Dynamic Web Services Framework (DWSF)

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)

```xml
<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?

Technology overview: web services in Java

- Axis and Axis 2

- XFire and CXF

- JAX-WS

- Spring WS

- Others?

- Deployment options

- <u>Code 1st or contract 1st</u>

- Annotations or XML

- Static or dynamic

http://xfire.codehaus.org/Stack+Comparison

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 sciense 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 implemention

- 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
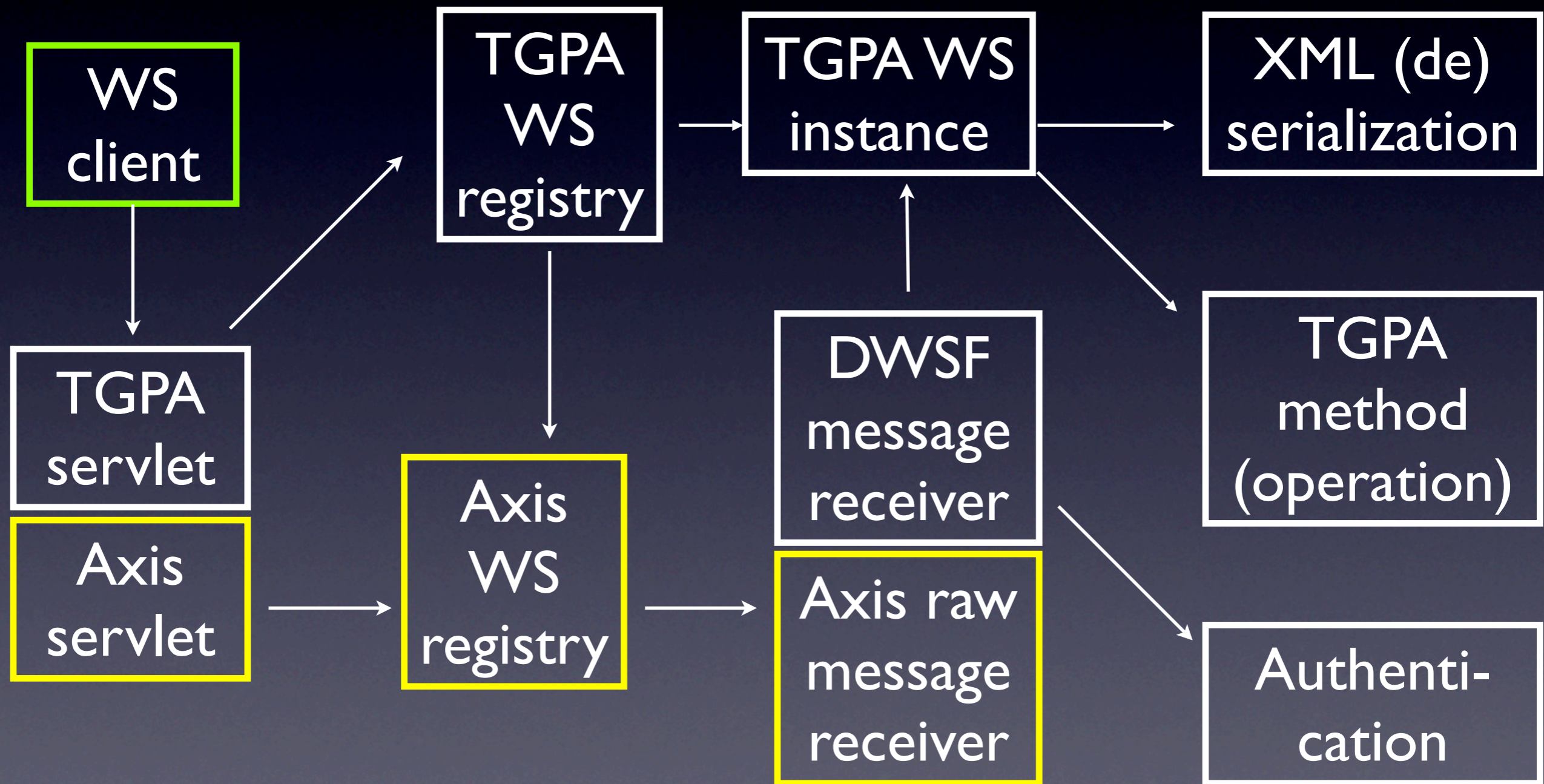
Axis2: packaging and deployment options

- 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
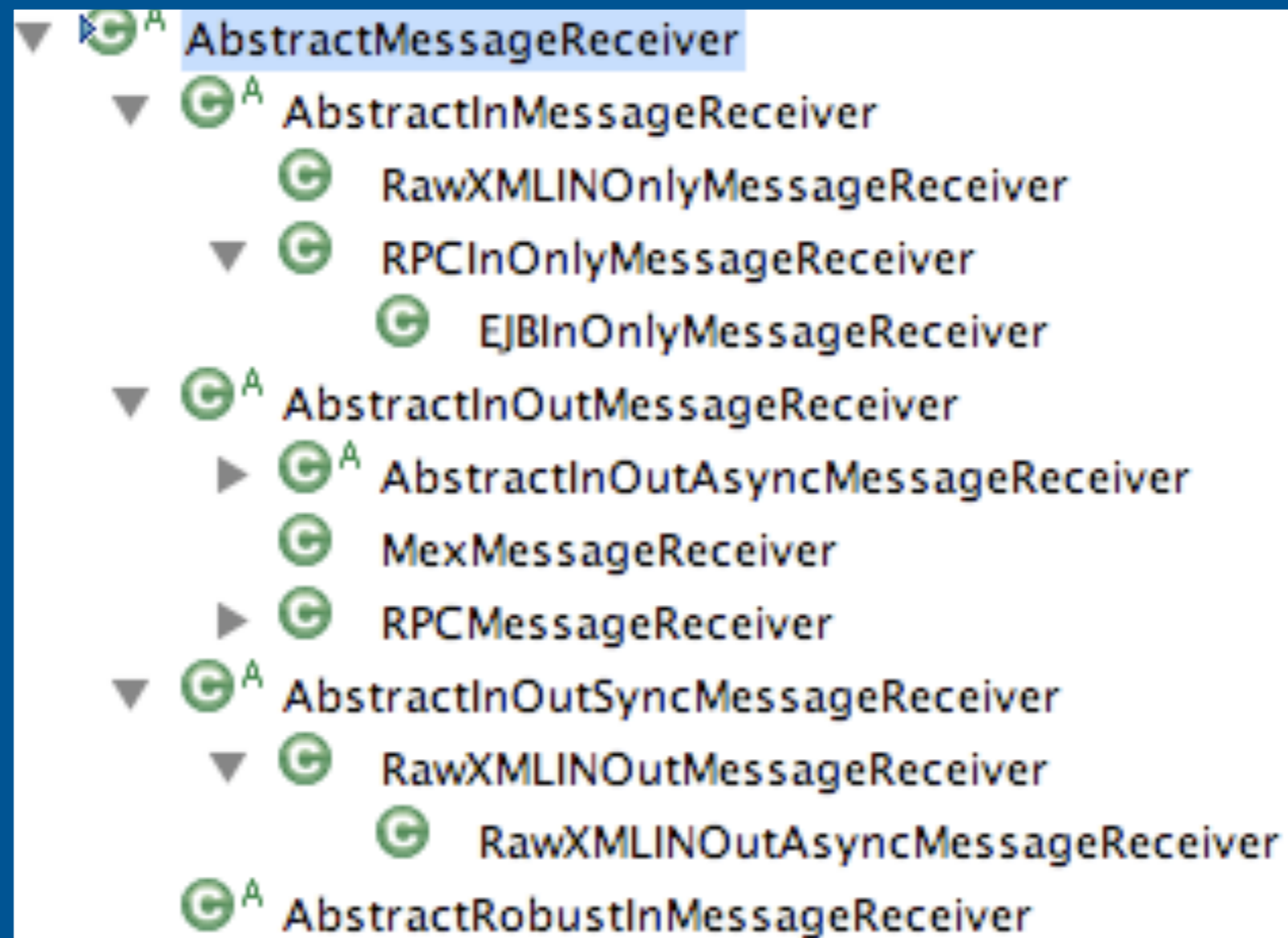
- Removing services

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