

Dynamic Web Services Framework (DWWSF)

Pontis project case study

- Environment overview: Pontis platform
- Technology overview: web services in Java
- DWSF design and implementation
- Field report: migration from the old WS
- Conclusions

Pontis platform

- Model driven development
- Domain specific language
- Meta-programming
- Separation of application and platform
- Customization

Domain specific language

```
<Benefit ID="ReturningCustomerBenefit"  
type="PriceBenefit">
```

```
  <Discount units="%">20</Discount>
```

```
  <ValidUntil>20-Jul-2008</ValidUntil>
```

```
  <Limit inits="$">1000</Limit>
```

```
</Benefit>
```

Meta-programming

- Class definitions are objects
- They are instances of a metaclass(es)
- Metaclass methods (reflection)
- Hierarchy of metaclasses
- Classes inherit, classes are objects - so objects inherit
- Extending and restricting inheritance

Metaclasses usage

```
MethodInfo {  
  String name  
  ParamInfo[] params  
  Type returnType  
}
```

```
Class {  
  String name  
  MethodInfo[] methods  
}
```

```
Class Object {  
  //methods  
  equals (Object o) : bool  
}
```

```
Object o = new Object  
o.equals ("other") //  
  returns false
```

```
StateMethodInfo : MethodInfo {  
  String in  
  String out  
}
```

```
StateClass : Class {  
  @Refine StateMethodInfo[] methods  
}
```

```
StateClass StateObject {  
  String state="new";  
  //methods  
  @in("new")@out("ready") init ()  
  @in("ready") run ()  
  @in("ready")@out("dead") stop ()  
}
```

```
StateObject o = new StateObject  
o.run () //fail: expected "ready"  
  have "new"
```

Pontis platform: TGPA

- TGPA declaring templates (classes)
- TGPA declaring profiles (beans)
- TGPA generic class (java implementation)
- TGPA implementation method example
- What user sees?

TGPA declaring templates (classes)

```
<TgpType ID="A_WebService" xsi:type="platform.tgp:BaseTypeDef"/>

<A_Template ID="WebService" xsi:type="platform.tgp:BaseTemplate">
  <Implements>A_WebService</Implements>
  <Properties>
    <Property
xsi:type="platform.framework:StringDataItem">
      <Name>serviceName</Name>
    </Property>
    <Property xsi:type="platform.tgp:BaseProperty">
      <Name>authenticator</Name>
      <Type xsi:type="platform.features.tgpType">
<tgptype>platform.features#A_WebServiceAuthenticator</tgptype>
      </Type>
    </Property>
  </Properties>
</A_Template>
```

TGPA declaring profiles (beans)

```
<A_WebService ID="Amazon_ws"  
  xsi:type="platform.features:WebService">  
  <serviceName>amazon</serviceName>  
  <authenticator  
    xsi:type="platform.features:WebServiceXPathAuthenticator">  
    <XPathUser>//userName</XPathUser>  
    <XPathPassword>//userPassword</XPathPassword>  
  </authenticator>  
</A_WebService>
```

//in java code:

```
A_WebService ws = (A_WebService) kernel.getProfile  
  ("Amazon_ws");  
ws.getAuthenticator().authenticate(inMessageContext);
```

```
A_Template wsTemplate = (A_Template)  
  kernel.getProfile("platform.features#WebService");
```

TGPA generic class (java implementation skeleton)

```
<A_Template ID="WebServiceXPathAuthenticator" based-  
on="platform.features#WebServiceAbstractBaseAuthenticator"  
xsi:type="platform.tgp:BaseTemplate">  
  <Implements>A_WebServiceAuthenticator</Implements>  
</A_Template>
```

```
public interface A_WebServiceAuthenticator extends A_GC  
{  
    void authenticate(MessageContext inMessage);  
}
```

```
public class WebServiceXPathAuthenticatorGC extends  
    WebServiceAbstractBaseAuthenticatorGC implements  
    A_WebServiceAuthenticator  
{  
}
```

TGPA implementation

```
public class WebServiceXPathAuthenticatorGC extends
    WebServiceAbstractBaseAuthenticatorGC implements
    A_WebServiceAuthenticator
{
    public void authenticate(MessageContext inMessage)
    {
        String user = getXPath (inMessage.getEnvelope().getBody(),
            myProfile.getXPathUser());
        String passwd = getXPath(inMessage.getEnvelope().getBody(),
            myProfile.getXPathPassword());
        doLogin(user, passwd);
    }
    private String getXPath(OMElement element, String xpath)
    {...}
}
```

What user sees?

The screenshot shows a web browser window titled "Pontis - User-defined content item structures (User-defined content item structures) - Microsoft Internet Explorer". The address bar shows the URL "http://localhost:8080/Pontis-WebDesktop/action/Desktop.do". The browser's menu bar includes "File", "Edit", "View", "Favorites", "Tools", and "Help". The address bar also contains "Back", "Forward", "Home", "Search", "Favorites", and "Go" buttons. The main content area is titled "User-defined content item structures (User-defined content item structures)" and features a navigation bar with "Admin" and "Desktop" tabs. The "Desktop" tab is active, showing a menu with "Application References", "Pricing", "Batch Jobs", "Catalog Filters", "User", "Recommendation Settings", "Questionnaire", and "Setup". The main content area is divided into sections: "XML" (with "Save & Exit", "Save", and "Cancel" buttons), "Lifecycle phase" (with "Draft" and "Availability" options), and "General" (with a sidebar icon). The "General" section contains the following fields and options:

- Name ***: Text input field containing "Movie item".
- Acts as ***: Dropdown menu showing "TV Content Item".
- Should be saved in database**
- Description ***: Text area.
- Add**: Button with a dropdown arrow.
- Export displayed columns**: Button with a dropdown arrow and a grid icon.

Below these fields is a table with two columns: "Name" and "Data type".

Name	Data type
rating	A Long

The browser's status bar at the bottom shows "Done" and "Local intranet".

Technology overview: web services in Java

- Axis and Axis 2
- XFire and CXF
- JAX-WS
- Spring WS
- Others?
- Deployment options
- Code 1st or contract 1st
- Annotations or XML
- Static or dynamic



[Continuous Integration](#) - 5 min setup, triggers, quiet period notifications, distributed tests. www.anthillpro.com

Ads by Google

Stack Comparison

There SOAP stack space has gotten more crowded recently. This chart is to help you decide which stack to use.

If you have any corrections/additions please direct them to the mailing list.

While feature matrices can be helpful, we think you should keep some other points in mind which are equally important.

Performance - XFire is one of the fastest SOAP stacks available. We'll have some benchmarks coming soon, but a rough guide is that we're 2-5x faster than Axis 1.

Robustness - XFire is now at 1.2, it has been in development for over 2.5 years, and it has deployed in many large organizations around the world.

Ease of Use - XFire is significantly easier to use than a lot of SOAP stacks.

Embeddability - The SOAP stacks below have various degrees of embeddability. This may not be a factor in your application design, but here are our thoughts on the issue:

Axis 1's big embeddability flaw is that its API was never meant to be used by an end user. Also, it uses static references to the AxisEngine everywhere in the code, making it impossible to run two completely separate instances side by side.

Axis 2 [seems to be a bit more embeddable](#), although the API seems kind of ugly and there isn't any documentation on the subject. (Dan Diephouse: It also makes the mistake in my opinion of trying to be the equivalent of a J2EE container, which means it mucks with classloaders, has its own deployment model, etc. That is what Spring, JBI, other containers are for, so XFire doesn't feel the need to replicate that.)

Celtix - seems embeddable.

Glue - definitely embeddable.

JBossWS - We haven't had time to play with this one yet.

XFire - [Check out our sample](#).

Codehaus XFire

[Home](#)
[Bug/Issue Reporting](#)
[Download](#)
[Eclipse Plugin](#)
[Maven Archetypes](#)
[FAQ](#)
[Get Involved](#)
[License](#)
[News](#)
[Performance](#)
[Stack Comparison](#)
[Support/Mailing Lists](#)
[Team](#)
[Who uses XFire](#)
[XFire and Celtix Merge](#)

Documentation

[Articles](#)
[Javadocs](#)
[User's Guide](#)
[Release Notes](#)

Quicklinks

[Aegis Binding](#)
[Bindings](#)
[Client](#)

ALEXEY ZAVIZIONOV

FRIDAY, JULY 6, 2007

Web Services: AXIS, XFire, CXF, JAX-WS

[wiki](#)

Axis1

[official and wiki](#)

[JIRA and mail](#)

[Release notes and what in this realise](#)

[User Guide](#)

[Commands - Axis Reference Guide](#)

- SOAP 1.1/1.2 compliant engine
- Flexible configuration / deployment system
- Support for "drop-in" deployment of SOAP services (JWS)
- Support for all basic types, and a type mapping system for defining new serializers/deserializers
- Automatic serialization/deserialization of Java Beans, including customizable mapping of fields to XML elements/attributes
- Automatic two-way conversions between Java Collections and SOAP Arrays
- Providers for RPC and message based SOAP services
- Automatic WSDL generation from deployed services
- WSDL2Java tool for building Java proxies and skeletons from WSDL documents
- Java2WSDL tool for building WSDL from Java classes.
- Preliminary security extensions, which can integrate with Servlet 2.2 security/roles
- Support for session-oriented services, via HTTP cookies or transport-independent SOAP headers
- Preliminary support for the SOAP with Attachments specification
- An EJB provider for accessing EJB's as Web Services
- HTTP servlet-based transport

ABOUT ME



ALEXEY ZAVIZIONOV

[All my issues](#) [All my photos](#) I'm originally from

Krasnodar, Russia. I'm 24 years old. I'm a Russian computer science engineer, have a Master in Applied Mathematics from the Kuban State University, Russia. Now i'm developer of eXo Platform (Ukraine Office), joined in May of 2006.

[VIEW MY COMPLETE PROFILE](#)

Any feedback and suggestions are welcome.

Любые пожелания и предложения приветствуются.

8 февраля моему блогу исполнился один год!

ПОМОГ ЛИ ВАМ МОЙ БЛОГ?

<http://zavizionov.blogspot.com/2007/06/axis.html>

Axis and Axis 2

- Popular and well supported
- Exists for Java and C
- Progressing in good direction
- Customizable and extendable
- Worked from a first try
- “No one gets fired for choosing IBM Axis”

XFire and CXF

- New, cool, easy to use
- Comparable to Axis in WS-* coverage
- Solves problems when Axis fails
- ServiceMix ESB support

JAX-WS

- Official standard
- Annotation spec
- Used by many Axis2, CXF, etc.
- Understood by JavaEE 5 container
- etc.?

Spring WS

- Makes the Best Practice an Easy Practice
- Powerful mappings
- XML API support
- Flexible XML Marshalling
- Reuses your Spring expertise

Other WS frameworks?

Code 1st or contract 1st

- Code 1st

- Developer works with familiar concepts, WSDL is created automatically
- Control on WSDL via annotations and XML configuration files
- Java interfaces don't change
- WSDL is unstable

- Contract 1st

- Easy control what WSDL will look like
- WSDL does not change - good for remote teams
- Java interfaces are unstable

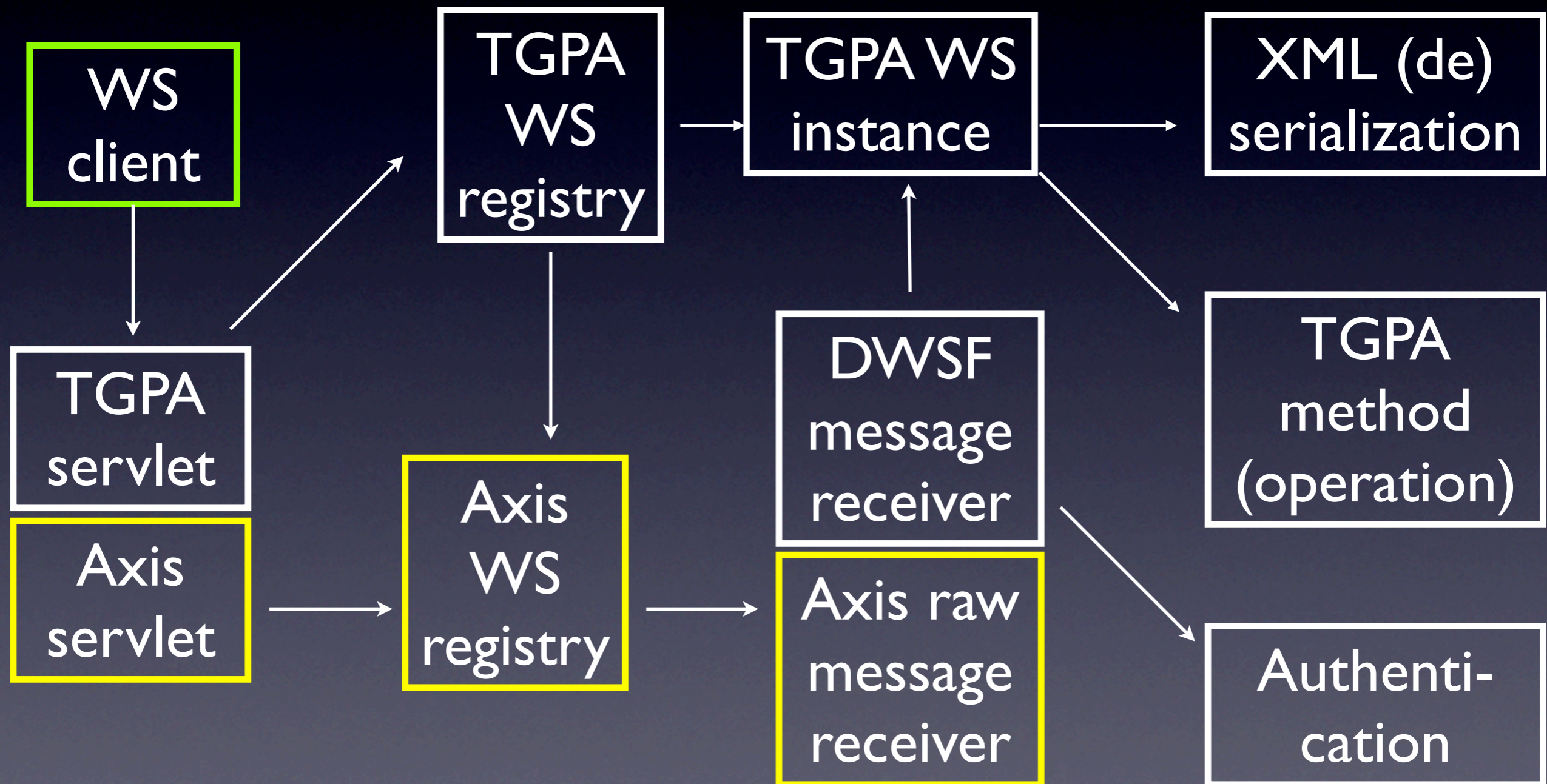
WS in Pontis

- Previous approach
- Motivation for change
 - Remove lots of generated proxy classes
 - Eliminate Java coding for creating webservices
 - Support of dynamic model changes at runtime

DWSF design and implementation

- DWSF high-level design
- Axis2: packaging and deployment options
- Axis2: dynamic services creation/removal
- Authentication – different options
- Custom handling of SOAP messages

DWSF high-level design



Axis2: packaging and deployment options

- WS provider is a library, Axis2 is a WAR
- WS provider is a WAR, Axis2 is Eclipse magic
- WS provider is a WAR, Axis2 is a servlet from library - undocumented
- WS consumer - Axis2 is a proxy
- WS consumer - Axis2 is a library
- Demo

Services creation/removal with Axis

- Dynamic nature of AxisConfiguration
- Creating services - few options
 - services.xml
 - implementing class
 - WSDL and message receivers
- Removing services

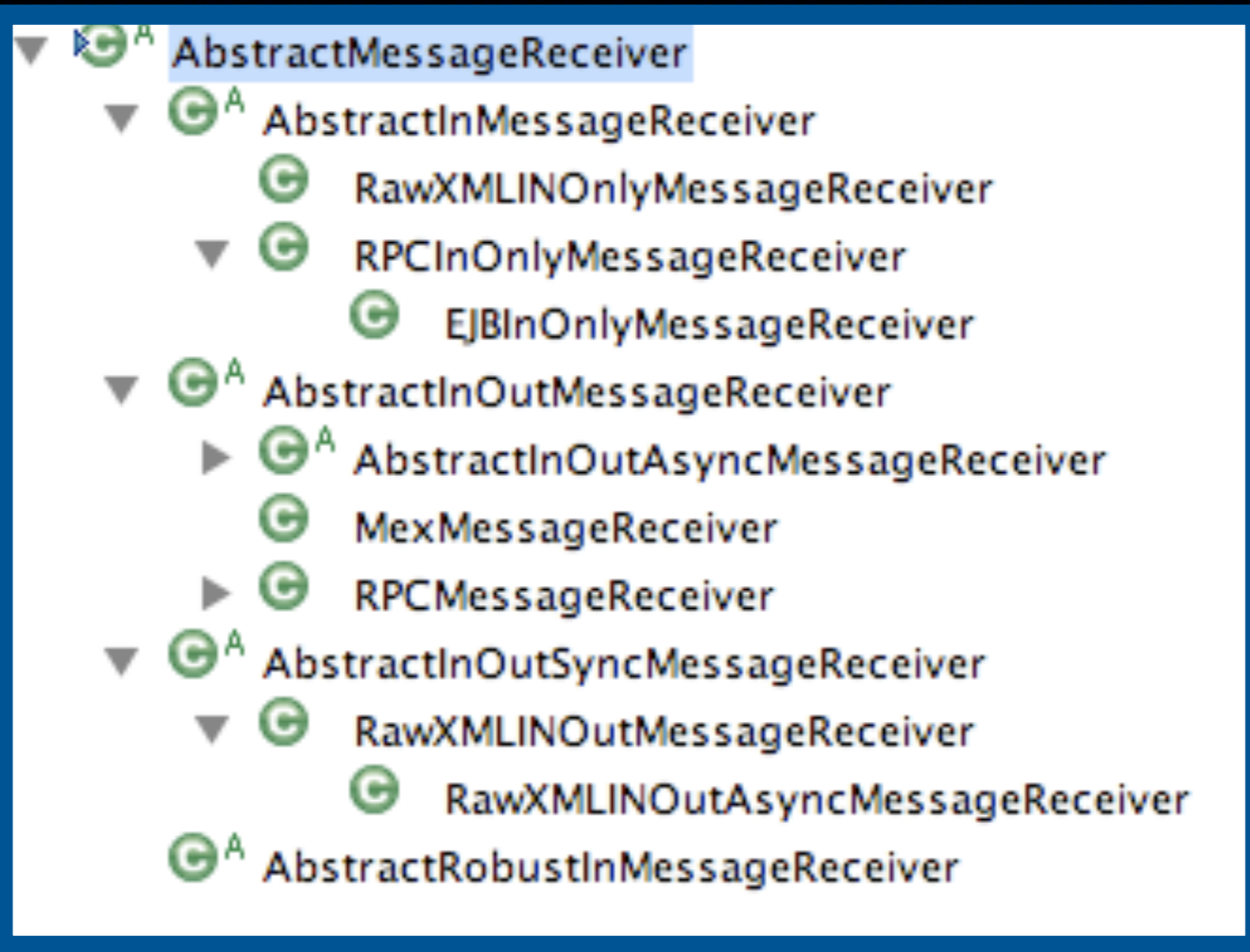
Authentication – different options

- HTTP headers
 - Basic
 - Digest
 - Certificate
- WS-Security
- Application-level

Custom handling of Soap messages

- Axis2 dynamic handler chains
 - First, last, before given handler, after
 - Transport in, pre-dispatch, dispatch, post-dispatch
 - Message receiver

Axis2 message receivers



Field test: replacing an old WS in Pontis

- WSDL should remain intact
- Schema types in different XML namespaces
- Short names not suitable for exact mapping in TGPA
- Result: hundreds LOC of real Java code and over thousand LOC of generated Java code replaced by 30-40 lines of TGPA

Conclusions

- Java WS frameworks are very flexible
- Dynamic abilities of WS are not used wide enough
- Open source frameworks must be learned by looking into the source code in addition to the documentation
- Pontis platform is very flexible
- Its dynamic nature demands using advanced features of Java technologies
- “Every line of code is design”
- decisions took by a coder often influence the whole system

Questions