OW2 code base
Component models in Action

Gaël Blondelle
Chairman Technology Council OW2

Alexandre Lefebvre
CTO OW2
At OW2, we always thought that for middleware developers a component model is not just a possibility: it’s a necessity!
Agenda

• OW2

• Component models: OW2 Fractal and OSGi
  • Component models in action
  • Component models benefits
A global *community* working together to *develop open source middleware* and to foster a *vibrant business ecosystem*

*Middleware at large, including tools for the development, deployment and management of distributed applications*
A Truly Global Community

Strategic Members

1. Alcatel Lucent
2. Beihang U.
3. Bull SAS
4. CVIC SE
5. DOCSC
6. Engineering
7. France Telecom
8. INRIA
9. ISCAS
10. NUDT
11. Peking U.
12. Red Hat
13. SERPRO
14. Thales

14 Strategic Members, 50+ Corporate Members
1000+ Individual Members
OW2 Industry-Grade Projects
2007 top downloads

- funambol
- asm
- exoplatform
- jonas
- xwiki
- ops
- spagobi
- jotm
- acceleo
- bonita
- javaservice
- shark
- lomboz
- salomé-tmf

2,456,900 downloads
Project life cycle

**Incubator**
- Code being developed, prototype
- No real end user yet

**Mature**
- Industry-grade code
- Users (success stories)
- Commercial support
- Academic activity

**Archive**
- No more development
- Maybe some light evolutions
<table>
<thead>
<tr>
<th>Incubator</th>
<th>Mature</th>
<th>Archive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspire RFID</td>
<td>Acceleo</td>
<td>SaloméTMF</td>
</tr>
<tr>
<td>Azuki</td>
<td>ASM</td>
<td>SAT4J</td>
</tr>
<tr>
<td>BEEN</td>
<td>Bonita</td>
<td>SOFA</td>
</tr>
<tr>
<td>CMI</td>
<td>CARDAMOM</td>
<td>Spagic</td>
</tr>
<tr>
<td>Dream</td>
<td>CAROL</td>
<td>Spago</td>
</tr>
<tr>
<td>Dysoweb</td>
<td>CLIF</td>
<td>Spago4Q</td>
</tr>
<tr>
<td>Elastic Grid</td>
<td>EasyBeans</td>
<td>SpagoBI</td>
</tr>
<tr>
<td>JASMINe</td>
<td>eXoPlatform</td>
<td>Telosys</td>
</tr>
<tr>
<td>JWTGen</td>
<td>FederId</td>
<td>Xwiki</td>
</tr>
<tr>
<td>Perseus</td>
<td>Fractal</td>
<td>Rubis</td>
</tr>
<tr>
<td>xPlus</td>
<td>Funambol Sync4J</td>
<td>Jonathan</td>
</tr>
<tr>
<td></td>
<td>GASP</td>
<td>JORM</td>
</tr>
<tr>
<td></td>
<td>InterLDAP</td>
<td>Massiv</td>
</tr>
<tr>
<td></td>
<td>JavaService</td>
<td>MEDOR</td>
</tr>
<tr>
<td></td>
<td>JOnAS</td>
<td>Monolog</td>
</tr>
<tr>
<td></td>
<td>JOnASPlugin for Eclipse</td>
<td>Odette-J</td>
</tr>
<tr>
<td></td>
<td>JORAM</td>
<td>Oscar</td>
</tr>
<tr>
<td></td>
<td>JOTM</td>
<td>ParGRES</td>
</tr>
<tr>
<td></td>
<td>Lomboz</td>
<td>SNAP</td>
</tr>
<tr>
<td></td>
<td>OpenMobile IS</td>
<td>Speedo</td>
</tr>
<tr>
<td></td>
<td>Orchestra</td>
<td>SURF</td>
</tr>
<tr>
<td></td>
<td>PETALS</td>
<td>Tribe</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XAPool</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RmiJdbc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XQuare</td>
</tr>
</tbody>
</table>
Agenda

- OW2
- Component models: OW2 Fractal and OSGi
  - Component models in action
  - Component models benefits
Why components
Components are more than simple objects

• Higher abstraction level
• Business $\leftrightarrow$ technical separation
• Better life cycle coverage
  ○ Conception, implementation, packaging, deployment, execution
Component models, applications and middleware

Applications

• for application: EJB, CCM, .NET/COM+, SCA, Spring

Middleware

• for middleware: Fractal, OpenCOM, OSGi
Components & architecture

• A software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them. [Bass 98]

• Language: ADL, often in XML
• Complementarity
  ○ Architecture: built using components (top-down)
  ○ Components: assembled to build an architecture (bottom-up)
Component models in OW2

OSGi

Fractal
What Is OSGi

- OSGi Alliance Consortium founded in March 1999
- Objective: Create open specifications for delivering administrated services through a network
- Defines
  - A common platform (framework)
    - Services deployment
    - Services execution and administration
  - A set of basic services
    - Log Service, HTTP Service...
  - A device access specification
  - A deployment unit, a bundle
- First Applications
  - Embedded: automotive, home...
- Recent trends
  - Server side, middleware
OSGi Main Concepts

- **Services:**
  - Java Object implementing a well-defined contract

- **Bundles:**
  - Services diffusion and deployment unit

- **Framework:**
  - Bundles execution environment
    - Felix (Apache), Knopflerfish, Equinox, SMF, ProSyst, …
  - Event notification
OSGi Middleware and Application Packaging

• Modularize the middleware/application
  ○ Provisioning the different middleware services
  ○ Better component visibility
  ○ Partial update without restart all

• Easy implementation
  ○ Based on Jarfile and Manifest entries
  ○ Explicit Package dependencies and Versioning
OW2 Fractal component model

- France Telecom Orange Labs R&D, INRIA
- Open source http://fractal.ow2.org

- **History:**
  - End 2000: first thoughts around Fractal
  - 06/2002:
    - 1st stable API version
    - reference implementation (Julia)
    - 1st version of the ADL
  - 01/2004:
    - definition of ADL v2 (extensible ADL)
    - availability of the implementation 03/2004
The Fractal component model

- Systems and middleware engineering
- **Generic** enough to be applied to any other domain
- **Fine grain** (wrt EJB or CCM) close to a class model
- **Lightweight** (low overhead on top of objects)
- Independence from programming languages
- Homogeneous vision of **all layers** (OS, middleware, services, applications)
  - Fractal everywhere
The Fractal component model

- Open and adaptable
- Usable as a component framework to build applications
  - with “standard” Fractal components
- Usable as a component framework framework
  - building different kinds of components
  - with minimum introspection and simple aggregation (à la COM)
  - with binding and lifecycle controllers (à la OSGi)
  - with a two-level hierarchy and bindings (à la SCA)
  - with persistence and transaction controllers (à la EJB)
  - with attribute controllers (à la MBean)
  - …
Fractal components

- Basic units of a Fractal application
- Type model and instance model
- **Components are compile-time and run-time entities**
- Hierarchical model
  - composite and primitive components
- Component sharing
  - For modelling common resources
  - E.g.: data, pools, caches, activities (threads, processes, transactions)
Fractal components

<definition name="foo">
  <interface name="r" role="server" signature="Main" />
  <component name="a">
    <component ... />
    <component name="aa"> ... </component>
  </component>
  <component name="b">
    <component name="aa" definition="a/aa" />
  </component>
</definition>
The 2 dimensions of Fractal components

• Business

• Control
  ○ Non functional (tech’al) properties
  ○ Implemented in the membrane
  ○ Made of a set of controllers
  ○ E.g. security, transaction, persistence, start/stop, naming
  ○ Controllers accessible through a control interface
  ○ Controllers and membranes are open
• The role of Fractal frameworks (Julia, AOKell, …) is to offer means to develop
  ○ Fractal applications
  ○ Controllers and membranes
Fractal, graphically

- A collection interface
- Interface name
- A component
- A binding

On the left:
- External server interfaces
- Internal client interfaces

On the top: a control interface

On the right:
- External client interfaces
- Internal server interfaces
3 complementary tools for developing Fractal applications
- Fraclet
  - programming model based on annotations
- Fractal ADL
  - XML-based architecture description language (ADL)
- Fractal API
  - set of Java interfaces for
    - introspection
    - reconfiguration
    - dynamic creation/modification
  - of Fractal components and component assemblies
Fractal4Eclipse tool

• Eclipse development environment for Fractal applications
FractalExplorer tool

- Management console and run-time monitoring of a Fractal application
Fractal standardization
**Programming**

**Java Parallel Frameworks**
for HPC, Multi-Cores, Distribution, Enterprise Grids and Clouds.

Featuring: Async. comms, Master-Worker, Monte-Carlo, SPMD, components and legacy code wrapping.

**Optimizing**

**Eclipse GUI (IC2D)**
for Developing, Debugging, Optimizing your parallel applications.

Featuring: graphical monitoring and benchmarking with report generation.

**Scheduling**

**Multi-Language Scheduler**

Featuring: graphical user interface, resource acquisition and virtualization.
From Objects to Distributed Components

- Typed Group
- Java or Active Object

Example of component instance

JVM
From Objects to Distributed Components

Example of component instance

Typed Group  Java or Active Object  JVM
From Objects to Distributed Components

Example of component instance

□ Typed Group  □ Java or Active Object  □ JVM
From Objects to Distributed Components

Example of component instance

Typed Group  Java or Active Object  JVM
From Objects to Distributed Components

Example of component instance

Typed Group  Java or Active Object  JVM
From Objects to Distributed Components

Example of component instance

- Typed Group
- Java or Active Object
- JVM
From Objects to Distributed Components

IoC: Inversion Of Control (set in XML)

Typed Group

Java or Active Object

Example of component instance

JVM
From Objects to Distributed Components

IoC: Inversion Of Control (set in XML)

Example of component instance

Truly Distributed Components

Typed Group

Java or Active Object

JVM
From Objects to Distributed Components

IoC: Inversion Of Control (set in XML)

Example of component instance

Truly Distributed Components

Typed Group  Java or Active Object  JVM
Scopes and Objectives:

Grid Codes that Compose and Deploy

No programming, No Scripting, … No Pain

Innovation:

Abstract Deployment

Composite Components

Multicast and GatherCast
Scopes and Objectives:
Grid Codes that Compose and Deploy
No programming, No Scripting, … No Pain

Innovation:
Abstract Deployment
Composite Components
Multicast and GatherCast
Scopes and Objectives:
Grid Codes that Compose and Deploy
No programming, No Scripting, … No Pain

Innovation:
Abstract Deployment
Composite Components
Multicast and GatherCast
Scopes and Objectives:
Grid Codes that Compose and Deploy
No programming, No Scripting, … No Pain
Innovation:
Abstract Deployment
Composite Components
Multicast and GatherCast
invocation parameters
M components
N components
Scopes and Objectives:
Grid Codes that Compose and Deploy
No programming, No Scripting, … No Pain

Innovation:
Abstract Deployment
Composite Components
Multicast and GatherCast
GCM

Scopes and Objectives:
- Grid Codes that Compose and Deploy

Innovation:
- Abstract Deployment

- Composite Components
- Multicast and GatherCast

invocation parameters

M components

N components

MultiCast

www.devoxx.com
GCM

Scopes and Objectives:
Grid Codes that Compose and Deploy
No programming, No Scripting, … No Pain
Innovation:
Abstract Deployment
Composite Components
Multicast and GatherCast

invocation parameters

M components

N components

MultiCast
Scopes and Objectives:
Grid Codes that Compose and Deploy
No programming, No Scripting, ... No Pain

Innovation:
Abstract Deployment
Composite Components
Multicast and GatherCast

---

invocation parameters

M components

N components

MultiCast
GatherCast
Overall, the standardization is supported by industrials: BT, FT-Orange, Nokia-Siemens, Telefonica, NEC, Alcatel-Lucent, Huawei …
ETSI GCM TC Grid Standard

GCM Interoperability Deployment

GCM Application Description

GCM Fractal ADL
   (Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

GCM Interoperability Deployment

GCM Application Description

GCM Fractal ADL
(Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

Official Standard No 1 (June 2008)

GCM Interoperability Deployment

GCM Application Description

GCM Fractal ADL
   (Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

Official Standard No 1 (June 2008)

GCM Interoperability Deployment

GCM Application Description

GCM Fractal ADL
(Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

Official Standard No 1 (June 2008)

GCM Interoperability Deployment

Official Standard No 2 (June 2008)

GCM Application Description

GCM Fractal ADL
(Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

Official Standard No 1 (June 2008)

GCM Interoperability Deployment

Official Standard No 2 (June 2008)

GCM Application Description

GCM Fractal ADL
(Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

**Official Standard No 1 (June 2008)**

GCM Interoperability Deployment

**Official Standard No 2 (June 2008)**

GCM Application Description

**Work Item No 3 (Q1 2009)**

GCM Fractal ADL  
(Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

**Official Standard No 1 (June 2008)**

GCM Interoperability Deployment

**Official Standard No 2 (June 2008)**

GCM Application Description

**Work Item No 3 (Q1 2009)**

GCM Fractal ADL
(Architecture Description Language)

GCM Fractal Management (Java, C, WSDL API)
ETSI GCM TC Grid Standard

**Official Standard No 1 (June 2008)**

GCM Interoperability Deployment

**Official Standard No 2 (June 2008)**

GCM Application Description

**Work Item No 3 (Q1 2009)**

GCM Fractal ADL
   (Architecture Description Language)

**Work Item No 4 Q1 2009)**

GCM Fractal Management (Java, C, WSDL API)

www.devoxx.com
More about GCM

- Attend Denis Caromel’ talk

**Effective SOA and GRIDs with ProActive Parallel Suite**

- Friday 12 December between 10:30 and 11:30
Agenda

- OW2
- Component models: OW2 Fractal and OSGi
  - Component models in action
  - Component models benefits
OSGi @ OW2/ObjectWeb

- Initially
  - OSCAR

- OW2 projects using OSGi
  - Acceleo (Eclipse)
  - AspireRFID (Felix + iPOJO)
  - DysoWeb (Felix)
  - EasyBeans/OSGi
  - JASMINe
  - JOnAS 5 (Apache Felix + iPOJO)
  - Lomboz (Eclipse)
  - Talend (Eclipse)
JOnAS and OSGi:

Scalable platform

OSGi’s architecture and technology
- Dynamic and on demand services
- Services grouped into bundles
- Modularity (Java EE 6 Profiles)
- Pluggable: easier integration with existing software
- Dynamic re-configuration
JOnAS and OSGi:

Scalable platform

User benefits
- Adaptation to user’s needs
- Ease of operations
- Modularity and lightness: facilitates maintenance
- Optimize resources consumption, system footprint
- Dynamic Adaptation, configuration
PEtALS distributed ESB

Front Office

WebServices

Portal

Soap/http Connector

Orchestration BPEL engine

Routing Peer to Peer Adaptative Protocol

Back Office

JMS connector

Tuxedo connector

Soap/http connector

Distributed technical registry

JEE services

Leverages legacy COBOL services

WebServices
Petals Component Architecture using Fractal

Monitoring Components
Petals Component Architecture using Fractal
Extracting monitoring components from PEtALs
PVM and components

Process Languages

The Process Virtual Machine

PVM Core

Services Container (environment)

Pluggable Services
Attend Miguel Valdes’ talk

The Process Virtual Machine, a revolutionary technology for BPM

- Thursday 11 December between 17:30 and 18:50
OW2 CLIF

- Java software framework for load injection and performance measurement
  - Adaptable/extensible because independent
    - from the system tested (invocation protocols, abstractions...)
    - from the definition of load scenarios
    - from the type of monitored resources
    - from the control user interface
  - 100% Fractal based
  - Distributed infrastructure adapted to high injection levels
  - 100% Java (+ specific or native code for system probes)
OW2 CLIF in Fractall
Fractal advantages for CLIF

- Maturity of the Fractal component framework
- Components exist at run-time, key for monitoring and observation
  - Dynamic reconfiguration: adding, replacing or removing load injectors and probes without restarting CLIF
  - Component hierarchy: code re-use can be optimized at the right level
- Distribution support through bindings: important since probes and injectors are distributed
  - ADL: customization possible without programming skills
RFID Everywhere

• Tag everything!

• Readers everywhere!

- Store portals
- RFID tunnel
- Reader for pickin
- NFC phones
  - Over 47,000,000 in Japan
- Nabztag/tag
  - Over 200,000
The ASPIRE FP7 Project Overview and Goals

Develops and will deliver a lightweight, royalty-free, programmable, privacy-friendly, standards-compliant, scalable, integrated and intelligent middleware platform

Make RFI technology solutions affordable to SME:
- Free Middleware running on low-cost hardware
- Lower effort for managing the infrastructure and developing applications

Enable RFID scenarios (based on ASPIRE middleware and added value sensors)

Validate the developments in RFID trials

- United Kingdom
- France
- Greece
- Portugal
- Denmark
- European Union
OW2 project

- Leaders: John Soldatos & Didier Donsez
- Targets non-trivial RFID-based applications
- FOSS SW
  - RFID/Sensor Middleware
  - RFID/Sensor Apps Tools
  - RFID/Sensor Applications samples
- Compliances
  - with standards (*EPCGlobal*, *NFCForum*, …)
  - with legacies (*1D/2D barecods*, …) and proprietary (*iButtons*, …)
- Extensions
  - Sensors (temp., geoloc, video streams, …) and BAT
  - Filters and BEG
- Namespace: `org.ow2.aspirerfid`
- Licence: LGPL v2.1 and CC by-sa
OW2 project

- Leaders: John Soldatos & Didier Donsez
- Targets non-trivial RFID-based applications
- FOSS SW
  - RFID/Sensor Middleware
  - RFID/Sensor Apps Tools
  - RFID/Sensor Applications samples
- Compliances
  - with standards (EPCGlobal, NFCForum, …)
  - with legacies (1D/2D barecods, …) and proprietary (iButtons, …)
- Extensions
  - Sensors (temp., geoloc, video streams, …) and BAT
  - Filters and BEG
- Namespace: `org.ow2.aspirerfid`
- Licence: LGPL v2.1 and CC by-sa
Agenda

- OW2
- Component models: OW2 Fractal and OSGi
  - Component models in action
  - Component models benefits
Component Models Benefits

- A modular OW2 code base
- Software reuse
  - Separation of concerns (aspects)
  - Adaptation to user’s needs
- Ease of operation
  - Modularity and lightness: facilitates maintenance
  - Optimize resources consumption, system footprint
- Dynamic adaptation
OW2 component users

**Fractal**
- CLIF [http://clif.ow2.org](http://clif.ow2.org)
- Dream [http://dream.ow2.org](http://dream.ow2.org)
- Frascati [http://frascati.ow2.org](http://frascati.ow2.org)
- JASMINe [http://jasmine.ow2.org](http://jasmine.ow2.org)
- PEtALS [http://petals.ow2.org](http://petals.ow2.org)
- Speedo [http://speedo.ow2.org](http://speedo.ow2.org)

**OSGi**
- AspireRFID [http://aspire.ow2.org](http://aspire.ow2.org)
- Dysoweb [http://dysoweb.ow2.org](http://dysoweb.ow2.org)
- Easybeans [http://easybeans.ow2.org](http://easybeans.ow2.org)
- JASMINe [http://jasmine.ow2.org](http://jasmine.ow2.org)
- JOnAS [http://jonas.ow2.org](http://jonas.ow2.org)
- Talend [http://talend.ow2.org](http://talend.ow2.org)
Further directions

- Foster the use of component models in other OW2 projects

- Work on convergence between OSGi and Fractal
  - Define a language-independent Fractal deployment model
  - Use of OSGi in the Fractal deployment for Java
- Joint usage of Fractal, OSGi and iPojo to be investigated
Q&A
Enjoy the Technology!
... Join the Community!

Free beer on the OW2 booth at 4pm