

Wellington Java User Group  
For people interested in Java

# Best of '06

17<sup>th</sup> January 2007

Wellington Java User Group

For people interested in Java

# Webstart, and the Webstart changes in Java 6

Martin Paulo



What the formerly proprietary  
ColdFusion developer  
community learned  
from Java in 2006

Kai König, 17/01/2007



## Some context

- I've started doing Java in 1996 when working at a university in Germany
- Got pulled into using Adobe (f.k.a. Macromedia, f.k.a. Allaire) ColdFusion in 1998/99 during the dot-com boom all over Europe
- At that time CF 3.1/4 was up to date and it was totally proprietary and locked!

# ColdFusion on Java



- 2002 Macromedia started to rebuild CF on top of Java (as a J2EE application sitting on their JRun 4 application server) and introduced object-like components
- But: it was easily possible to hook CF into Java now... well... somehow
- Traditionally a lot of CF people have a non-programming background, they start getting into it via scripting HTML pages... which often ends up in non-maintainable and evil code!

## 2004-2005: a new level of professionalism



- In 2004, the first few people started dealing with object-based or object-oriented approaches in ColdFusion
- The first MVC frameworks were developed and released
  - Fusebox (front controller based, page centric)
  - Mach II (the CF equivalent to Struts)

# 2006: even better



- Adobe gave us Flex – a Rich Client technology based on Flash which could be used perfectly with CF or Java backends (don't miss my talk next month! 😊)
- Flex is OO – CF people were forced to re-think their backend approaches as well
- More and advanced frameworks came up...
  - ColdSpring (the ColdFusion Spring equivalent)
  - Reactor (persistence framework)
- ...but also ideas from other web technologies were introduced
  - Various scaffolding frameworks (as Ruby on Rails offers)
- Finally CF people got into open source: RIAForge.org was launched, for open source projects built on ColdFusion, Flash and Flex.



2007: ???

- ColdFusion MX 8 is due, and Adobe promised to make the Java integration more flexible
- ColdFusion hopefully moves to Java 5 or 6

# Jython case study

---

Dealing with the edge cases.

Richard Schmidt  
Metservice  
hangstrap@gmail.com

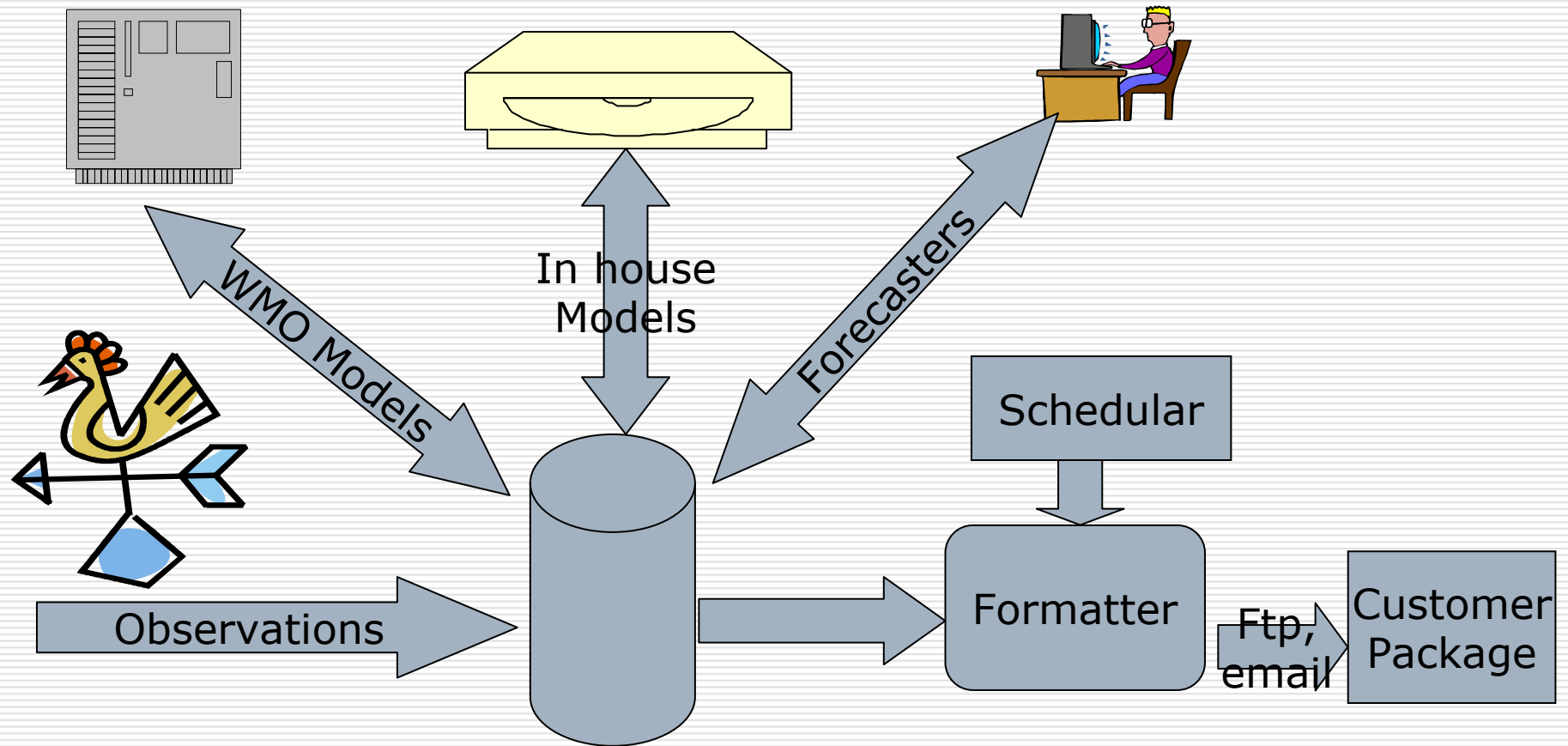
# Metservice

---

- ❑ Seller of weather related information
  - ❑ Most of this information is public domain
  - ❑ VAX / VMS legacy system
  - ❑ Java demon applications running in a pipeline
-

# Metservice Data flow

---



# Data flow

---

- ❑ Observations are open to all.
  - ❑ Super computers generate large scale models, available to all.
  - ❑ In house programs refine models.
  - ❑ Forecasters add human touch.
  - ❑ Metservice sells information to customers.
-

# Packages

---

- 98% map to a small set of user cases
  - System designed for these user cases.
  - How do you handle the unusual requirements?
-

# Standard Package Fabrication

---

- ❑ Package defined as a set of scripts stored in the database
  - ❑ Selector scrip
    - Generates a tree of weather objects from the database. (magic)
  - ❑ Transform script
    - Renders the tree of objects as a document.
    - Proprietary language.
-

# Those damn edge cases

---

- Currently need to
    - Extend the language
    - Implement code
    - Test, Quality Control
    - Release to production
  - Slow process (not agile)
  - Depends on one developer!
-

# Example: Txt messages

---

- Send forecasts as TXT message to customers.
    - Need to convert forecast to 'TXT' speak.
    - Will require dictionary, parsers, etc
    - Do we want to add this feature to our standard language?
      - No idea if product will sell
      - Customer wants it now!
-

# Current workaround

---

- ❑ Ftp package to known directory
  - ❑ External application scans directory for file
  - ❑ Processes file when it arrives
  - ❑ Sends the final output file to customer
    - End up with multiple disjointed applications – yuk!
-

# Jython to the rescue

---

- Jython is a java version of Python.
    - Python popular at Metservice.
  - Can be run from inside a Java VM.
  - Jython code can create Java objects, call static methods, etc.
-

# Jython fabrication

---

- ❑ Edge case packages are interpreted as a Jython script.
  - ❑ Script must return the same object as a normal package which will be transmitted in the normal way.
  - ❑ Common fabrication processes are made available as services
    - Use selection script to generate tree of objects.
    - Velocity service ....
-

# Advantages

---

- ❑ Can use the best solution for the problem.
  - ❑ Quick fix to customer requirement.
  - ❑ Producing edge case packages part of the standard process.
-

# Disadvantages

---

- ❑ Easy to break scripts when refactoring code.
    - Need to have a way of 'freezing' objects used in services.
  - ❑ How do we unit test?
  - ❑ Script writer has the keys to the safe!
-

# Running Jython from Java

---

```
PythonInterpreter interp = new PythonInterpreter();
```

```
//add services
```

```
interp.set( "owner", owner);
```

```
//run the script
```

```
interp.exec( script);
```

```
//extract the output
```

```
PyObject pObj = interp.eval( "output");
```

```
return (IOutput )pObj.__tojava__( IOutput.class);
```

---

# Jython script

---

```
RADARS = ""
    <p>%s</p>
    ""

def generateRadar( ):
    radarCube = cubeCombiner.findCube("RADAR")
    radarIntervalAxis = radarCube.findIntervalAxis( "dunedin","radar")
    radars = []

    for item in radarIntervalAxis.all( IntervalAxisIteratorStrategyFactory.allNotMissing())
        validFrom=DateFormatter.format( item.validityPeriod().from(),"Mask{'Radar image at '
        h24z minz} Round: true Tz: Pacific.Auckland")
        attachmentName= DateFormatter.format( item.validityPeriod().mid(),
        "Mask{'Wel_radar_' h24z minz '.gif'} TZ:UTC")
        parameters = {'validFrom': validFrom, 'attachmentName':attachmentName}
        radars.append(RADARS % parameters

    return ".join(radars)
```

---

# Wellington Java User Group

For people interested in Java



Nigel Charman

# Groovy

- Dynamic language that compiles to Java bytecode
- Can call Groovy from Java code and vice versa
- Groovy 1.0 recently released
- Standardized under JSR 241
- Language features include
  - static and dynamic typing
  - native syntax for lists, maps, arrays and regex
  - closures
  - operator overloading
- Groovy Development Kit:
  - adds new methods to the JDK eg. `java.lang.String.reverse()`
  - adds new classes, eg Groovy SQL, GPath, Builders
- Projects using Groovy include:
  - Grails, GSP, Groovlets, Groovy SOAP, .....
- Groovy Beans can be managed using Spring 2.0

# Groovy Example

// Connect to a database, read rows with ID 1, 3, and 4,

// For each row, read zipped XML from a blob column, unzip it, parse it, find and print a specific element

```
String dbName="SAMPLE"
```

```
println "-- Running Example.groovy on database ${dbName}"
```

```
def sql = groovy.sql.Sql.newInstance("jdbc:db2:${dbName}", "user", "pass", "com.ibm.db2.jcc.DB2Driver")
```

```
[[1, "1234"], [3, "2211"], [4, "8301"]].each({  
    int rowId, String customerId ->
```

```
    sql.eachRow("SELECT ZIPPED_XML FROM JUG.EXAMPLE WHERE ID=${rowId}", {
```

```
        def zippedXml = it[0]
```

```
        if (zippedXml != null) {
```

```
            InputStream is = new ZipInputStream(zippedXml.getBinaryStream())
```

```
            is.getNextEntry()
```

```
            String xml = is.getText()
```

```
            def root = new XmlSlurper().parseText(xml)
```

```
            def customerNode = root.Update.Customer.find{it.@Id == customerId}
```

```
            println customerNode.GivenName
```

```
        }  
    })
```

```
})
```

Wellington Java User Group

For people interested in Java

# Dependency Injection and the Spring Framework

Russell Healy

# DI and Spring

Dependency Injection  
and the Spring Framework

# Who am I?

Russell Healy

Solution Architect

Ministry of Social Development

Spring user since 2004

# Agenda

- Interfaces
- Simple scenario
- Naïve implementation
- Dependency injection without Spring
- Dependency injection with Spring

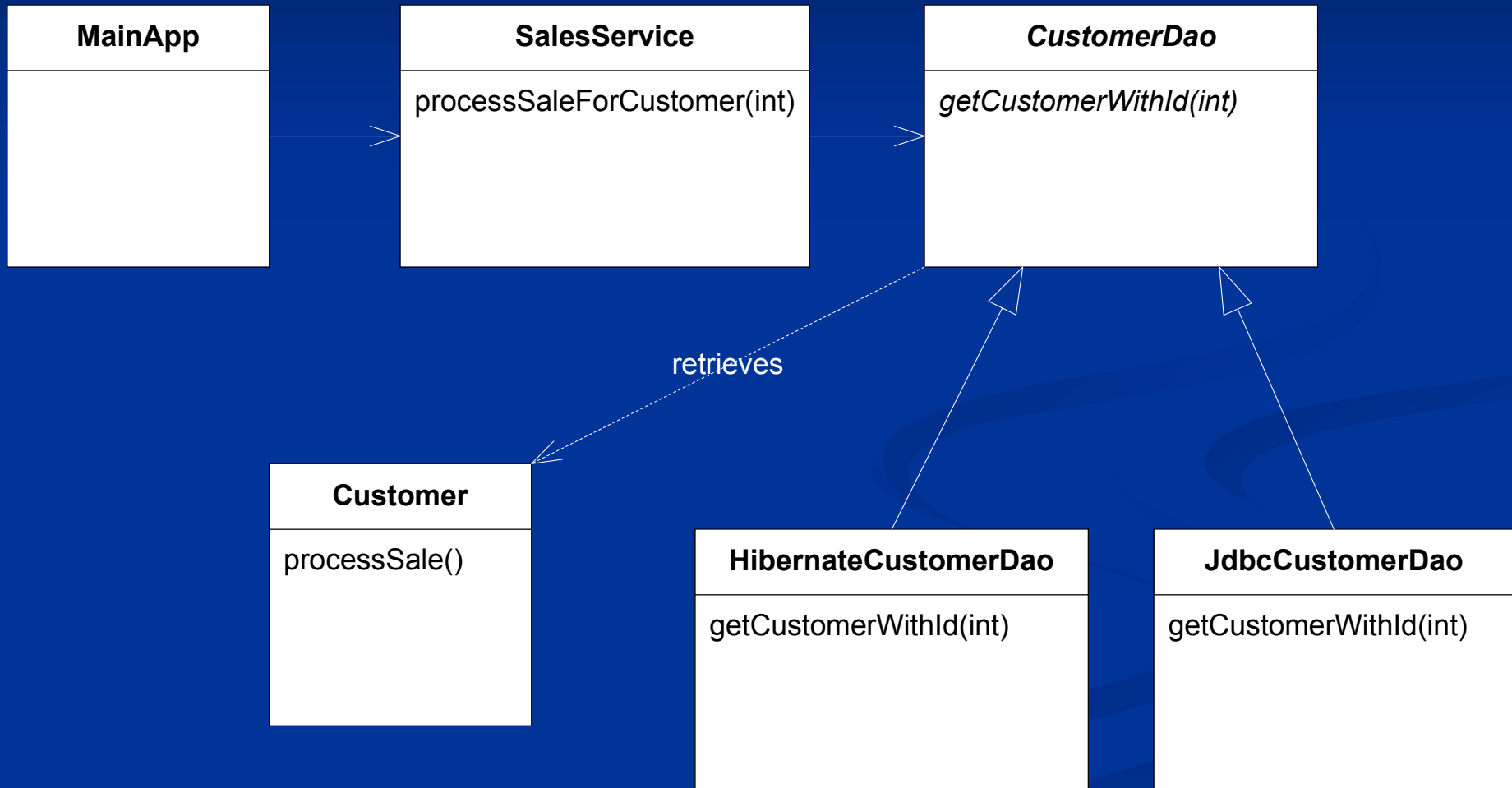
# Program to Interfaces

- GoF OO design principle:
  - “Program to an interface, not an implementation”
    - “Clients remain unaware of the specific types of objects they use, as long as the objects adhere to the interface the clients expect”
- Enables loose coupling - dependencies can be changed without changing the client
- One benefit is easier unit testing

# Scenario: Customer DAO

- An application for processing customer sales
- Classes:
  - MainApp has a SalesService.
  - SalesService has a CustomerDao responsible for returning a Customer given a customerId.
  - CustomerDao is an interface and two implementations are provided: Hibernate and JDBC
  - Customer has a processSale() method

# Class Diagram



# Dependencies

- SalesService *has a* CustomerDao
- SalesService *depends on* CustomerDao
- CustomerDao is a *dependency* of SalesService
- SalesService is a *client* of CustomerDao
- SalesService and CustomerDao are *collaborators*

# Wiring Collaborators

- Naïve implementation: client instantiates concrete dependency
- Dependency injection: collaborators are wired up by a third party
- Spring can be that third party
- Alternatives: abstract factory, service locator

Demo

# Apache Axis 2 1.1

Thilo Frotscher

# Apache Axis2 1.1

- StAX-Parsing with AXIOM
- Communication: sync, async & arbitrary MEPs
- Pluggable data binding: ADB, XML Beans, JiBX...
- “Contract First” or “Code First”
- POJO support / Spring integration
- Arbitrary service implementations possible
- Service lifecycle support & session handling
- Support for REST and MTOM/XOP
- Plug-ins for Eclipse and IntelliJ IDEA
- Simple extension mechanism through “modules”

# Apache Axis2 1.1

- Supported WS-Specs
  - WS-Addressing (built-in)
  - WS-Policy (built-in)
  - WS-Security & WS-SecureConversation (Rampart)
  - WS-Trust (Rahas)
  - WS-ReliableMessaging (Sandesha2)
  - WS-Transactions (Kandula2)
- Next releases
  - JAX-WS support
  - Data binding support for JaxMe, JAXB-RI
  - Code generation for C#, C++

Wellington Java User Group  
For people interested in Java

# Data Binding and the tie-ins to Web Services

Dennis Sosnoski

Wellington JUG - 17 January 2007

# The Best of 2006

Dennis M. Sosnoski  
<http://www.sosnoski.co.nz>

<[www.sosnoski.co.nz](http://www.sosnoski.co.nz)>

---

# Web services and data binding

- In general:
  - Axis2 1.0, 1.1, 1.1.1 stack releases
  - JAX-WS/JAXB stack releases
    - Glassfish open source projects
    - Bundled into Java 6
  - .NET 3.0
- Common set of SOAP extensions:
  - WS-Security, WS-ReliableMessaging, WS-Addressing, XOP/MTOM

# My JiBX project

- JiBX flexible data binding for Java
  - Binding definitions relate code to XML
  - Bytecode enhancement implements conversions
- 2006 progress:
  - 1.0.x, 1.1.x releases (1.1.3 most recent)
    - Many new features (“flexible” unmarshalling, etc.)
  - Axis2 support

## JiBX state now

- Existing JiBX tools weak
  - Xsd2Jibx – horrible trash
  - GenBinding, GenSchema – outdated
  - JibxSoap – outdated, needs more support
  - Eclipse plug-in – not supported, outdated
- Need new tools for Web service push
  - Expose existing code as Web service
  - Generate Web service client from WSDL

# Work in progress

- Developing Jibx2Wsdsl tool
  - Given existing source code, generate binding, schema, and WSDL
  - Uses updated GenBinding and GenSchema tools
  - Uses part of JibxSoap WSDL code
- Developing code generator from schema
  - Full schema model and analysis code
  - Build binding using binding model, classes using Eclipse AST

## Ahead in 2007

- Wsdl2Jibx tool
  - Generate Java code and binding from schema
  - Generate client and/or server glue code for JibxSoap, Axis2, and perhaps XFire
- Eclipse plug-in support
  - Commercial plug-in with full support
  - Initial version integrates binding compiler
  - Later versions add support for JiBX tools
- XOP/MTOM support in JiBX 1.2 release

# Ahead in 2007

- JibxSoap 1.0?
  - Would like to bring to full production level
    - Add WS-Addressing, alternate transports and asynchronous messaging
    - Add XOP/MTOM attachment support
    - Add POX support as full alternative to SOAP
  - Finding a paying client would help...

# JiBX 2.0

- The big enchilada
  - Complete rewrite on bytecode generation
  - Source code as alternative to bytecode
    - Generate conversion code as separate source files
    - Generate/update conversion code embedded in source files
  - Binding definition rationalization
    - Needs cleanup after years of incremental changes
- May have a client for this...

Wellington Java User Group  
For people interested in Java

# JUnitFactory

Nigel Charman

# JUnitFactory

- An experimental “characterization” test generator
- Characterizes and records what the code actually does - not what it's **supposed** to do
- Useful for working with legacy code and supplementing manual unit testing
- “Test Helpers” allow user input to test data selection and assertions

# JUnitFactory

## Non-commercial use - JUnitFactory (free)

Request an invitation from [www.junitfactory.com](http://www.junitfactory.com)

Upcoming Webinar -

<http://www.agitar.com/news/events/webinar junit factory.html>

## Commercial use – AgitarOne™

Server based product that includes Agitator, JUnit Test Generation, Code Rules, Continuous Integration Server, Management Dashboard

See [www.agitar.com](http://www.agitar.com)

For further details, email [nigel.charman@assurity.co.nz](mailto:nigel.charman@assurity.co.nz)

## Academic institutions – ‘No Java Class Left Behind’

– Free AgitarOne™ license

– Unit testing course material being made available

For further details, email [nigel.charman@assurity.co.nz](mailto:nigel.charman@assurity.co.nz)