

# The World's Fastest Web Services

Dennis M. Sosnoski

The JiBX Project  
<http://www.jibx.org>

# Outline

- JiBX background
- JiBX 1.2 code-first support
  - Schema generation
  - WSDL generation
- JiBX web services
  - Axis2 and JiBX/WS implementations
  - Performance comparison
- JiBX 1.2 schema-first support

# Introduction to JiBX

- Project started in 2003
- First production release end of 2005
- JiBX binding definitions specify conversions
  - Binding compiler adds bytecode to implement
  - Added bytecode calls JiBX runtime methods
- 1.2 release (now in beta) adds major features:
  - Flexible schema/WSDL generation from code
  - Flexible code generation from schema
  - Modular binding compilation

# Code-first with JiBX

- Code-first approach often maligned
  - Exposes implementation, breaks loose coupling?
  - Generated schemas unusable for other purposes?
  - Loses benefits of SOA with different data version?
- Code-first done correctly has advantages
  - Much easier than working with schema tools
  - Allows reuse of existing code assets
- But requires a proper tool!

# Default schema generation

- JiBX BindGen tool demonstration
- Show basic data structure for example
- Generate schema directly from code
- Features:
  - “Smart” schema structure – embedded vs. global
  - Javadoc export to schema documentation

# Customizing code usage

- Defaults reasonable, but sometimes wrong
- Can use customizations structure to improve
- First customizations example:

```
<custom property-access="true">  
  <class name="org.jibx.starter1.Address" includes="street1 city state  
    postCode country" requires="street1 city"/>  
  <class name="org.jibx.starter1.Customer"  
    requires="firstName lastName customerNumber"/>  
  <class name="org.jibx.starter1.Item" excludes="description"  
    requires="id quantity price"/>  
  <class name="org.jibx.starter1.Order"  
    requires="orderNumber customer billTo shipping orderDate"/>  
</custom>
```

- Demonstrate customizations

# Customizing XML usage

- Controlling namespaces, element vs. attribute

```
<custom namespace-style="fixed"
  namespace="http://www.jibx.org/starters/fromcode" property-access="true">
  <package name="org.jibx.starter1">
    <class name="Address" includes="street1 city state postCode country"
      requires="street1 city"/>
    <class name="Customer" requires="lastName firstName /customerNumber"/>
    <class name="Item" excludes="description"
      requires="@id @quantity price"/>
    <class name="Order"
      requires="orderNumber customer billTo shipping orderDate"/>
  </package>
</custom>
```

- Demonstrate

# Non-Java 5 usage

- Untyped collections and non-standard enums

```
<custom namespace-style="fixed" force-mapping="true" strip-prefixes="m_"
  namespace="http://www.jibx.org/starters/fromcode">
  <package name="org.jibx.starter2">
    <class name="Customer" requires="lastName firstName /customerNumber">
      <value field="m_middleNames" item-type="java.lang.String"/>
    </class>
    <class name="Item" excludes="description">
      <value get-method="getId" set-method="setId" attribute="id"
        required="true"/>
      <value property-name="quantity" attribute="quantity" required="true"/>
      <value field="m_price" required="true"/>
    </class>
    ...
    <class name="Shipping" form="simple"
      deserializer="org.jibx.starter2.Shipping.fromString"/>
  </package>
</custom>
```

- Demonstrate

# Code-first web services

- Some tools require only schema and binding
- Axis2-JiBX support needs WSDL
- Demonstrate WSDL generation from code
- Jibx2WsdI tool builds on BindGen tool
  - Same customizations for data structure schema
  - Added customizations for service definition
- Best way from code to WSDL, even if not using JiBX

# Apache Axis2

- Apache web services support since 2001
- Axis2 the latest generation
  - Core is flexible framework for SOAP infrastructure
  - Supports modular framework extensions
  - Supports pluggable data binding for XML payloads
- Currently at 1.4.1 release (1.5 in progress)

# Axis2 data binding

- Intended to allow pluggability
  - Originally shipped with XMLBeans included
  - “Axis Data Binding” (ADB) as built-in alternative
  - JiBX support implemented as of Axis2 1.0
  - JAXB 2.0 support added since
- Implemented via extensions to WSDL2Java
  - Takes input WSDL service description
  - Generates linkage code (and data model, for most frameworks) as output

# JiBX in Axis2

- WSDL2Java generates linkage code
  - Need to supply your own data classes and binding
  - Binding must define elements referenced in WSDL
  - Your code works with your own data classes
- Demonstrate code generation and options

# JiBX/WS

- JiBX/WS delivers fast web services
  - JiBX/WS uses JiBX data binding exclusively
  - Supports multiple transports (HTTP, TCP, etc.)
  - Multiple styles (SOAP, POX, REST)
  - Multiple encodings (text XML, XBIS, etc.)
- Working toward best ease of use
  - Services need only JiBX binding and deployment descriptor
  - Client has direct wrapped doc/lit support
  - Generation of unwrapped interfaces in future

# Status

- JiBX/WS based on earlier JibxSoap
  - Demonstration code from 2004, never productized
  - Performance about the same
  - JiBX/WS adds alternative transports, styles, and formats
- Nigel Charman co-lead on project
  - Working on REST support, documentation
  - Also making JiBX/WS Spring-friendly
- Beta release RSN

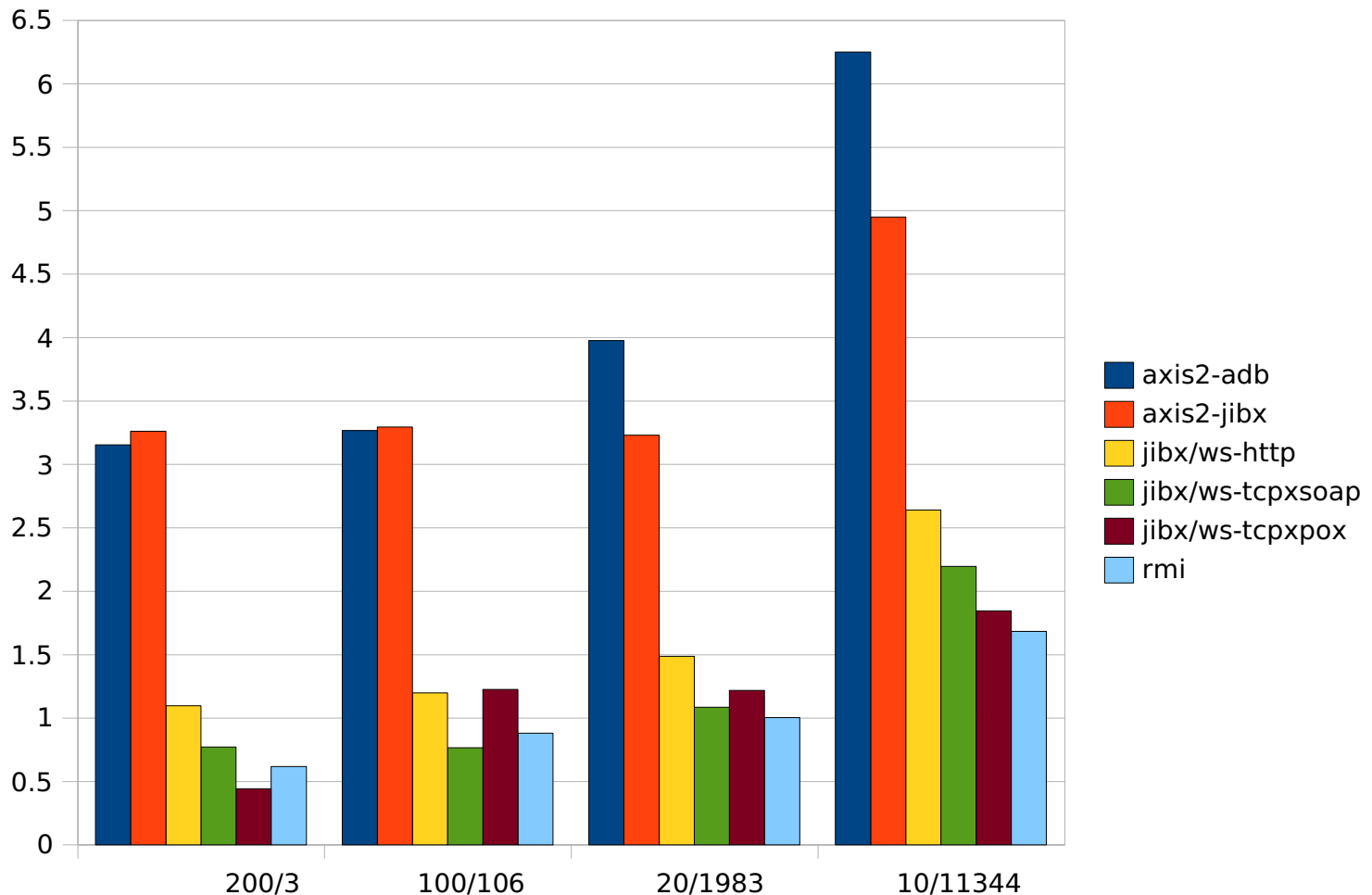
# Example application

- Earthquake information web service
  - Query by date, location, magnitude, etc.
  - Returns results sorted by area
  - Test allows scaling size of responses
- Variations tested:
  - Axis2, with both ADB and JiBX data binding
  - JiBX/WS, using SOAP over HTTP, and both SOAP and POX over TCP with XBIS (encoded XML)
  - RMI (binary, not XML)

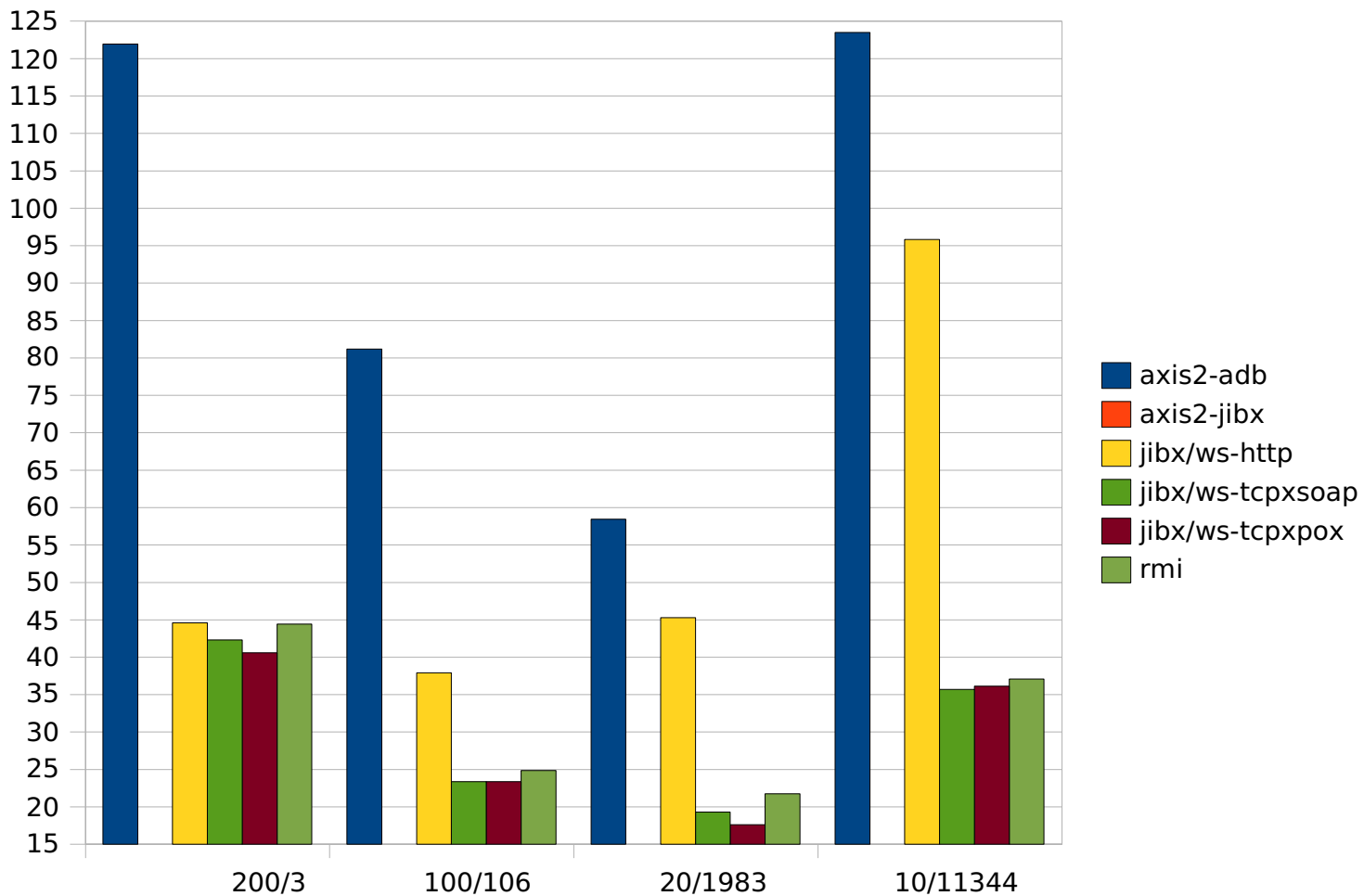
# Performance test

- Use pseudo-random sequence for queries
- Tune request ranges for different densities:
  - Very low – 200 queries returning 638 quakes
  - Medium – 100 queries returning 10638 quakes
  - High – 20 queries returning 39669 quakes
  - Very high – 10 queries returning 113442 quakes
- Verify number of quakes returned for each, etc.

# Local test



# Remote test



# Why the differences?

- Remote adds transport latency
  - Partially a fixed per-request cost (especially HTTP)
  - Partially dependent on message size (XBIS, RMI)
- JiBX data binding faster than ADB
- JiBX/WS framework much faster than Axis2
  - Less per-connection overhead even with HTTP
  - Much less overhead with TCP/IP
- XBIS encoding more compact than text XML

# Web services with JiBX

- Other frameworks also support JiBX binding:
  - Spring-WS
  - Apache CXF
- Performance about the same as Axis2
- Other data binding alternatives used with these frameworks also about the same performance

# Schema-first with JiBX

- Schema-first approach often needed
  - Client for existing web service
  - Industry-standard schema for message definitions
  - Architect group supplies WSDL and schema
- JiBX 1.2 provides very good support
- **Demonstrations**

# Schema-first features

- Strengths
  - Best data model for schema (choice, etc.)
  - Smart interpretation of schema (inlining, etc.)
  - Allows schema subsetting
- Weaknesses
  - Still no `xsi:type` runtime support
  - Missing some schema datatypes

# Questions?

Dennis Sosnoski – [dms@sosnoski.co.nz](mailto:dms@sosnoski.co.nz)

The JiBX Project  
<http://www.jibx.org>