

# Web Services - Overview

- Machine-to-Machine interaction over a network (Internet)
- Message oriented
- Public Interface
- Relatively easy to use

# Web Services - Terminology

- SOAP envelope – XML
- RPC – Remotely call a method
- HTTP(S) – Transport mechanism
- WSDL – Description of services provided
- UDDI – Discover available services

# Web Services - Resources

- <http://ws.apache.org/axis/index.html>
- <http://java.sun.com/webservices/>
- <http://java.sun.com/webservices/docs/1.6/tutorial/doc/index.html>
- <http://msdn.microsoft.com/webservices/>

# Web Services – eWiSACWIS

- County Cross-Reference (XRef) Update
  - Send county IDs for person, provider & case directly to eWiSACWIS
- Common Intake Agent (CIA)
  - Send intake details to eWiSACWIS
  - Replacement for current Java/MQ Common Intake Agent

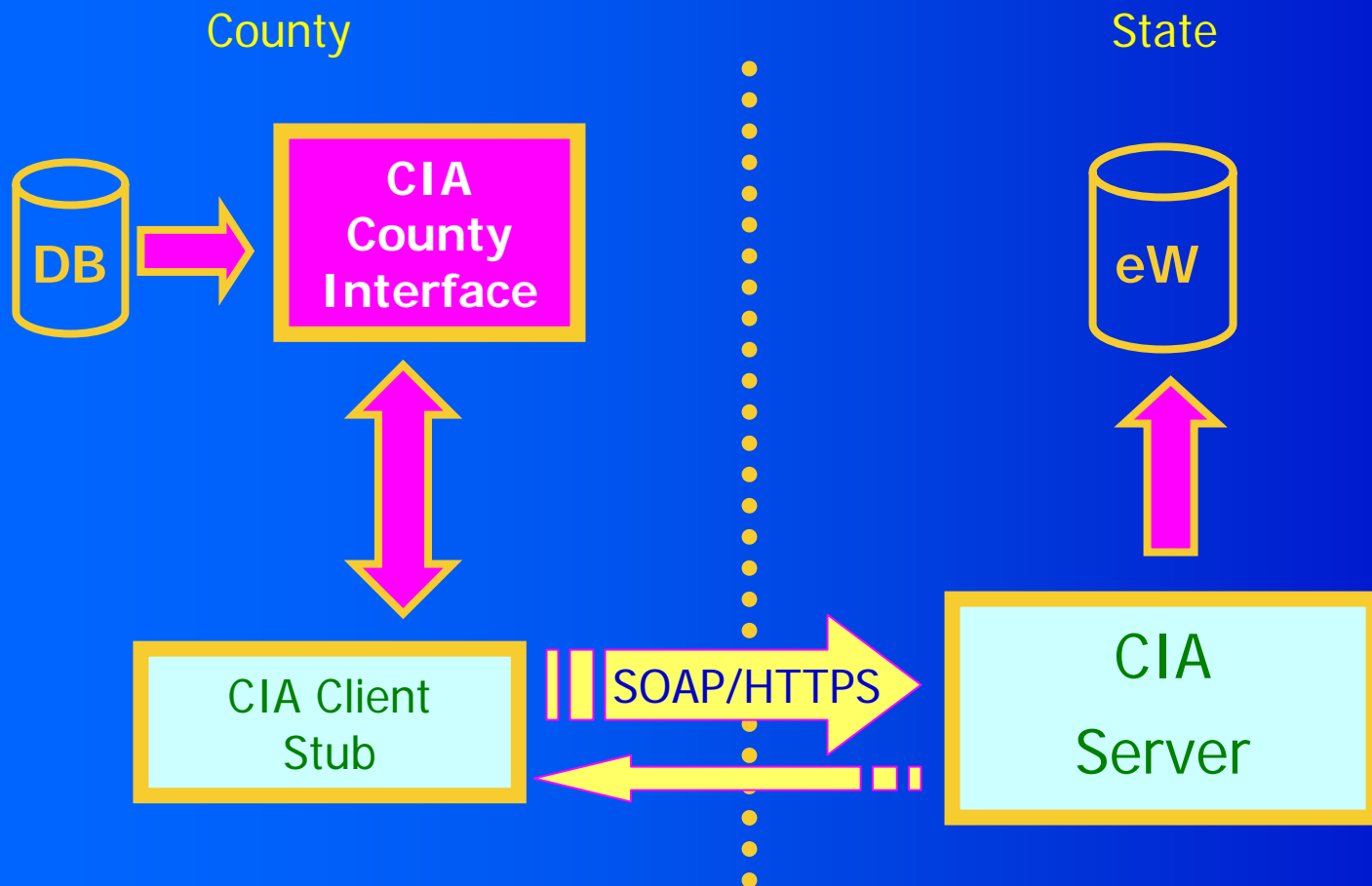
# Web Services – County XRef

- Short and simple...Try this first!
- County sends:
  - Entity Type (Person, Provider, Case)
  - County Entity ID - Alphanumeric
  - State (eWiSACWIS) ID #
- State performs:
  - Message Validation & ID Verification
  - Update DB tables
  - Reply with OK (or error message)

# Web Services – CIA

- Replacement for Java-based MQ client
- No need for VPN to State
- State builds:
  - Server process
  - Client stub (Use ours or build your own)
- County develops:
  - Extract county data elements
  - Interface to client stub

# Web Services CIA Components



# Web Services – WSDL

- WSDL - Public interface to WS

- What ?

Data types

- Where ?

State server

- How ?

Methods & operations

# Web Services – Data Types

```
<complexType name="Intake">
  <sequence>
    <element name="afterHoursFlag" nillable="true" type="xsd:string"/>
    <element name="countyCode" type="xsd:int"/>
    <element name="countyIntakeId" nillable="true" type="xsd:string"/>
    <element name="cpsAddress" nillable="true" type="tns2:Address"/>
    <element name="intakeMethodCode" type="xsd:int"/>
    <element name="intakeTypeCode" type="xsd:int"/>
    <element name="interpreterFlag" nillable="true" type="xsd:string"/>
    <element name="languageCode" type="xsd:int"/>
    <element name="packetReceivedDate" nillable="true" type="xsd:dateTime"/>
    <element maxOccurs="unbounded" name="participants" nillable="true" type="tns2:IntakeParticipant"/>
    <element name="policeNotifiedFlag" nillable="true" type="xsd:string"/>
    <element name="referralDate" nillable="true" type="xsd:dateTime"/>
    <element name="reportDueDate" nillable="true" type="xsd:dateTime"/>
    <element name="reporterDescriptionCode" type="xsd:int"/>
    <element name="assignedWorkerId" type="xsd:int"/>
    <element maxOccurs="unbounded" name="narratives" nillable="true" type="tns2:DocNarrative"/>
    <element name="createdByWorkerId" type="xsd:int"/>
    <element name="requestedServices" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
```

# Web Services – Address

```
<complexType name="Address">
  <sequence>
    <element name="addressLine2" nillable="true" type="xsd:string"/>
    <element name="addressLine3" nillable="true" type="xsd:string"/>
    <element name="apartment" nillable="true" type="xsd:string"/>
    <element name="countryCode" type="xsd:int"/>
    <element name="state" nillable="true" type="xsd:string"/>
    <element name="streetName" nillable="true" type="xsd:string"/>
    <element name="streetNumber" nillable="true" type="xsd:string"/>
    <element name="town" nillable="true" type="xsd:string"/>
    <element name="zip" nillable="true" type="xsd:string"/>
    <element name="alternativeExtension" nillable="true" type="xsd:string"/>
    <element name="alternativePhone" nillable="true" type="xsd:string"/>
    <element name="cell" nillable="true" type="xsd:string"/>
    <element name="cellExtension" nillable="true" type="xsd:string"/>
    <element name="email" nillable="true" type="xsd:string"/>
    <element name="fax" nillable="true" type="xsd:string"/>
    <element name="primaryExtension" nillable="true" type="xsd:string"/>
    <element name="primaryPhone" nillable="true" type="xsd:string"/>
  </sequence>
</complexType>
```

# Web Services

## IntakeParticipant

```
<complexType name="IntakeParticipant">
  <sequence>
    <element name="person" nillable="true" type="tns2:Person"/>
    <element name="relationshipCode" type="xsd:int"/>
    <element maxOccurs="unbounded" name="roles" type="xsd:int"/>
    <element name="age" nillable="true" type="xsd:string"/>
    <element name="address" nillable="true" type="tns2:Address"/>
  </sequence>
</complexType>
<complexType name="Person">
  <sequence>
    <element name="birthDate" nillable="true" type="xsd:dateTime"/>
    <element name="countyPersonId" nillable="true" type="xsd:string"/>
    <element name="firstName" nillable="true" type="xsd:string"/>
    <element name="gender" nillable="true" type="xsd:string"/>
    <element name="hispanicFlag" nillable="true" type="xsd:string"/>
    <element name="initial" nillable="true" type="xsd:string"/>
    <element name="lastName" nillable="true" type="xsd:string"/>
    <element name="raceCode" type="xsd:int"/>
    <element name="ssn" nillable="true" type="xsd:string"/>
    <element name="wisacwisPersonId" type="xsd:int"/>
  </sequence>
</complexType>
```

# Web Services – Where?

```
<wsdl:service name="IntakeServiceService">  
  <wsdl:port binding="intf:IntakeServiceSoapBinding"  
    name="IntakeService">  
    <wsdlsoap:address  
      location=  
        "http://localhost:9080/eW_WS/services/IntakeService"/>  
    </wsdl:port>  
  </wsdl:service>
```

# Web Services – How?

```
<wsdl:portType name="IntakeService">  
  <wsdl:operation name="hello">  
    <wsdl:input message="intf:helloRequest"  
      name="helloRequest"/>  
    <wsdl:output message="intf:helloResponse"  
      name="helloResponse"/>  
  </wsdl:operation>  
  
  <wsdl:operation name="doIntake">  
    <wsdl:input message="intf:doIntakeRequest"  
      name="doIntakeRequest"/>  
    <wsdl:output message="intf:doIntakeResponse"  
      name="doIntakeResponse"/>  
  </wsdl:operation>  
  
</wsdl:portType>
```

# Web Services – WSDL Code

- Start with State client
  - Or build your own
- WSDL2JAVA
  - Generates required scaffold & plumbing code – **Do not alter**
- Other tools can generate code also
  - WSAD, Eclipse, Apache Axis

# Web Services – WSDL Code!

- Plumbing ...

- IntakeService.java
- IntakeServiceService.java
- IntakeServiceServiceLocator.java
- IntakeServiceSoapBindingStub.java

# Web Services – WSDL Code!

```
/**
 * IntakeServiceSoapBindingStub.java
 *
 * This file was auto-generated from WSDL
 * by the IBM Web services WSDL2Java emitter.
 * 4
 */

package gov.wi.dhfs.ewisacwis.cia.intake;

public class IntakeServiceSoapBindingStub extends com.ibm.ws.webservices.engine.client.Stub implements
gov.wi.dhfs.ewisacwis.cia.intake.IntakeService {
    public IntakeServiceSoapBindingStub(java.net.URL endpointURL, javax.xml.rpc.Service service) throws
com.ibm.ws.webservices.engine.WebServicesFault {
        if (service == null) {
            super.service = new com.ibm.ws.webservices.engine.client.Service();
        }
        else {
            super.service = service;
        }
        super.engine = ((com.ibm.ws.webservices.engine.client.Service) super.service).getEngine();
        initTypeMapping();
        super.cachedEndpoint = endpointURL;
        super.connection = ((com.ibm.ws.webservices.engine.client.Service) super.service).getConnection(endpointURL);
        super.messageContexts = new com.ibm.ws.webservices.engine.MessageContext[2];
    }

    ...
}
```

# Web Services – WSDL Data!

- Intake.java
  - Intake\_Deser.java
  - Intake\_Helper.java
  - Intake\_Ser.java
  - IntakeParticipant.java
  
  - IntakeParticipant\_Deser.java
  - IntakeParticipant\_Helper.java
  - IntakeParticipant\_Ser.java
  - ...
- Address.java
  - Address\_Deser.java
  - Address\_Helper.java
  - Address\_Ser.java
  
  - Person.java
  - Person\_Deser.java
  - Person\_Helper.java
  - Person\_Ser.java
  - ...

# Web Services – Go!

```
public class Go {
    public static void main(String[] args) {

        IntakeServiceServiceLocator
            loc = new IntakeServiceServiceLocator();
        IntakeService port = loc.getIntakeService();

        Intake intake = new Intake();

        /* County code goes here to set the Intake() elements... */

        //Call the Web Service...
        CIAStatus ciaStatus = port.doIntake(intake);
        ...
    }
}
```

# Web Services – Hello

```
C:\eWS>go run test
```

```
Connecting...
```

```
http://localhost:9080/eW_WS/services/IntakeService
```

```
Tue Oct 31 15:28:21 CST 2006
```

```
2006-10-31 15:29:42
```

```
Hello:test
```

# Web Services – doIntake

```
C:\eWS>go run
```

```
Connecting...
```

```
http://localhost:9080/eW_WS/services/IntakeService
```

```
CIA status:Failed
```

```
CIA errors:12
```

```
CIA messages:4
```

# Web Services – Return Info

## CIA Errors:

County Intake Id is required.  
Invalid value 0 for  
    IntakeCategoryCode  
Invalid value 0 for CountyCode  
Referral date is required.  
Worker not found:0  
...  
At least one participant  
    required.

## CIA Messages:

Default value override  
    Interpreter flag: null --> N  
Default value override  
    Language code: 0 --> 99  
...  
Default value override  
    After Hours flag: null --> N

# Web Services – Security

- HTTPS
  - Point-to-Point
  - Lightweight
- WS-Security
  - End-to-End
  - SOAP Envelope
  - Heavyweight
- X509 certificates

# Web Services – Security

- XRef & CIA built but not secured
- Novell iChain 2.3
- Installing & Testing @ DHFS
- Spring 2007

# Web Services – Future

- **County → State**
  - ONLY XRef and CIA
- **State → County**
  - Case Management
  - REPLite
  - ???
- We're looking for counties to pilot

# Web Services – Future

- Questions
- Comments