XJ: Easy and Efficient XML Processing in the Java™ Programming Language

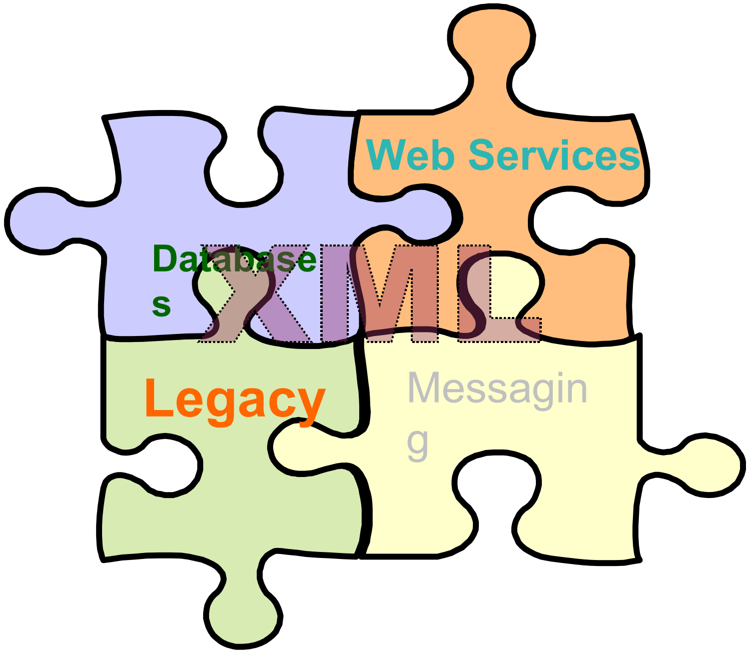
Mukund Raghavachari

Research Staff Member
IBM Research

http://www.research.ibm.com/xj

TS-8339

XML Is Pervasive



Programming is tedious, error-prone, and inefficient.

How can we make it better?

DOM

- Verbose
- No static schema type checking
- Reflection-like programming style
- Performance an issue

```
Element recipe = new Element(“recipe");
Element ingredient = new Element(“ingredient");
recipe.appendChild(recipe);
```

Java Architecture for XML Binding (JAXB)

- Converts XML Schema types into Java-based classes
  - Type system cannot express schema content models
  - Difference between attributes and elements
  - Ordering
- Some static type checking
- XPath evaluation by passing strings
  - No static type checking or optimization

XSLT and XQuery

- Domain-specific languages for processing XML
- Certain programming tasks may be complex
- Integration with Java-based programs is awkward
  - Java-based programs must pass and process XML data to these languages using runtime APIs:
    - DOM
    - XQJ

XJ Design Goals

- Make XML a first-class construct
  1. Consistency with XML standards (Namespaces, XML Schema, XPath)
  2. Minimal compiler extensions
  3. Efficiency
  4. Support both schema and schema-less processing
- XML processing applications are easier to write, more robust, and efficient

XJ Features

- Language extensions
  - Construction using XML literals
  - Integrated XPath
  - In-place XML updates
  - All of the above with or without XML Schema
    - Import of XML Schemas
- Efficient implementation
  - Native XPath compilation support
- Eclipse plug-in
  - To be released soon

Sample XJ Program

Refer to schema declarations

Inline XPath

XML Construction

```
import com.amazon.devheavy.*;
import com.amazon.devheavy.ProductInfo.Details;
import myOutput.*;
public Book checkReviews (ProductInfo pi,  String title, String author, float goodRating) {
    Sequence<Details>  bookSeq = pi[| .//Details[ProductName = $title] |];
    for (Details book : bookSeq) {
            if (!book[| [$author = .//Author] |].isEmpty())  {
                double discountMult = 0.5;
                if (!book[| [.//AvgCustomerRating >  $goodRating] |].isEmpty() )
                    discountMult = 0.75;
                 Book result = new Book(<Book name={title} >
                                        <isbn> {book[| Isbn |] } </isbn>
                                                 <price> {book[| .//Price |] *
       discountMult} </price>                       </Book>);
                return result;
}}}
```

XPath Navigation

context [| query |]

- The primary means of navigating XML data
- Semantics as defined by the XPath 1.0 standard

```
ProductInfo p = ….;
/* Generics for XML types */
Sequence<Details> items;
items = p[|.//Details[.//AvgCustomerRating > 3.5] |];
UsedPrice price1 = p [| UsedPrice |];
String price2 = p [| UsedPrice |];   /* Automatic unboxing as in Java 5*/
```

Compiler checks that XPath is well formed and valid according to the schema.

Construction of XML

- Construction can be specified as:
  - Inline XML (can use xmlns and xsi:type)
    - Can construct “untyped” XML as well
  - InputStream constructor

```
int price = 54;
XMLElement b = new XMLElement(
            <Book year="2004">
              <title>C# Concisely</title>
                <author>
                   <first>Judith</first><last>Bishop</last>
                </author>
                <author>
                   <first>Nigel</first><last>Horspool</last>
                </author>
                <publisher>Addison-Wesley</publisher>
                <price> {price} </price>
            </Book>);
```

XML Updates

- Value update

```
book[|./price|] = 100; or book[|
./price|].updateValue(100);
```

- Insertion (can operate on a sequence)

```
book.insertChild(author);
```

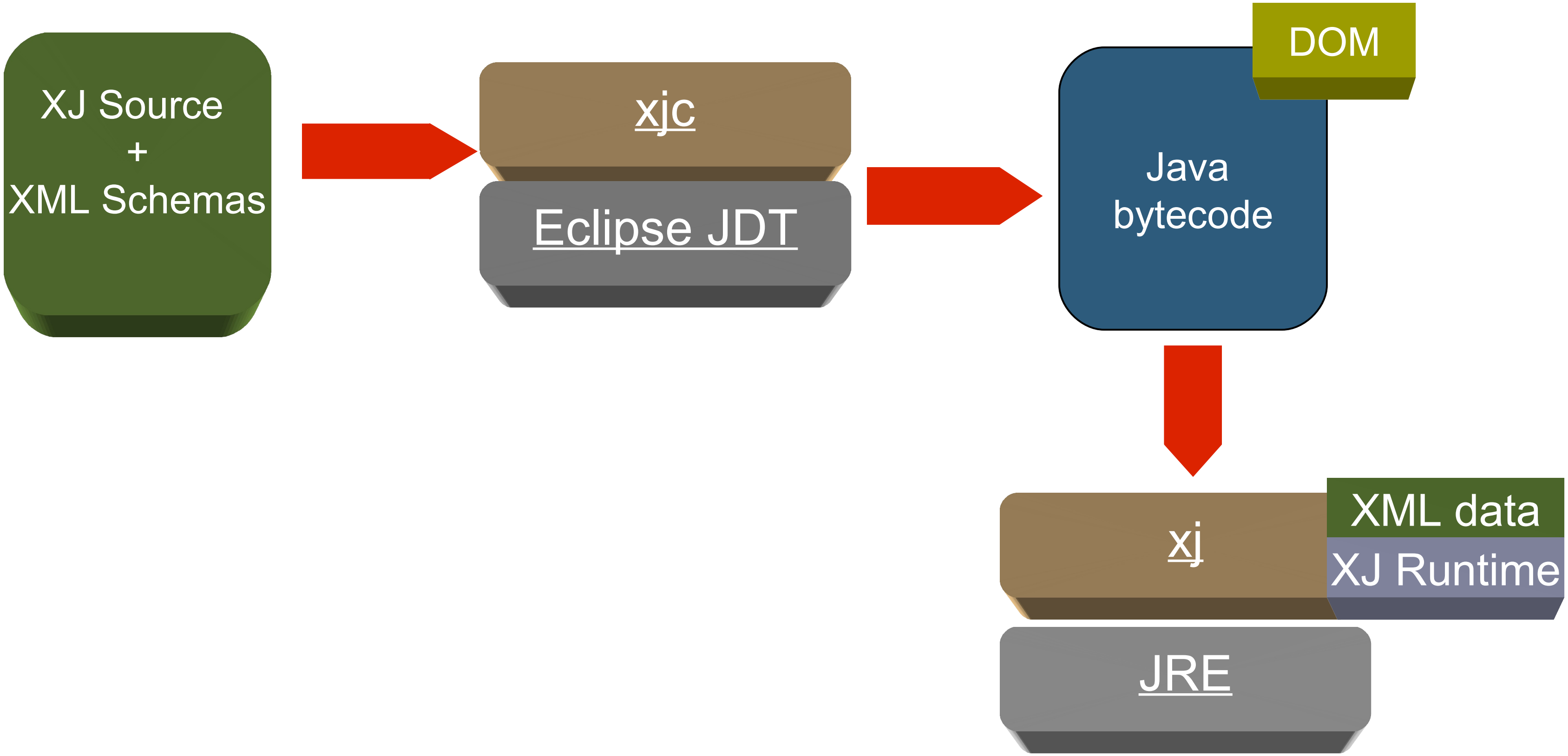
- Deletion (can operate on a sequence)

```
author.detachNode();
```

- Compound update

```
article.rename(“inproceedings”);
```

XJ Compiler and Runtime System



XJ Source + XML Schemas

xjc

Eclipse JDT

DOM

Java bytecode

xj

XML data

XJ Runtime

JRE

Demo Details

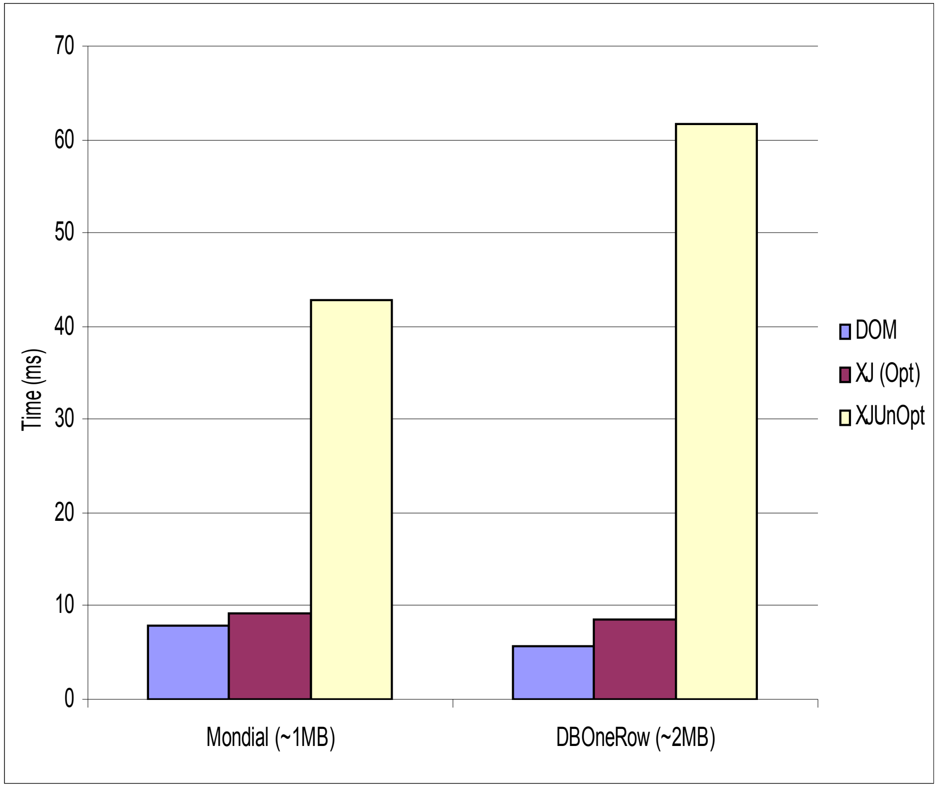
Based on “Recipes” Example From XML Tutorial

1. Convert list of recipes to HTML
   - Originally XSLT (construction, XPath)
2. Consolidate shopping list
   - XPath + updates

DEMO

Performance

- Using the DOM API to evaluate XPath expressions at runtime is inefficient
- Solution: Compile expressions directly to accesses on DOM tree



Advantages of XJ

- W3C standards compliance
- Ease of programming
  - Seamless integration with Java
  - Robustness: more errors discovered statically
  - Eclipse plug-in coming soon…
- Easier maintenance
  - Adapt to changes in XML Schemas
  - Independent of the runtime data representation
- Optimizations
  - Compiler rewritings

Links

- Web site: http://www.research.ibm.com/xj/
- Available for *free* download on *alphaWorks*
  - http://www.alphaworks.ibm.com/tech/xj/
- E-mail: *xj@watson.ibm.com*

Q&A

the POWER of JAVA™

XJ: Easy and Efficient XML Processing in the Java™ Programming Language

Mukund Raghavachari

Research Staff Member
IBM Research

http://www.research.ibm.com/xj

TS-8339