentation. The alphaWorks team mission is to provide early adopter developers direct access to IBM's emerging "alpha-code" technologies. They make the latest software technologies available for download and evaluation.

The parser allows the XML document to be transformed into a tree which is usable by the computer.

This tree allows easy searching and manipulation of the data within the tree.

The SAX API is another alternative and supported by most Parsers. SAX stands for Simple API for XML - The SAX API is an event driven parser that notifies you when certain events happen as it parses your document. When it encounters a start tag or an end tag, it will present information on those events that your program can process as required. For example, the program could check for the reader level element and change "beginner" to "starter".

Why would you use SAX or XML4J? If your document is very large, using SAX will save significant amounts of memory when compared to using XML4J. This is especially true if you only need a few elements in a large document. On the other hand, the rich set of standard functions provided by XML4J isn't available when you use SAX.

DOM Parser

Document Object Model (DOM)

W3C Recommendation

Uses a "tree' representation of an XML document

- Tree is created as a result of parsing a document
- Navigates tree by node

SAX Parser

Simple API for XML (SAX)

A de-facto standard by Dave Megginson, not from W3C

Event based parser presents elements as encountered

- Instead of getting a complete DOM tree, get notifications of arrival of each element
- Less memory required, suitable for large documents

Support available from Xerces parser

Defines a number of events, you can write code to implement behaviors for each. Examples:

startDocument() & endDocument()

startElement(String elementName, AttributeList attrs) & endElement(String elementName)

characters(char ch[], int start, int length)

Some Other XML Technologies

Namespaces

- W3C recommendation 01/1999
- collection of names, identified by a URI reference [RFC2396], which are used in XML documents as element types and attribute names
- Analogous to a field reference

XLink - XML Linking Language

- W3C Recommendation 1.0, June 27th 2001
- Link XML elements among separate XML documents
  - simple links similar to HTML <a href="http://www.w3c......
  - more sophisticated links, bi-directional, etc.
- Reduce data redundancy and functional dependencies
  - Sounds familiar?
  - Similar to DB normalization at 3NF?

Some Other XML Technologies

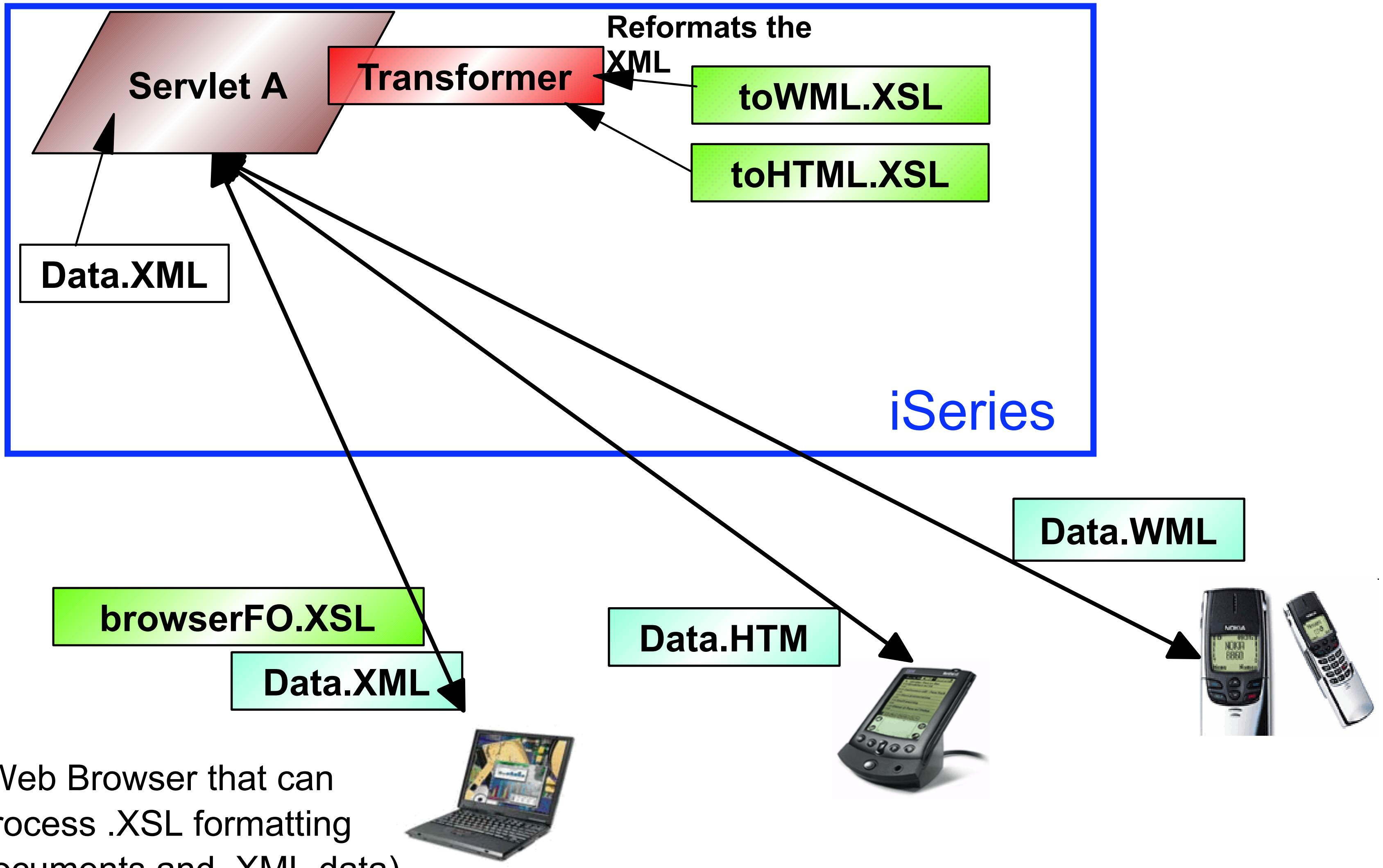
XPointer

- W3C Candidate Recommendataion 09/11/2001 - 03/04/2002
- Based on the XML Path Language (XPath), supports addressing into the internal structures of XML documents, identified by URIs (Uniform Resource Identifiers)

SOAP (Simple Object Access Protocol)

- simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralized, distributed environment using XML
-
  ```
  <SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
          <m:GetLastTradePrice xmlns:m="Some-URI">
              <symbol>DIS</symbol>
          </m:GetLastTradePrice>
  </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
  ```

XML - datastreams to various devices



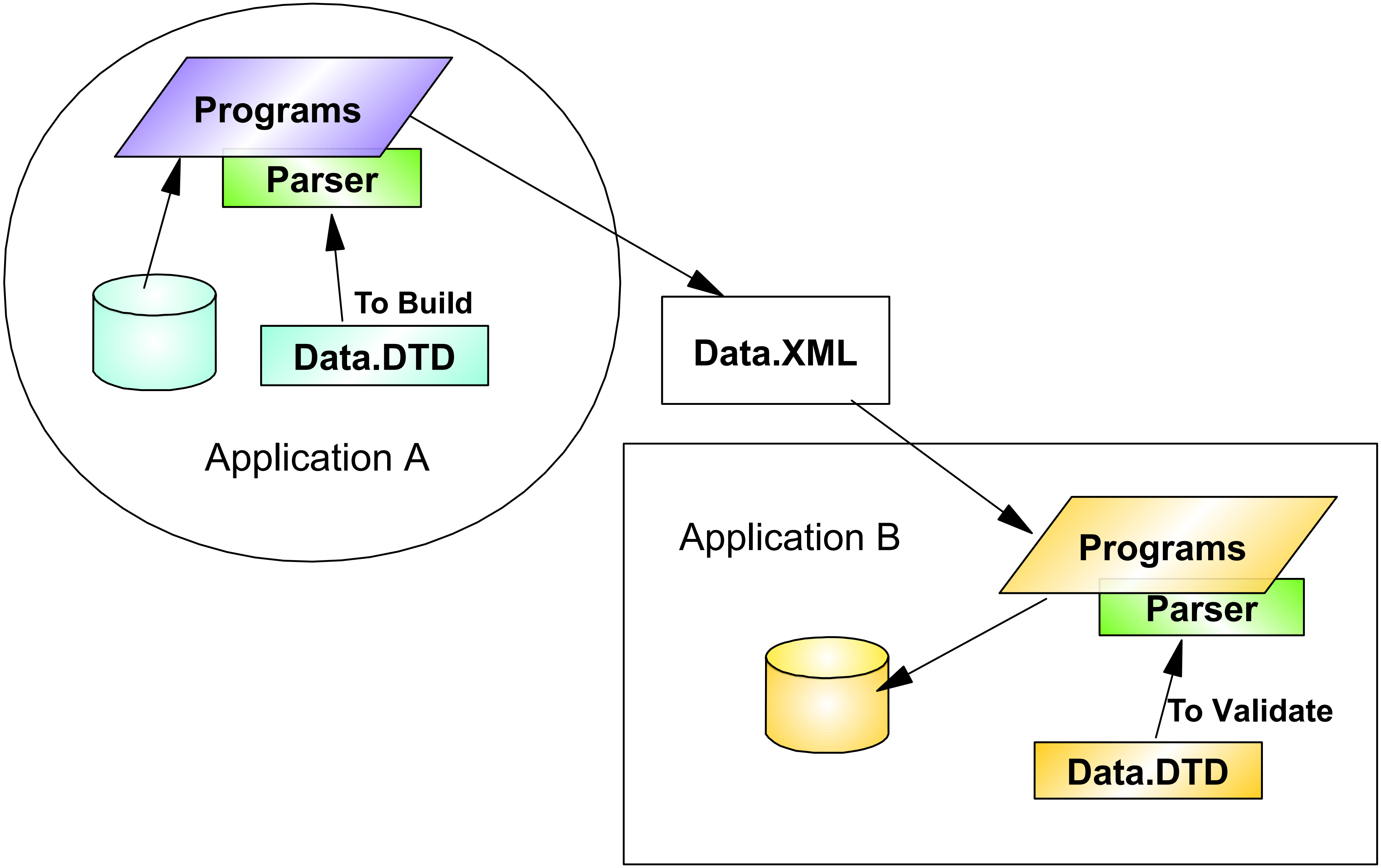
(Web Browser that can process .XSL formatting documents and .XML data)

Notes:

This example shows the use of a common XML document used by a variety of devices. This is accomplished by using XSL and the transformer.

The device could invoke a servlet on an iSeries by using a user-agent to determine the device. The servlet could then invoke the XSL transformer to reformat an XML document using an appropriate XSL stylesheet and create a resulting document that could be sent to the requesting device.

Example: Between applications



Notes:

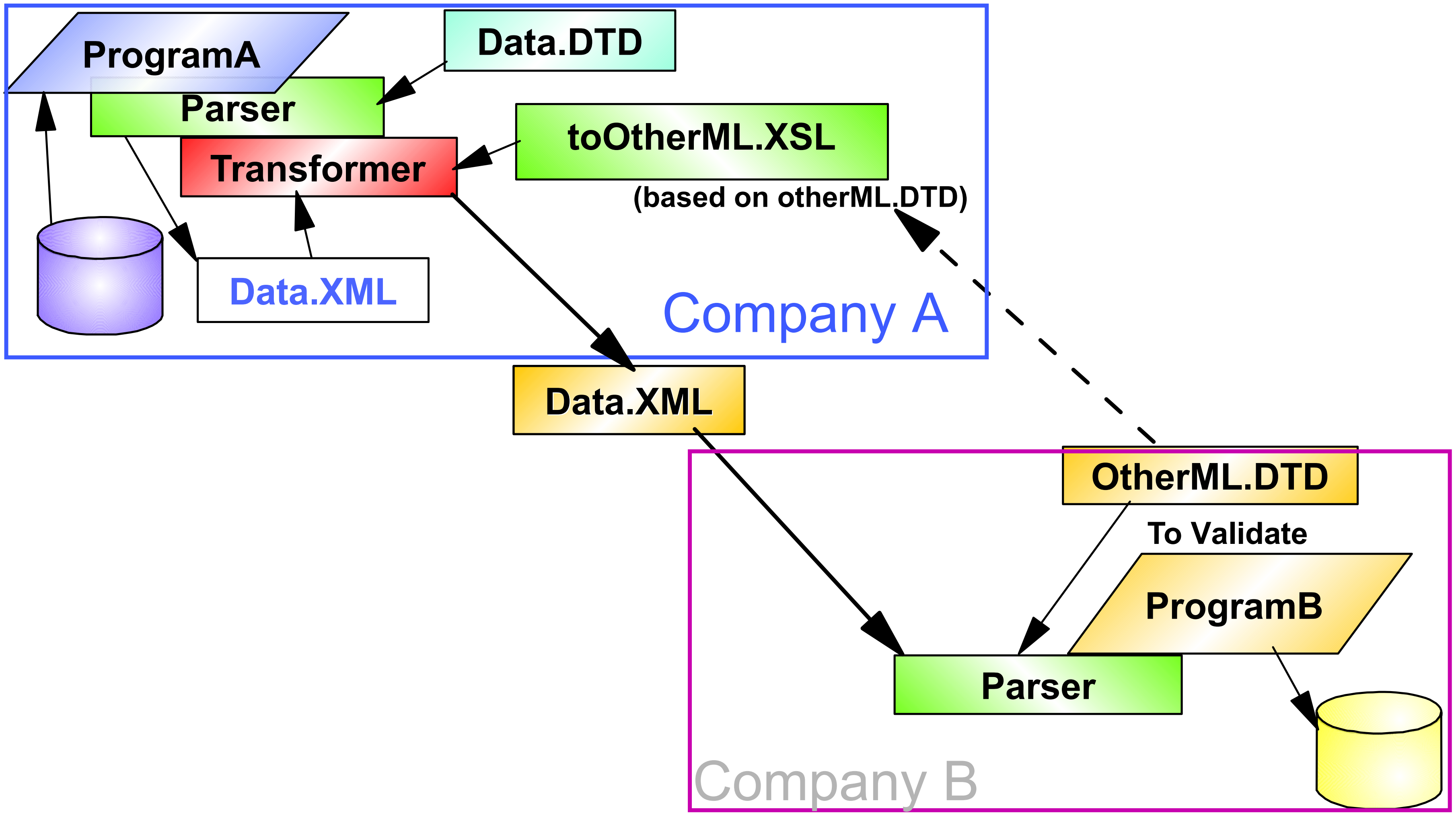
In this example, two different applications utilize XML. The programs use the DTD to ensure the XML document format created in the first program and sent to the second program meets the specifications for that data exchange.

Application A has a class/program that uses the parser to take data from a datastore to create an XML document. The parser uses the DTD to ensure the document is valid.

Application B has a program that processes the XML document by using the parser to read the document. The DTD is used by the parser to validate the document.

These two XML enabled applications could be used within a single company or they could be two XML enabled applications used by different companies using XML as the exchange format.

Example: B2B (unique XML dialects)



Notes:

This could be an example of a Business to Business (B2B) data exchange.

This example shows two companies that are communicating using XML to express the data.

Normally, they would use a common XML dialect. However, this picture show the two companies using different XML dialects. Communication is simplified by using XLS and the transformer.

Company A has created an XSL document with the rules to transform their XML data to the other dialect and a program to create an XML document using that stylesheet and an XSL transformer to the other dialect.

Company B can validate that XML document using the parser and their copy of the DTD.

XML on iSeries

XML and iSeries

WebSphere® Application Server (all editions)

- Includes XML4J (Java™) parser for XML and DTD library
- Includes LotusXSL processor

IBM Toolbox for Java (V4R4 onwards) and JTOPEN

- Panel Definition Markup Language (PDML)
  - XML used for defining GUI layout and components
  - Visual GUI builder and tool to convert from Microsoft® GUIs to XML
  - Runtime to generate Java/Swing GUI from PDML
- Program Call Markup Language (PCML)
  - iSeries program interface defined in XML
  - Runtime framework supporting program calls from Java
- Includes XML4J (Java™) parser for XML
  - XML4J parser used for both PDML and PCML

Notes:

WebSphere® provides the basic enablers to create XML documents and has plans to increase its capabilities with XML.

On the iSeries, V4R4 utilizes XML in our PDML (Panel Definition) and PCML (Program Call) Markup Languages. They are part of the IBM Toolbox for Java and are ready to use. The toolbox also includes the XML4J parser for working with the XML data.

JTOPEN is the open source version of the IBM Toolbox for Java

PDML (Panel Definition Markup Language) - a user interface framework to provide a productive development environment for building graphical panels. The framework automatically handles the exchange of data. The developer only needs to create one or more data beans and bind them to the panel components using PDML tags.

PCML (Program Call Markup Language) - a program-called framework, provided via a tag language used for supporting the program call function of the toolbox. The language fully describes all parameters, structures, and field relationships necessary to call an iSeries program.

iSeries development is using them today to create GUIs for the iSeries and to call between Java and other languages.

We use the IBM XML parser to interpret the PDML and PCML documents.

XML and iSeries

XML Toolkit for iSeries (5733-XT1)

To ensure latest versions of Apache XML parsers, currently Apache 1.6

- Includes XML Schema support

XML Parser for C++ (XML4C 4.0)

- Service program QXMLTOOLS/QXML4C400
- is available at http://www.alphaworks.ibm.com

XML Parser for Procedural Languages (XML4PR 4.0)

- RPG, C and COBOL
- Service program QXMLTOOLS/QXML4PR400
  - Facilitates the use of XML as both a datastore and IO mechanism in these languages.
  - XML version 4.0 parser API documentation, sample, and include files. The C, RPG, and COBOL development environment is installed in library QXMLDEV400. The C++ development environment is installed in the integrated file system directory

Notes:

In addition to the XML4J - the Java parser, a C++ parser for XML is available through IBM's alphaWorks web site at http://www.alphaworks.ibm.com for download, evaluation and use. Having both Java and C++ XML parsers provides more opportunity for customers/solution developers to incorporate XML enhancements in their current application programs. There is also an XML Interface for procedural languages, XML4PR.

As of V5R1, these are now packaged and supported in XML Toolkit for iSeries (5733-XT1). This ensures that the latest supported versions of the C++ and procedural parsers will be available.

WebSphere XMLConfig Tool

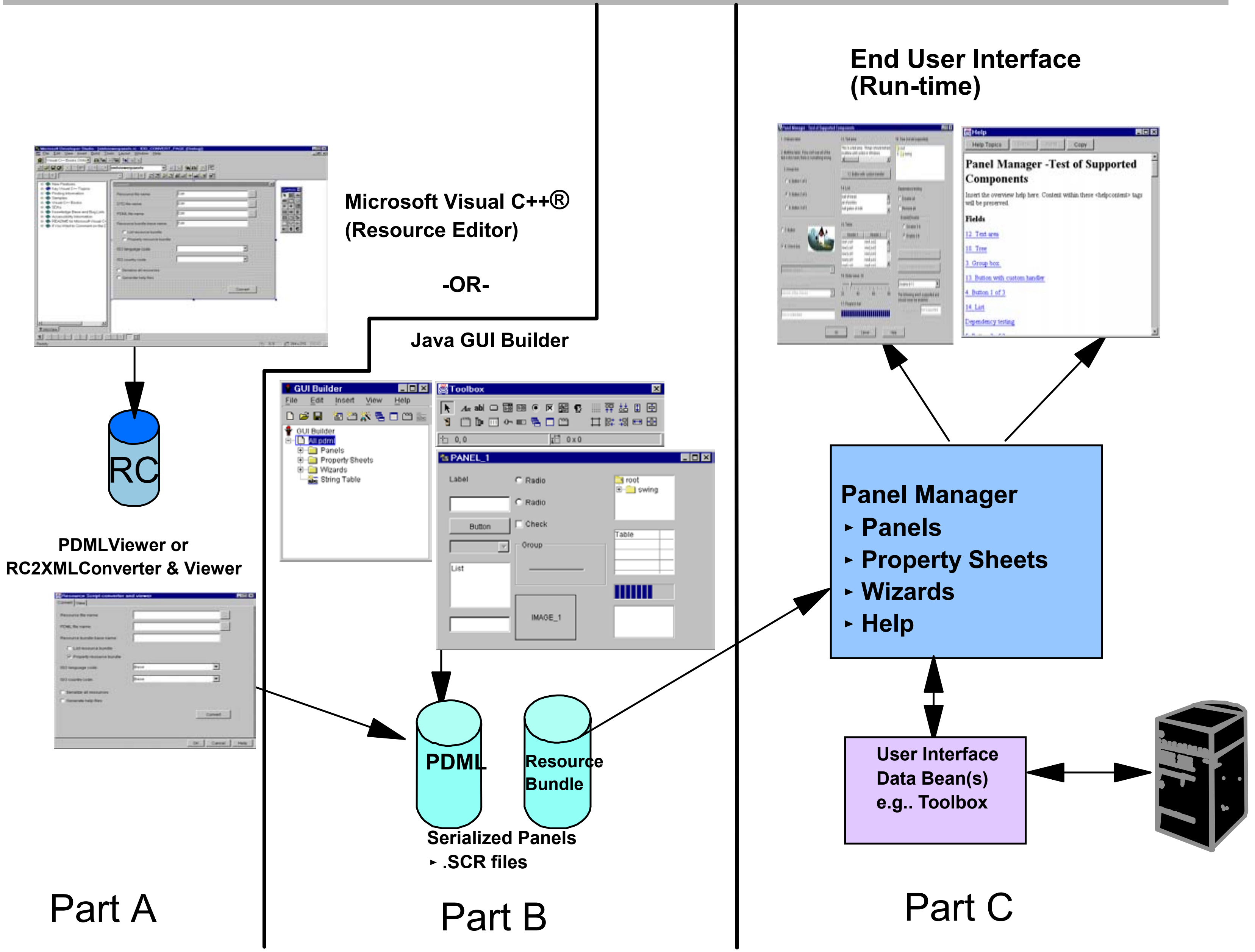
XML Usage for configuration

XML Configuration Management Tool (XMLConfig)

- Allows you to import and export configuration data to and from the WebSphere Application Server administrative repository.
- Uses XML documents to perform the same tasks that can be done in the WebSphere Administrative Console.
- Part of V3.5 and later versions of WebSphere Application Server. Available in version 3.02 of WebSphere as a technology preview.

The XML-based approach complements the administration you can perform through the WebSphere Administrative Console.

iSeries Usage of XML - PDML



End User Interface
(Run-time)

Microsoft Visual C++®
(Resource Editor)

-OR-

Java GUI Builder

PDMLViewer or
RC2XMLConverter & Viewer

Panel Manager

- Panels
- Property Sheets
- Wizards
- Help

User Interface
Data Bean(s)
e.g.. Toolbox

Serialized Panels

- .SCR files

Part A

Part B

Part C

Notes:

This picture shows how an existing Visual C++ display and a Java GUI can be turned into XML.

The XML can then be processed by the Panel Manager - to display the end user interface in a graphical form.

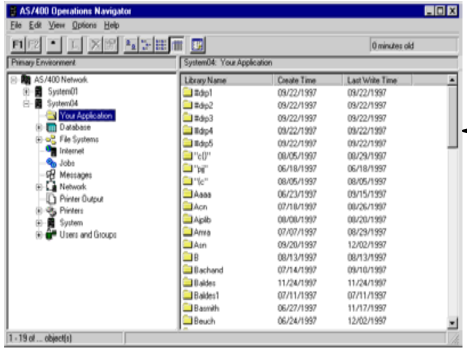
Part A shows taking an existing Microsoft Visual C++ user interface and using the RC2XML converter to create a PDML document representation of it.

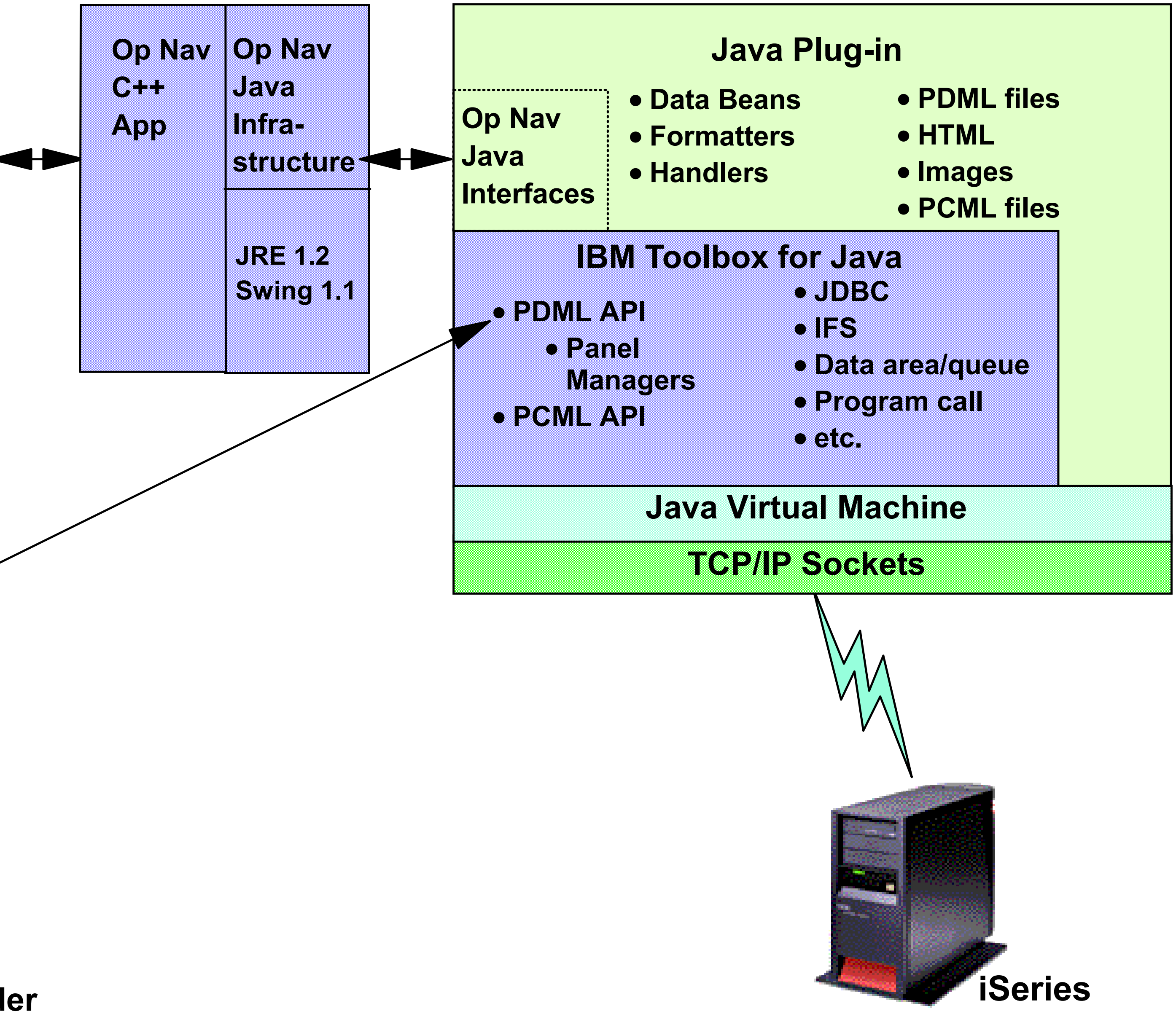
Part B shows using the GUI Builder to design a GUI which is represented as a PDML document.

Part C shows using the PDML and Resource Bundle as input to the Panel Manager at run time. Capabilities in the ToolBox support using the GUIs at run time.

Operations Navigator + Java Plug-ins

Operations Navigator





Op Nav C++ App

Op Nav Java Infra-structure

JRE 1.2 Swing 1.1

Java Plug-in

Op Nav Java Interfaces

- Data Beans
- Formatters
- Handlers

- PDML files
- HTML
- Images
- PCML files

IBM Toolbox for Java

- PDML API
  - Panel Managers
- PCML API

- JDBC
- IFS
- Data area/queue
- Program call
- etc.

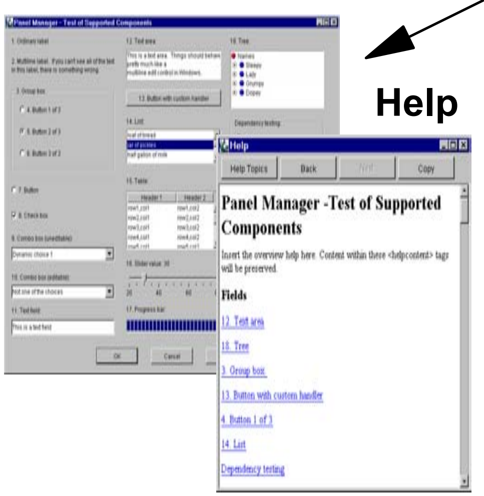
Java Virtual Machine

TCP/IP Sockets

iSeries

Panels

Help



Panel Manager -Test of Supported Components

Insert the overview help here. Content within these <helpcontent> tags will be preserved.

Fields

12. Text area

18. Tree

3. Group box

13. Button with custom handler

4. Button 1 of 3

14. List

Dependency testing

Created with the GUIBuilder

Notes:

This picture shows the ToolBox and its capabilities that support both the creation of GUIs using the GUI Builder and the run time environment.

Specifically, this picture shows how the Operations Navigator product is presented to the user.

Displaying your Panels at runtime

Graphical Toolbox runtime environment

Handles all data exchange between UI controls & Javabeans identified in the PDML

Validates user data

- common integer and character data types
- allows custom validation

Defines standardized processing of Commit, Cancel & Help events

Manages interactions between UI controls based on state information defined in PDML

Notes:

The toolbox runtime environment handles the manipulation of the GUI.

iSeries Usage of XML - PCML

- Program Call Markup Language
  - An XML-based language to define iSeries program interfaces
  - PCML describes an iSeries program interface
    - Used to drive automatic data translation
- com.ibm.as400.data.ProgramCallDocument
  - A Java class representing the interfaces described in a PCML source file
  - Performs data conversion and program calls to the iSeries

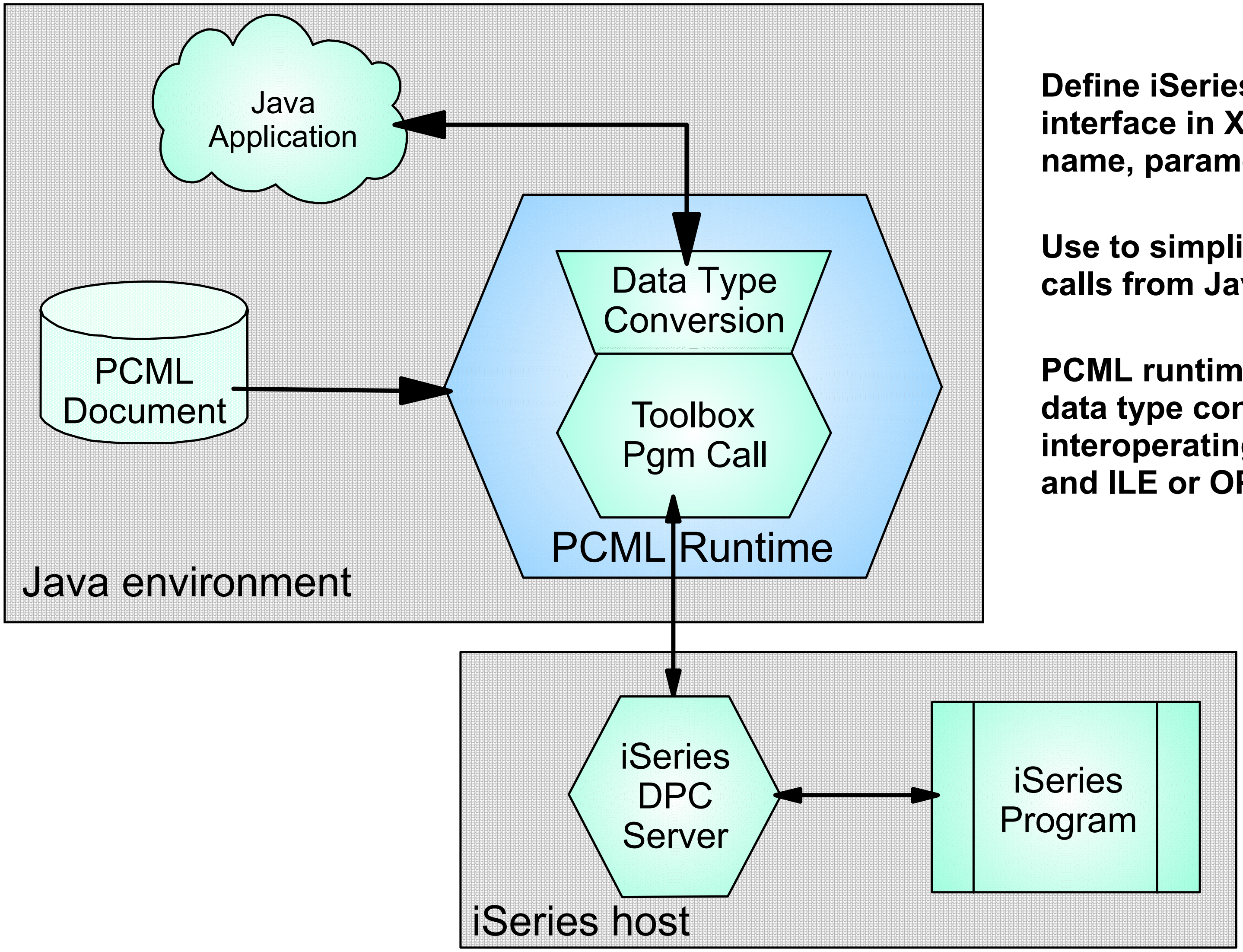
iSeries data type	Java data type
4 byte binary	Integer
Packed decimal	BigDecimal
Zoned decimal	BigDecimal
Text	String
and all the others . . .	

Notes:

PCML makes communicating from Java to AS/400 legacy applications easier.

By using the ProgramCallDocument class a Java application can send data to and receive data from a legacy application, without have to add extra code.

PCML - Details



Define iSeries program interface in XML (location, name, parameters/data types)

Use to simplify iSeries program calls from Java

PCML runtime handles required data type conversions when interoperating between Java and ILE or OPM languages

Notes:

PCML is a tag language that helps you call iSeries programs from Java while writing less JAVA code. This works for both local (same system) and remote calls.

You can describe the input and output parameters for the iSeries program

Previously, to use the Toolbox, the Java developer had to write more to construct the Toolbox calls for performing data translation.

Now your calls for Java to iSeries are automatically handled by ToolBox ProgramCallDocument objects. These ProgramCallDocument objects are generated from the PCML tags.

To increase run time performance, the ProgramCallDocument can be serialized during your product build time.

During runtime, your Java application uses the ProgramCallDocument.setValue() method to set the input values for the iSeries program call. Likewise your Java application uses the ProgramCallDocument.getValue() method to retrieve output from the iSeries program.

XML/XSL Development Tools on iSeries

- Development tools can assist in definition of XML documents, DTD and XSL
- XML development tools are mostly client based technologies
- Tools can create XML, XSL, DTD... that can be used on iSeries
- Examples
  - VisualAge for Java with ET/400
  - WebSphere Studio with iSeries affinity
  - WebSphere Studio Application Developer
- For the latest on tools, check out:
  - www.alphaworks.ibm.com
  - www.oasis-open.org
  - and others...

Notes:

XML and the related enablers have tools that can assist in the definition and utilization of those structures. The tools are client based and create platform independent output. Once created, XML documents, stylesheets and DTDs can be used on the iSeries.

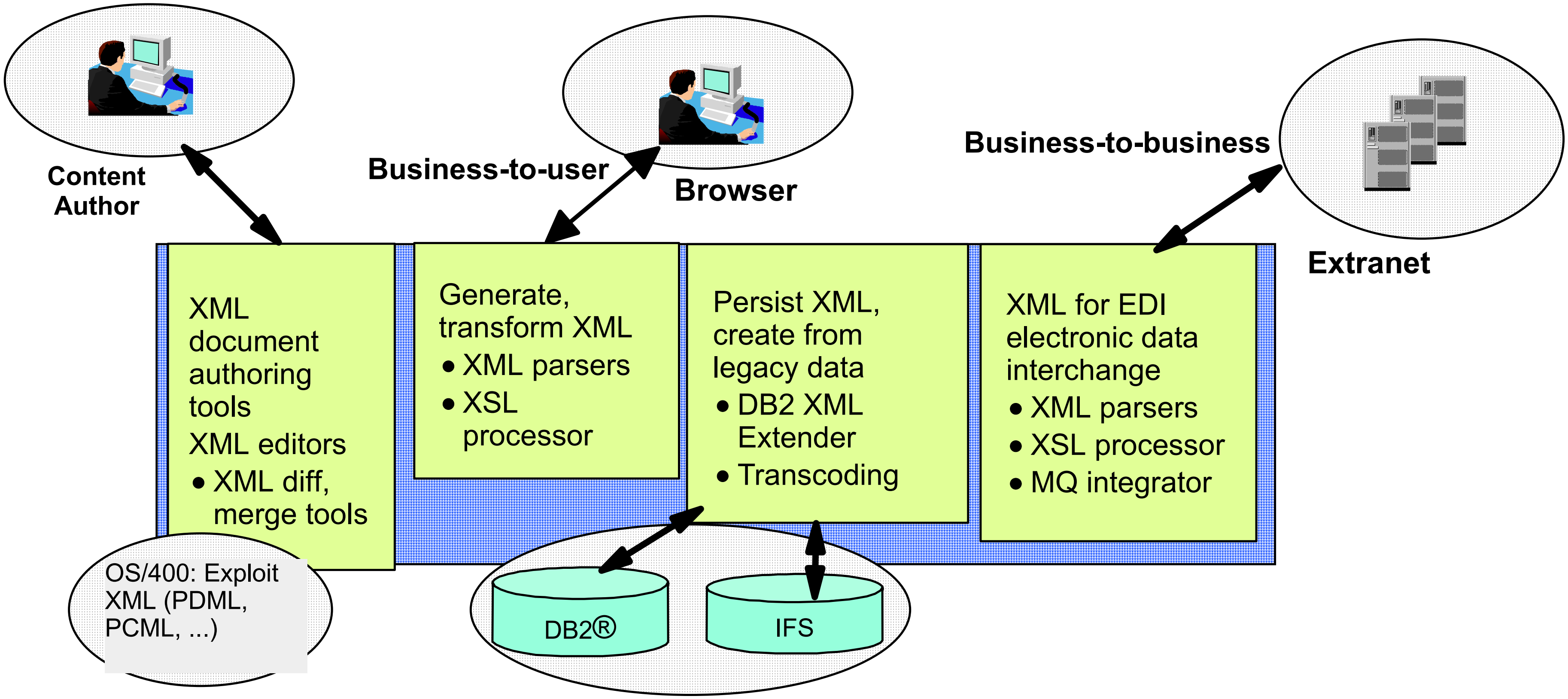
Tools continue to be added and once added, they evolve so it would be appropriate to frequently check for their current capabilities. The web pages are an excellent resource for that information.

Supplements to XML on iSeries

XML for iSeries Objectives

Establish iSeries as a fully enabled platform for exploitation of XML technologies:

- Business-to-business transactions using XML for data interchange
- Business-to-user provide business data in XML for viewing
- Keep the iSeries easy to use for Appl. Development and system administration tasks
  - Example: provide a logical view of relational schema (ala DDS)



Notes:

iSeries is a system built for business. We see XML as a key player is new e-business application environment.

There are at least three ways for businesses and the iSeries to take advantage of XML:

- **Business to Business** by providing a vendor and application neutral interface between businesses and business applications, everyone benefits by making the data easy to accept and process.
  - company to supplier data exchange
  - financial transactions
  - insurance
  - ...
- **Business to User** having business information represented in XML enables:
  - presentation by user preference
  - more specific searches
- **iSeries Application Development** made even easier
  - by improving using XML we plan to make information about the iSeries easier to use
  - DDS has been the way to express Data structures and Displays with XML this can be made platform neutral while remaining easy to develop

DB2 UDB XML Extenders

XML represents a fundamental change in computing ... away from proprietary file and data formats to a world of open interchange

XML = portable data

DB2 UDB provides stability, scalability and security

Your mission-critical business data is currently stored in DB2 UDB

Where do you store your XML documents?

How can you convert your business data into XML documents?

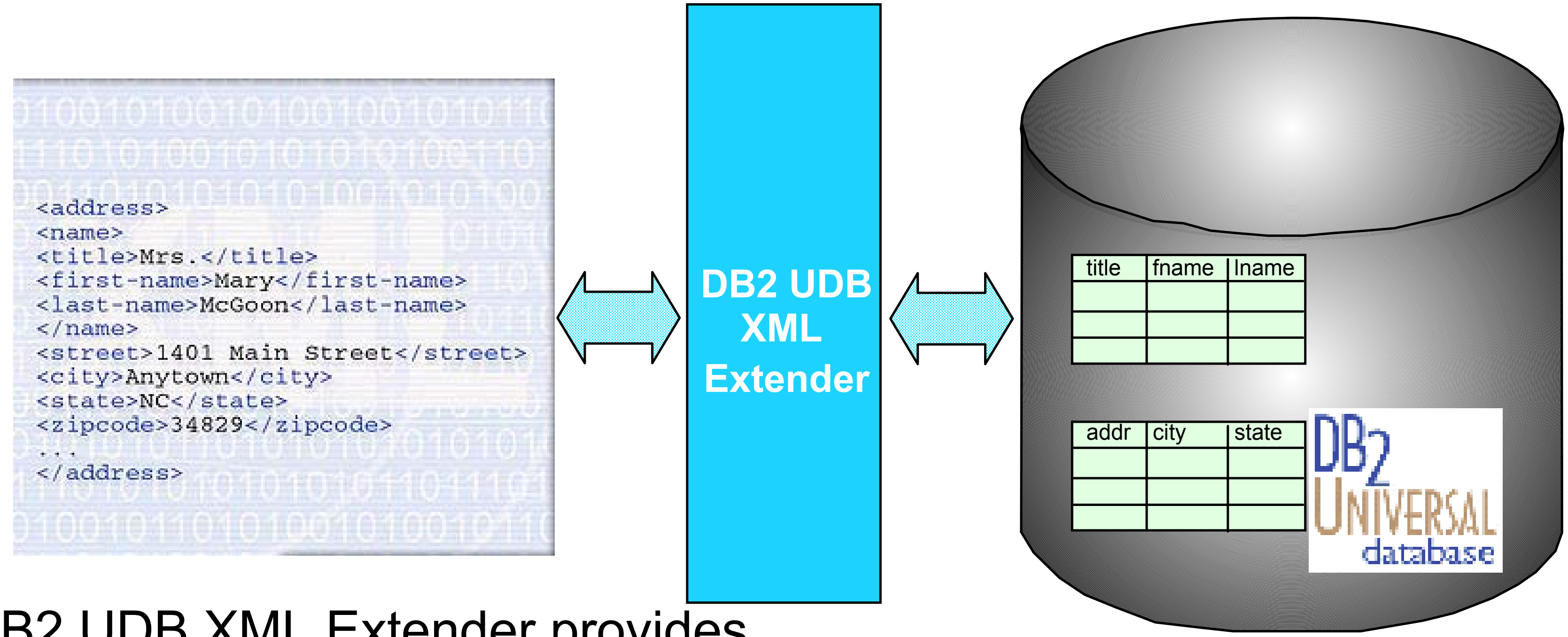
How can you turn the XML data into database data?

➡ Answer: DB2 UDB for iSeries Extenders

DB2 UDB XML Extender

XML Extensions to DB2 UDB

- Focus on interchange between data in XML and relational format
  - Map XML data elements to tables/columns in DB2
  - Reconstruct XML document from DB2 data



```
<address>
<name>
<title>Mrs.</title>
<first-name>Mary</first-name>
<last-name>McGoon</last-name>
</name>
<street>1401 Main Street</street>
<city>Anytown</city>
<state>NC</state>
<zipcode>34829</zipcode>
...
</address>
```

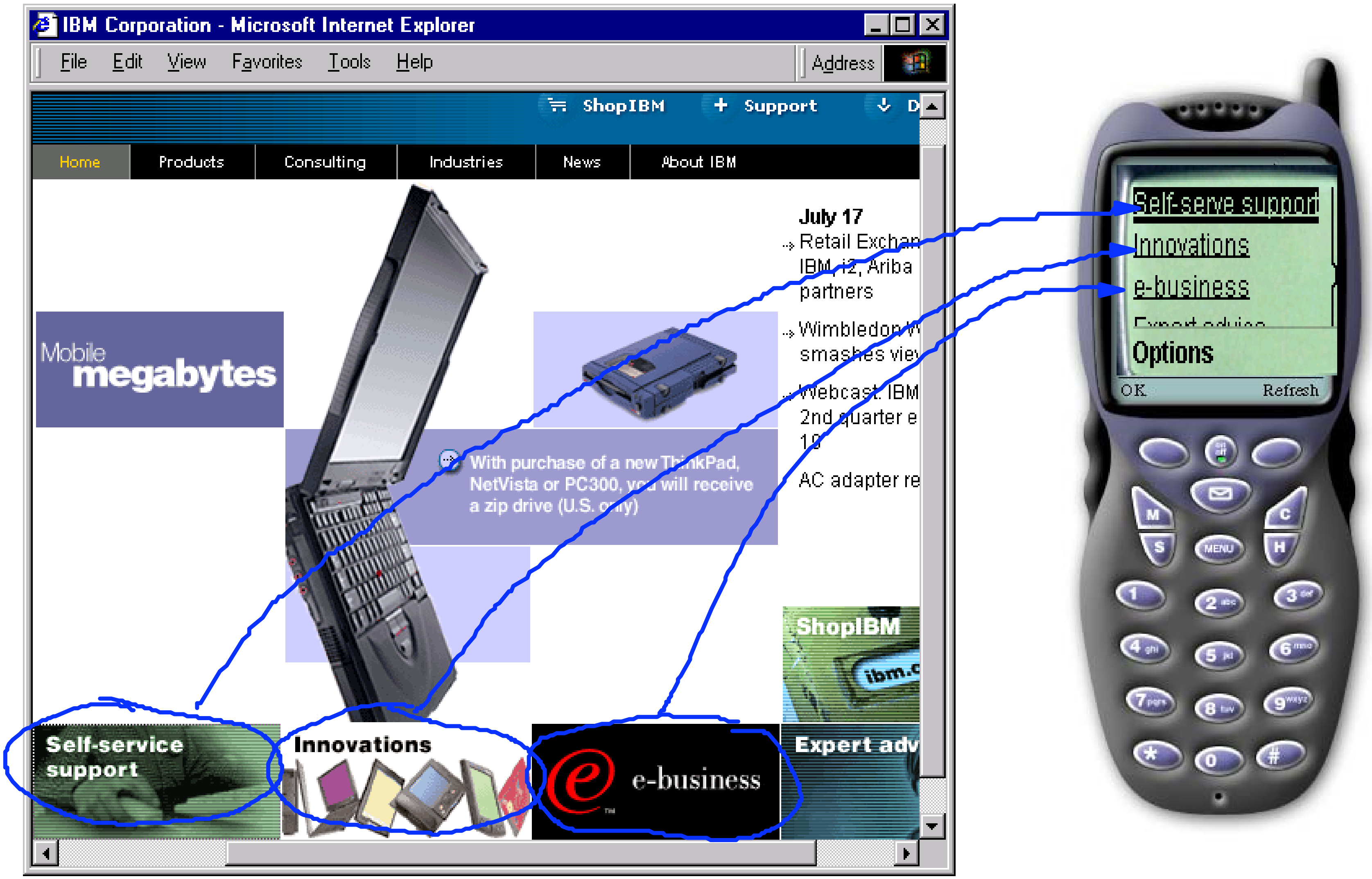
title	fname	lname

addr	city	state

DB2 UDB XML Extender provides

- new data types that let you store XML documents in DB2 UDB databases
- new functions that assist you in working with these structured documents

WebSphere Transcoding Publisher



July 17

Retail Exchan IBM, i2, Ariba partners

Wimbledon W smashes viev

Webcast: IBM 2nd quarter e 19

AC adapter re

With purchase of a new ThinkPad, NetVista or PC300, you will receive a zip drive (U.S. only)

WebSphere Transcoding Publisher

Bridges Markup Languages, e.g.,

- HTML → WML
- HTML → HDML
- HTML → iMode
- XML → XML via XSL stylesheets

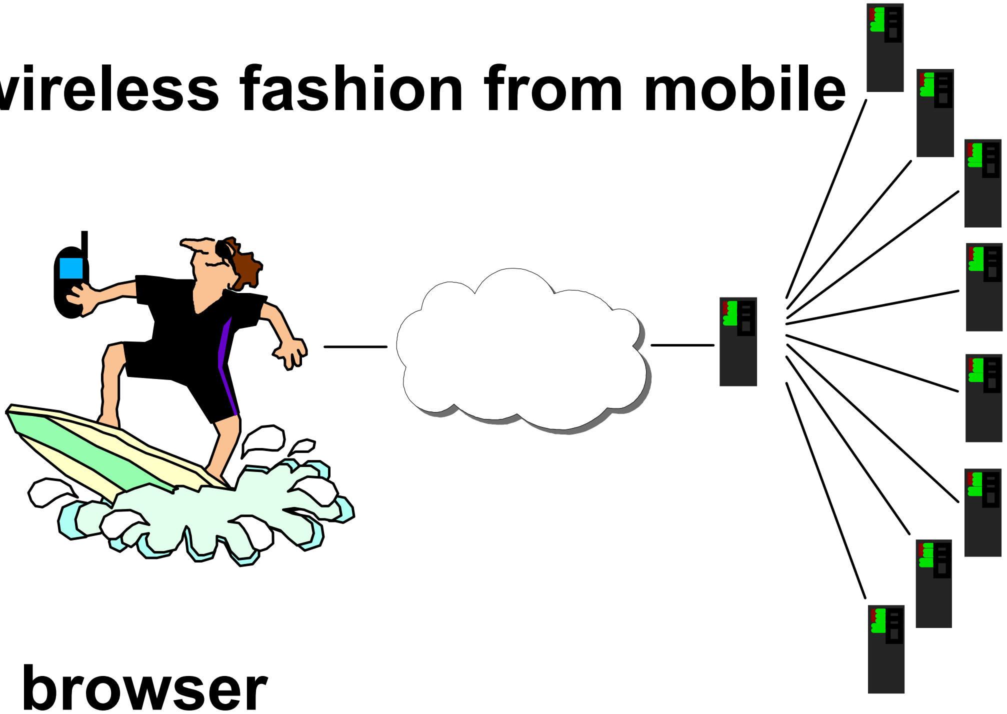
Converts Image Formats, e.g.,

- JPEG → GIF
- GIF → JPEG
- JPEG → WBMP
- GIF → WBMP

Management Central - Pervasive

Manage multiple systems in a wireless fashion from mobile devices

- PDA
- Cellular Phone (WAP)



Also runs in a workstation Web browser

Provides a subset of the Management Central capability

V4R5 availability

Additional capability in V5R1

Management Central Pervasive

V4R5

Watch iSeries System Status

Monitor Performance

V5R1

Watch iSeries System Status

Monitor Performance

Monitor Specific Jobs or Servers

Monitor Message Queues

Control Jobs

Execute Commands

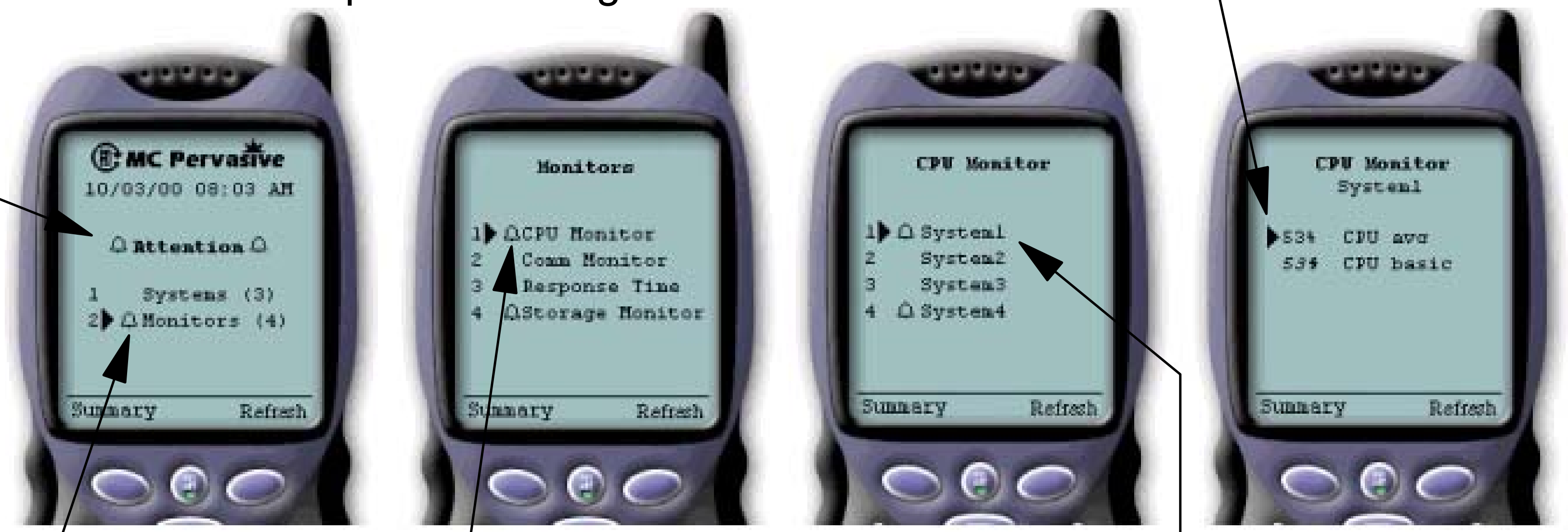
Manage Integrated xSeries Servers

Support for Additional Phone Devices

Management Central - Pervasive

Arrow indicates the item selected for drill down

Immediate indicator of a problem using the bell



CPU Monitor appears to need attention.

Select 'CPU Monitor' to drill down and see the system list

System1 is where the threshold occurred.

Select 'System1' to see the specific metric information such as CPU avg

Monitors appear to need attention.

Select 'Monitors' to drill down and see the monitor list

IBM Connect for iSeries 5733-B2B

A B2B Software Integration Framework

- Integrates your business applications with those of your trading partners
- Supports cXML, mXML and other trading partner protocols
- Provides connectivity to existing back-end applications

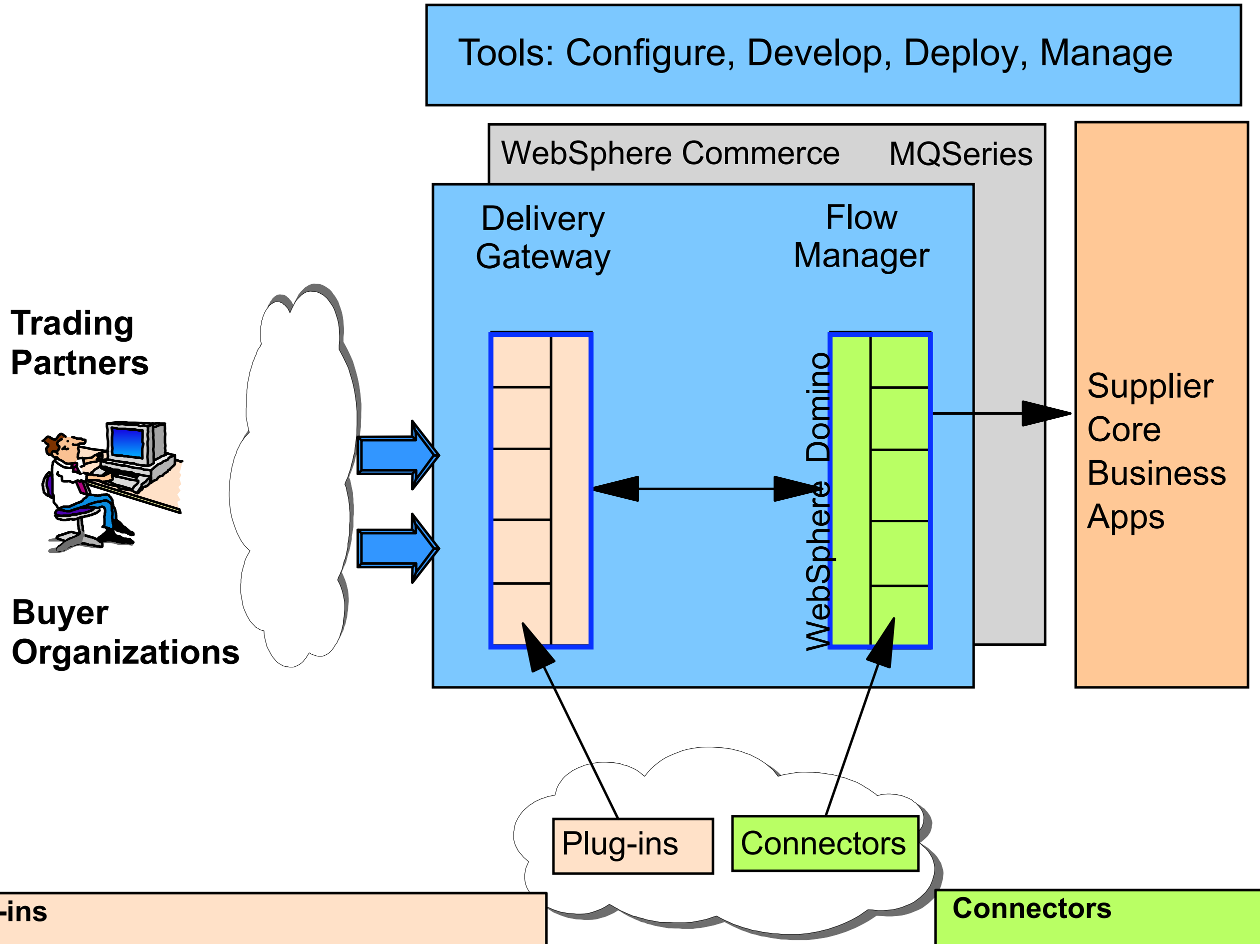
Targeted for the sell side of B2B

- Buyer/Supplier Registration tools
- Catalog Services
- Integration with WebSphere Commerce Suite

Focused on Small to Mid Market requirements

- Low cost
- Easy to deploy with minimal service costs
- Integrated tool set

Connect for iSeries Overview



Plug-ins

Release 1.0 (2/2/01)
--Ariba & Metiom Marketplaces

Release 1.1 (08/31/01)
--Additional e-Marketplace Connections
--Customized third party protocols

UPDATE

Connectors

Release 1.0 (2/2/01)
--MQSeries/DQ, PCML, Java

Release 1.1 (08/31/01)
--Additional data connector types
--Flow manager enhancements
(Multiple database access per request)

UPDATE

http://www.ibm.com/eserver/iseries/btob/connect

Notes:

Extending DB2 to support XML, allows storage and retrieval of XML documents making the information immediately usable.

XML and iSeries Summary

XML - a core technology for data oriented and web-based applications.

XML Strategic instrument for defining corporate data across application domains

- platform, vendor and language neutral
- open and create own meaningful tag set

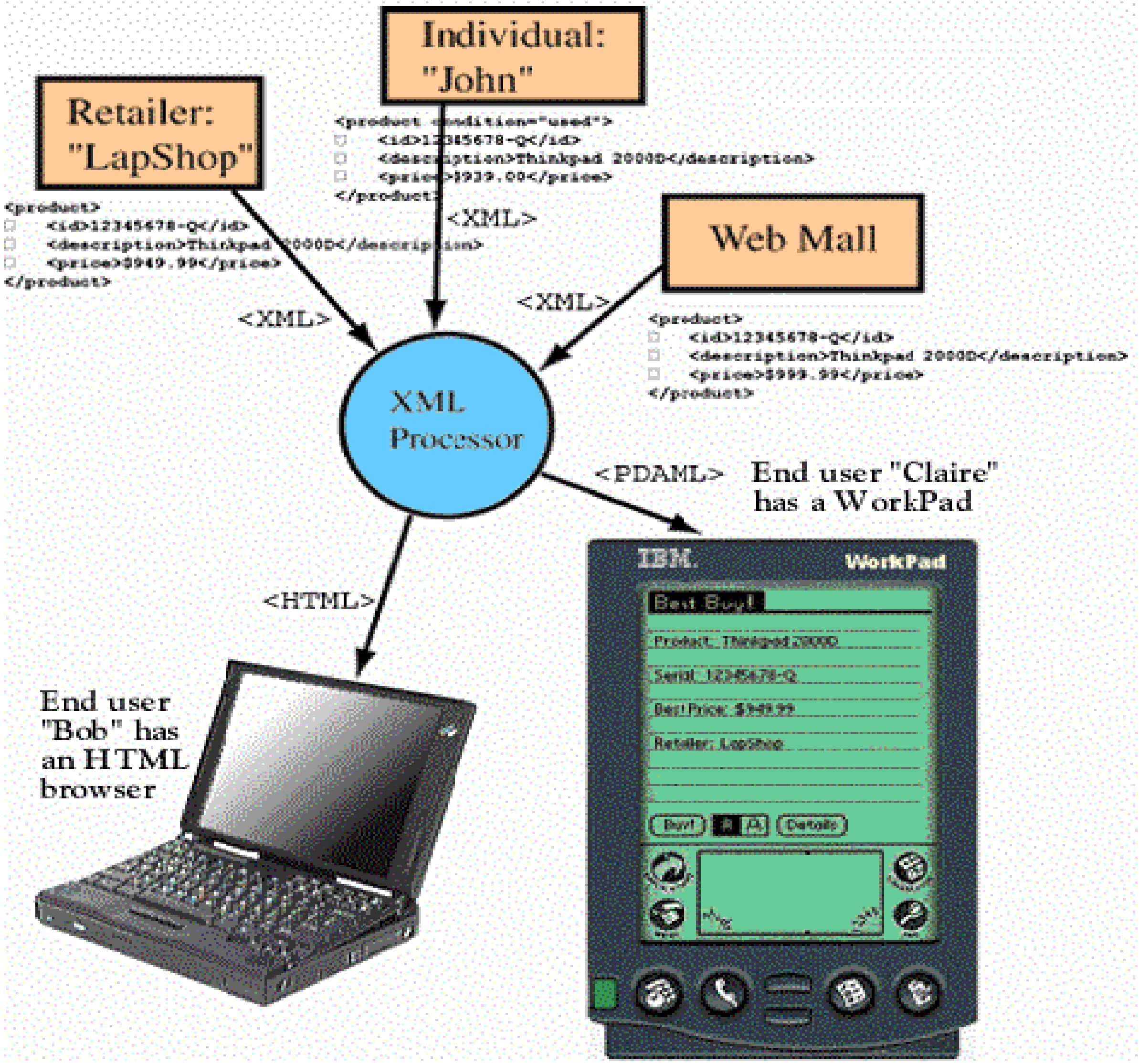
Core support to start using XML is available on iSeries

- WebSphere® Application Server
- IBM Toolbox for Java.

Notes:

IBM and iSeries are committed to support XML through currency with evolving standards and enabling use of XML through a variety of IBM software and middleware offerings.

XML - Connecting all the parts



```
<product condition="used">
□    <id>12345678-Q</id>
□    <description>Thinkpad 2000D</description>
□    <price>$939.00</price>
</product>
```

```
<product>
□    <id>12345678-Q</id>
□    <description>Thinkpad 2000D</description>
□    <price>$949.99</price>
</product>
```

```
<product>
□    <id>12345678-Q</id>
□    <description>Thinkpad 2000D</description>
□    <price>$999.99</price>
</product>
```

Notes:

This picture is a summary of all the components required for an XML implementation... XML, a parser, XSL, stylesheets, DTDs and the device protocols to use... But once implemented, this picture shows the tremendous flexibility available with XML in reusing data created once for a variety of users in a variety of formats.

Additional Information - web resources

IBM alphaWorks:

- Latest tools and enablers supporting XML
- http://www.alphaWorks.ibm.com/

IBM Toolbox for Java:

- Additional information on PDML and PCML
- http://www.iSeries.ibm.com/toolbox

W3C - XML standards and specifications:

- Status and detail on various XML-related standards
- http://www.w3.org/XML/

IBM - developer education:

- General Information and tutorials
- http://www.ibm.com/developer/XML/

IBM iSeries & XML

- http://www.iseries.ibm.com/developer/java/xml/index.html

Notes:

alphaWorks is the IBM site working with the latest IBM technologies. XML and the related enablers are all discussed on this page and you can keep track of the support provided by IBM.

The IBM Toolbox for Java page delivers the latest information specifically for iSeries.

The W3C pages let you in on the latest developments of the XML and related enabler specifications.

For more on IBM's XML education offerings, check out the tutorials and support information available on the IBM XML page.

Additional Information - web resources

OASIS - Organization for the Advancement of Structured Information Standards:

- Standards for defining data documents exchanged between businesses
- Mission: to simplify business information exchange
- http://www.oasis-open.org/

XML/EDI

- A grass roots advocacy for use of XML and EDI together
- http://www.xmledi.com/

Open Applications Group:

- Promote the easy and cost-effective integration of key business application software components
- http://www.openapplications.org

XML.org:

- Information on XML standards, tools & developments in XML
- Repository for industry standard DTDs that may be shared across companies
- http://www.xml.org/

Notes:

The XML.org page is one that shows progress in definition of industry standard DTDs. This is an excellent page to catch up on how others in a specific industry are using XML by their DTDs or just see how others are using XML if your specific industry is not yet represented.

XML and iSeries Summary

XML and Java Developing Web Applications

- Hiroshi Maruyama, Kent Tamura, Naohiko Uramoto
- ISBN 0-201-485435

XML Bible

- Elliotte Rusty Harold
- ISBN 0-7645-3236-7

Project Cool - Guide to XML for Web Designers

- Teresa A Martin
- ISBN 0-471-34401

The XML Files: Using XML and XSL with IBM WebSphere® 3.0

- IBM Redbook
- SG24-5479-00

Notes:

Finally, here are some books that have been helpful in our understanding of XML. There certainly are others and you should check them to make sure they will meet your needs...