Introduction to JiBX

A Java Data Binding Tool

David Lucek
dave@lucek.com
www.lucek.com
August 4th, 2005

What is Data Binding

- A Data Binding library converts Java objects to and from XML streams
- Converting to/from XML with existing parsers is time consuming and usually not your core competency
- It allows the application to work with an object model and automatically have XML serialization and deserialization
- Commonly used in SOAP services or any interface that uses XML

Data Binding Definitions

- Marshalling – Serializing Object to a XML stream
- Unmarshaling – De-serializing an XML stream to an object
- Binding – A mechanism to associate objects with XML elements, usually a mapping
- XSD – XML Schema, describes structure of an XML document
- Generator – Takes an XML XSD and create an object model with binding support

Generation Approach to Data Binding

- Use a XSD Generator to create objects then provide either:
  - Helper classes for each object
  - Static methods for marshaling/unmarshaling
- One of the first approaches to data binding
- Pros
  - Easy to create your object model quickly
- Cons
  - The generated model can be bulky and hard to use.
  - Forced to use a model possibly not suited to your domain

Mapping Approach to Data Binding

- Use Existing Object Model and mapping mechanism
  - The mapping mechanism defines the relationships between objects and XML elements.
- Pros
  - Can work with existing models
  - Do not have to learn a new model
- Cons
  - More initial work to create the mappings

Common Data Binding Frameworks

- JAXB 1.0 / JAXB 2.0
  - One of the first frameworks from Sun
  - Uses Generation
  - Supports almost all XSD syntax
  - Easiest to get up and running quickly
  - http://java.sun.com/xml/jaxb
- Castor
  - One of the first open source frameworks
  - Uses Generation
  - Generated model is heavy
  - http://castor.codehaus.org

Common Data Binding Frameworks

- XMLBeans
  - Formally from BEA, now Apache
  - Uses Generation
  - Becoming more popular
  - http://xmlbeans.apache.org
- JiBX
  - http://jibx.sourceforge.net
  - Uses mapping approach
  - About 2 years old
  - Fast

Why JiBX?

- Its fast
- Can your existing object model
- Using generated models is usually painful and forces you to copy to/from your object model
- High degree of control and customization
- Every time the schema changes you need to re-generate, with JiBX just change the mapping file

Good Features of JiBX

- Almost complete control of translation process
  - Moving data from attribute to element mappings is trivial
- Custom serialize rs/deserializers
- Custom pre-get and pre-set marshaller/unmarshaler methods
- XSD schema changes => just change the binding file. Very fast for schema updates
- Re-usable abstract bindings
- etc..

Performance of JiBX

- Look at https://bindmark.dev.java.net/
  - Compares the different frameworks
- JiBX is fastest or 2nd fastest for marshaling/unmarshaling among the most common open source frameworks
- Also has small memory foot print
- JiBX uses XML Pull Parser technology with Byte Code generation

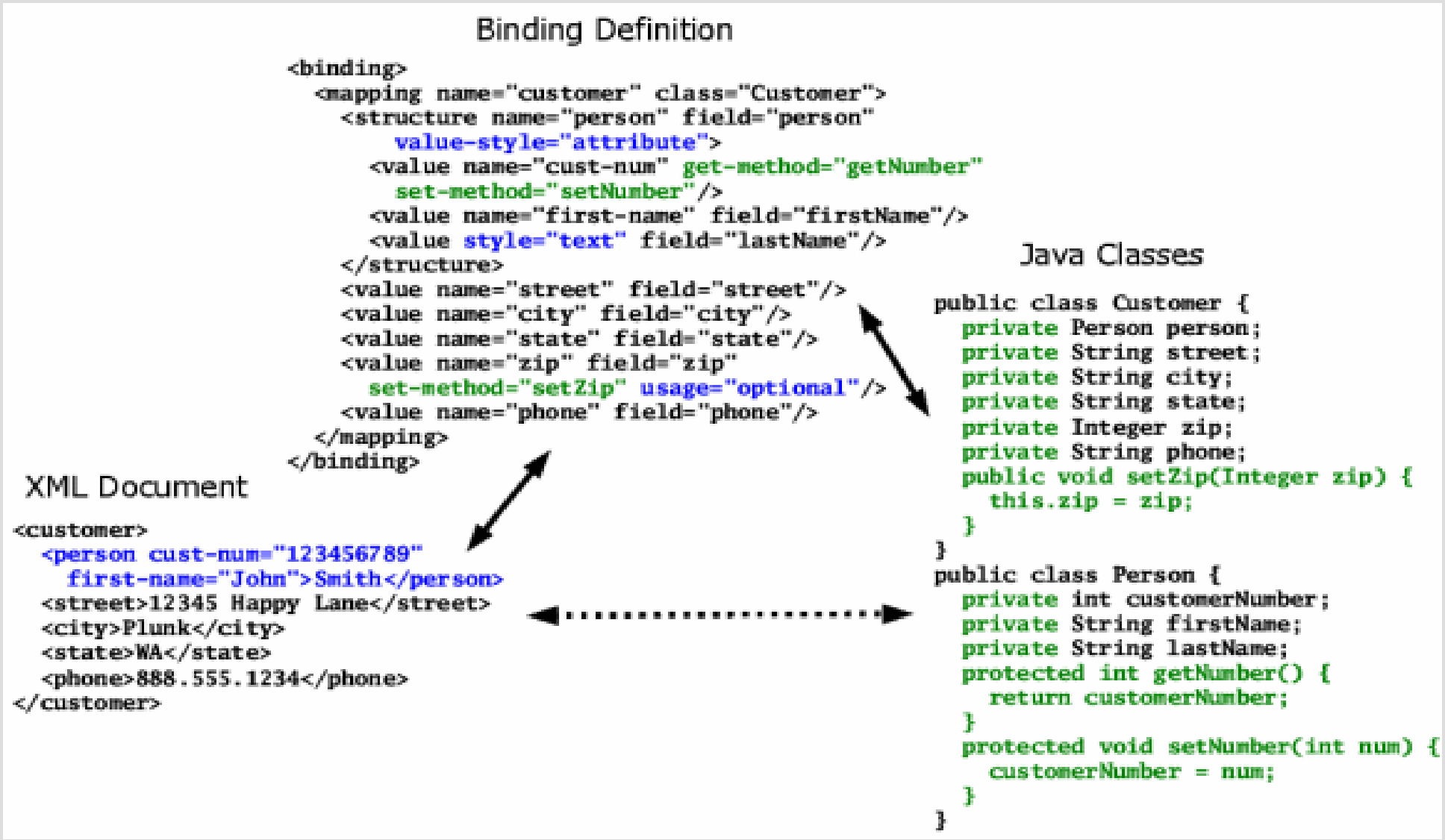
Using JiBX

- Download the latest code, 1.0 RC1
- Put jars into your classpath
- Create mapping
  - Manually Generate
  - Or use Xsd2Jibx to generate mapping file from a object hierarchy
- Run classes and binding file through compiler. Use Ant task
- At run time
  - Create binding factory and marshaling/unmarshaling context
  - Marshall/UnMarshal using context

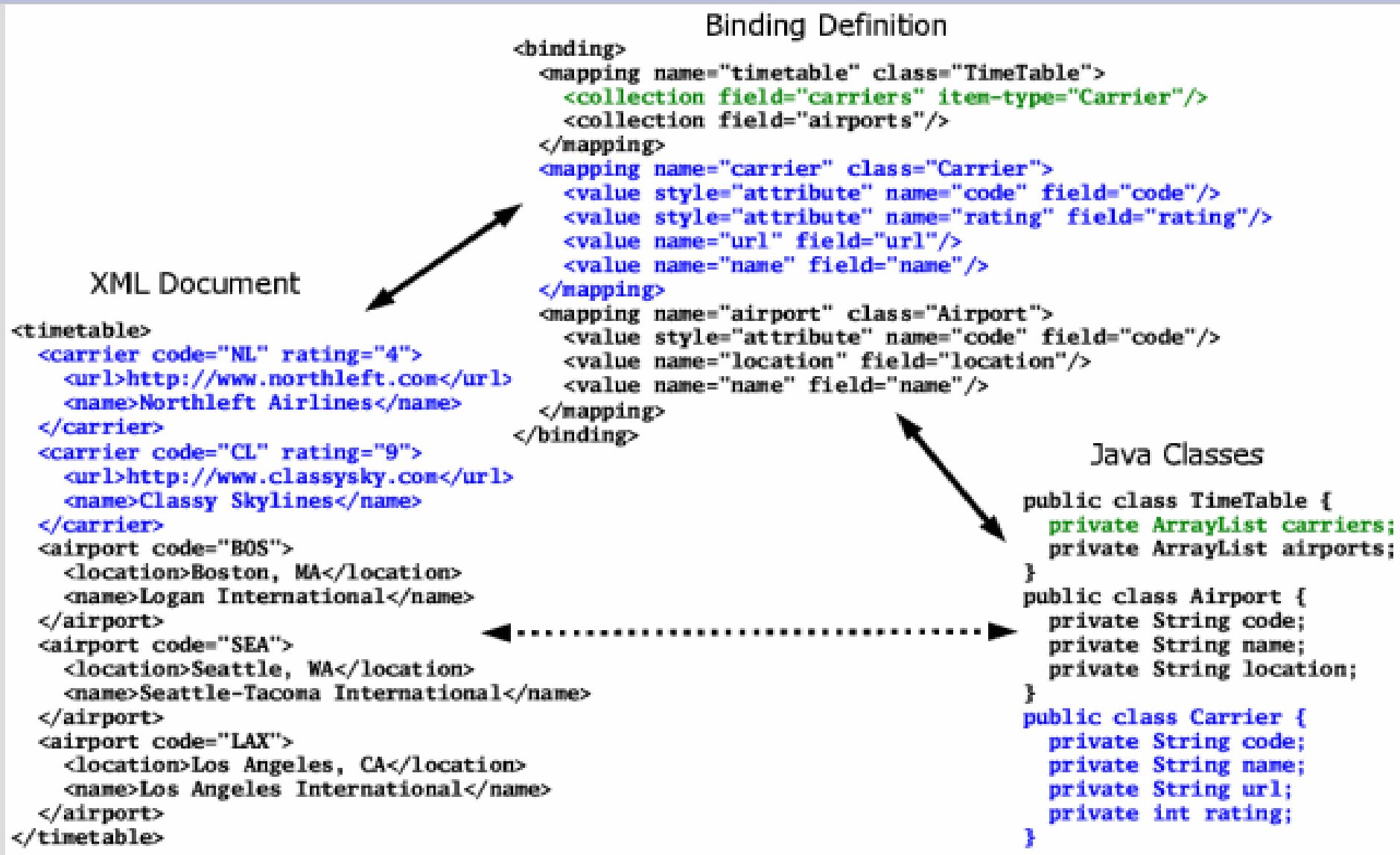
Common Initial Setup

- Use the xsd2Jibx tool to get initial mapping
  - However, the existing tool is flakey
  - It does not support xs:unions, xs:any and other XSD syntax
  - Use it as a starting point
- Manually change/support the mapping file from that point on
- Use Ant task for builds
- There is also an Eclipse plug-in

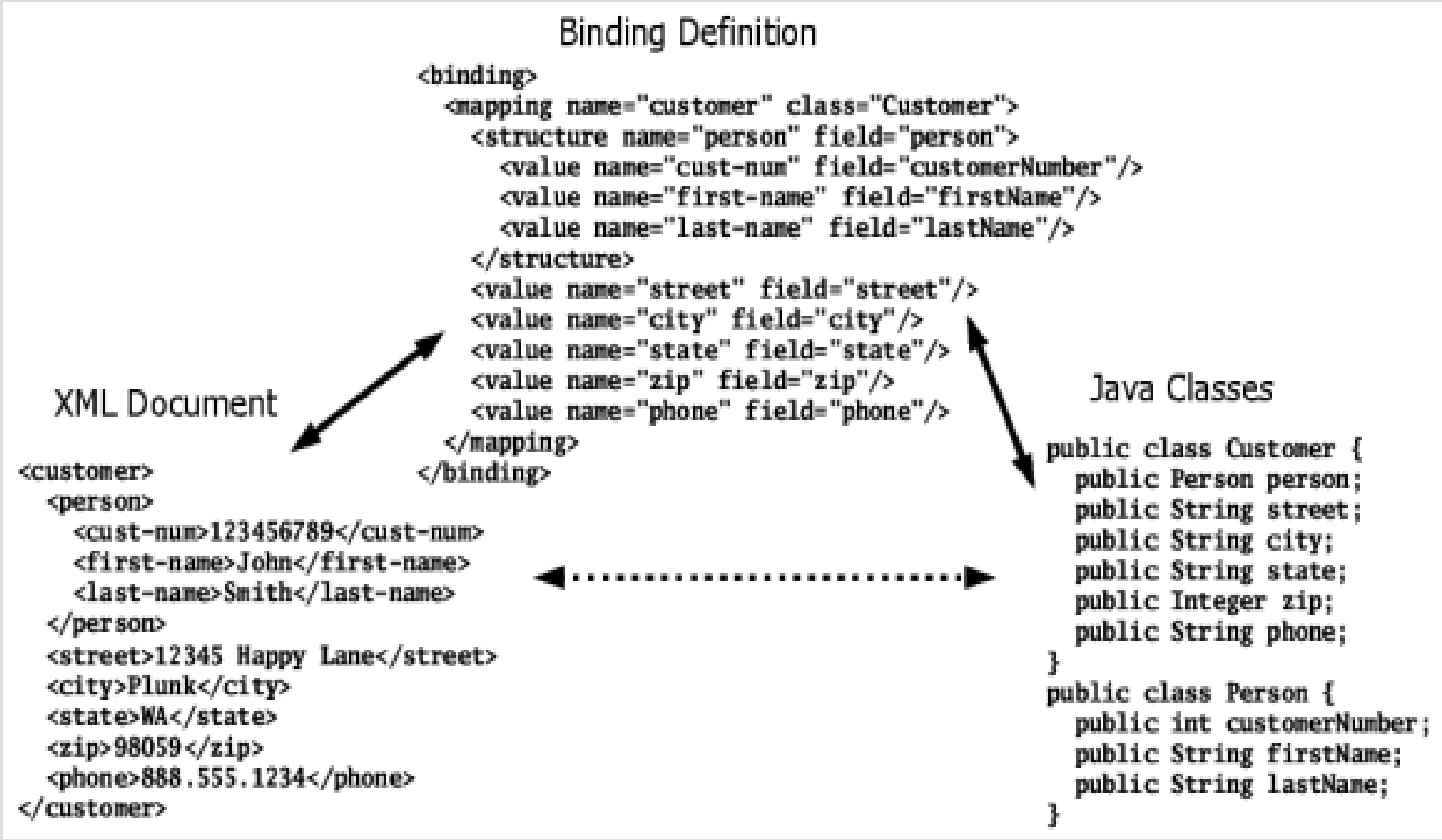
Common Mapping



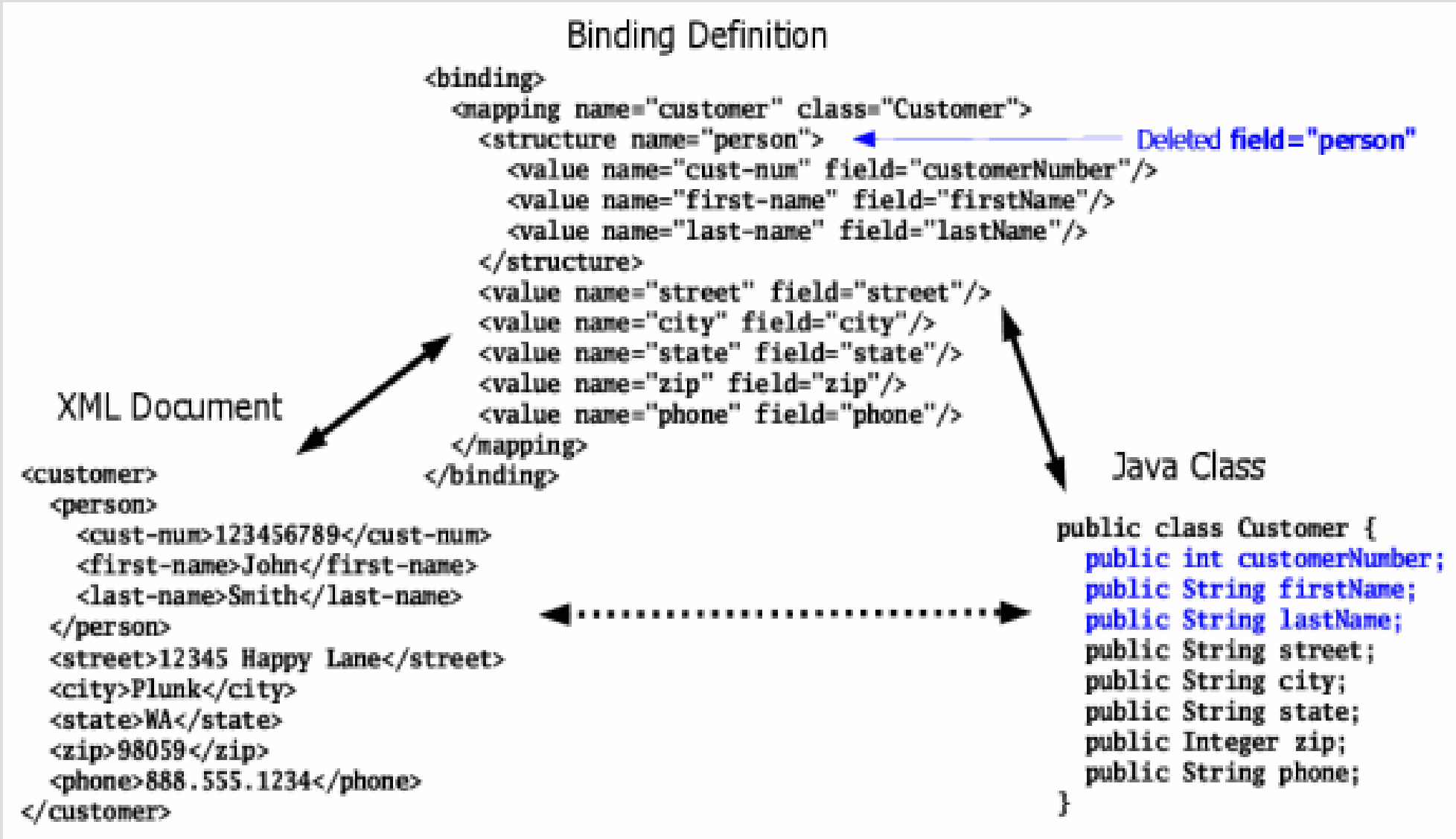
Collection Mapping



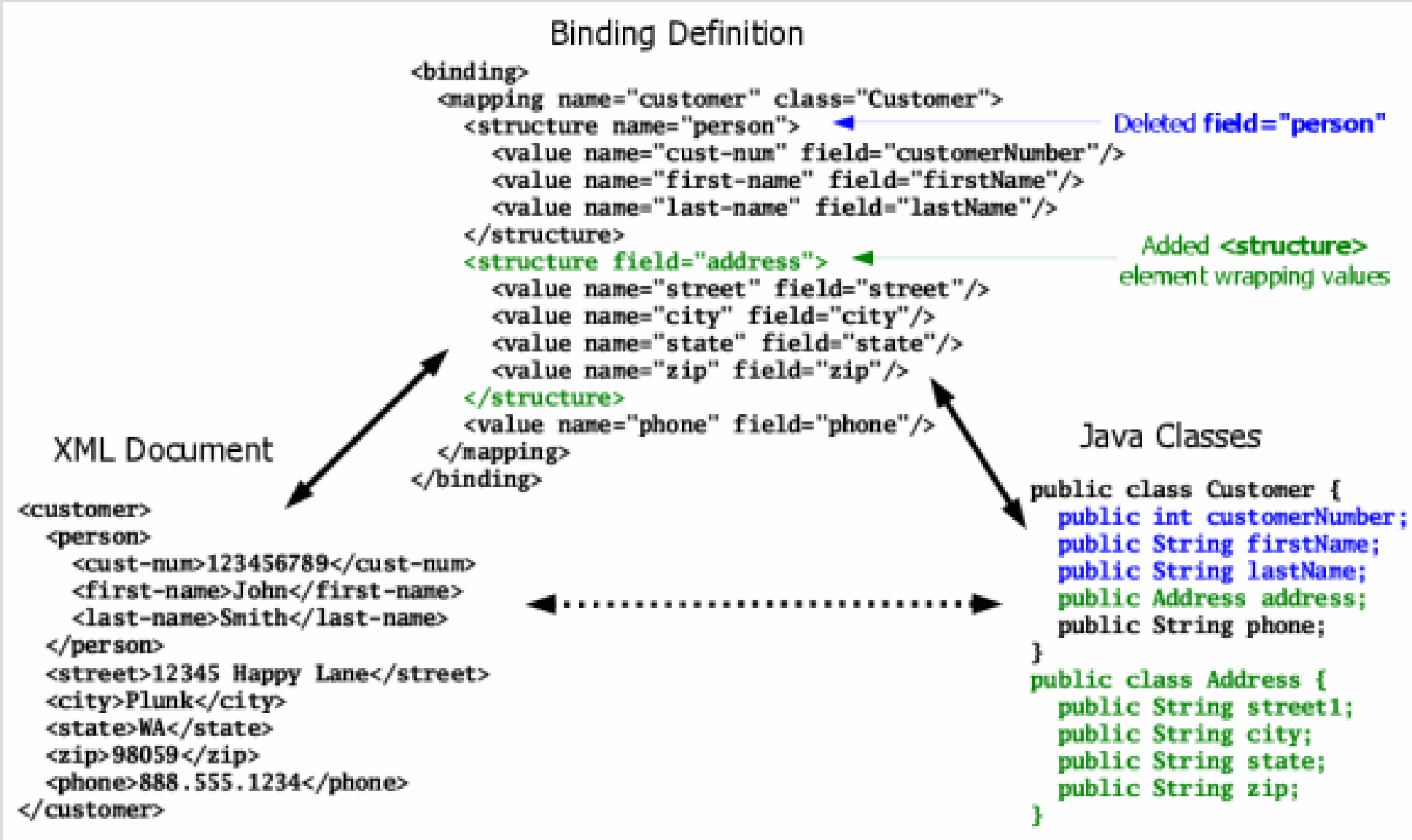
Structure Mapping 1



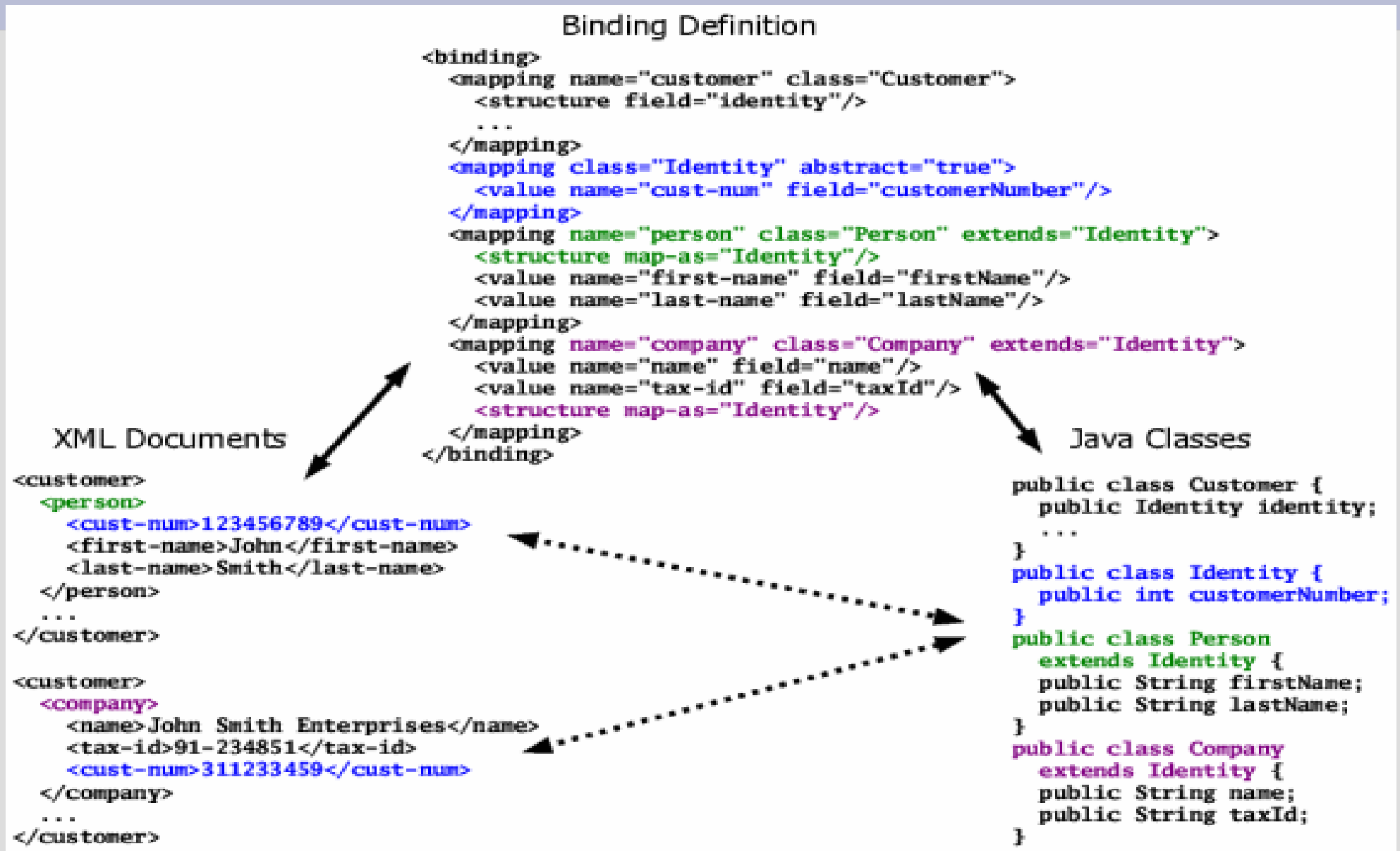
Structure Mapping 2



Structure Mapping 3



Abstract Mapping



SOAP Services

- JiBXSoap is a separate project
- Provides a deployable WAR for SOAP services
- Document/Lit Support Only
- Maps Top Level XML element to a method in a handler classs
- Configurable via a descriptor
- Soap Fault handling is a little flaky
- Its a fast and simple architecture
- Example

SOAP Services

- Performs about 2-3 times as as Axis for doc/lit
- Use a simple Servlet for the Soap Protocol handling
- Servlet Container Config

```
<web-app>

  <servlet>
    <servlet-name>soap_servlet</servlet-name>
    <servlet-class>org.jibx.soap.server.SOAPServlet</servlet-class>
    <init-param>
      <param-name>quake-service</param-name>
      <param-value>quake-service.xml</param-value>
    </init-param>
  </servlet>

    <servlet-mapping>
    <servlet-name>soap_servlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>

</web-app>
```

SOAP Services

- Service Configuration

```
<service name="quake">
  <schema>SeismicSchema.xsd</schema>
  <wsdl-uri>http://seismic.sosnoski.com/wsdl</wsdl-uri>
  <handler-
    class>com.sosnoski.seismic.jibxsoap.QuakeBase</handler-
    class>
  <operation method="process"/>
</service>
```

Demo and Wrap Up

- Demo
- JiBX is great when
  - You need to use an existing model
  - Or want to work in a specific model
  - Need very specific control of the translation process
  - Need Speed
- When JiBX might not be best
  - Want to get up and running fast
  - Not worried about performance