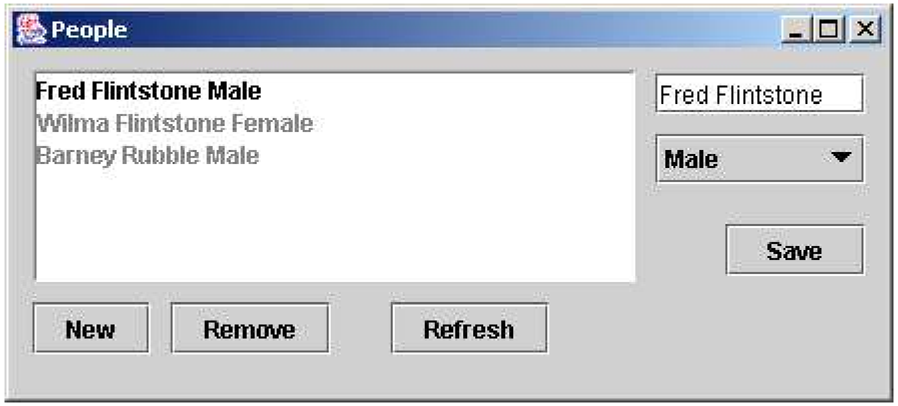
Persistence Neutrality using the Enterprise Object Broker application service framework

Erik Dörnenburg

JAOO 2003

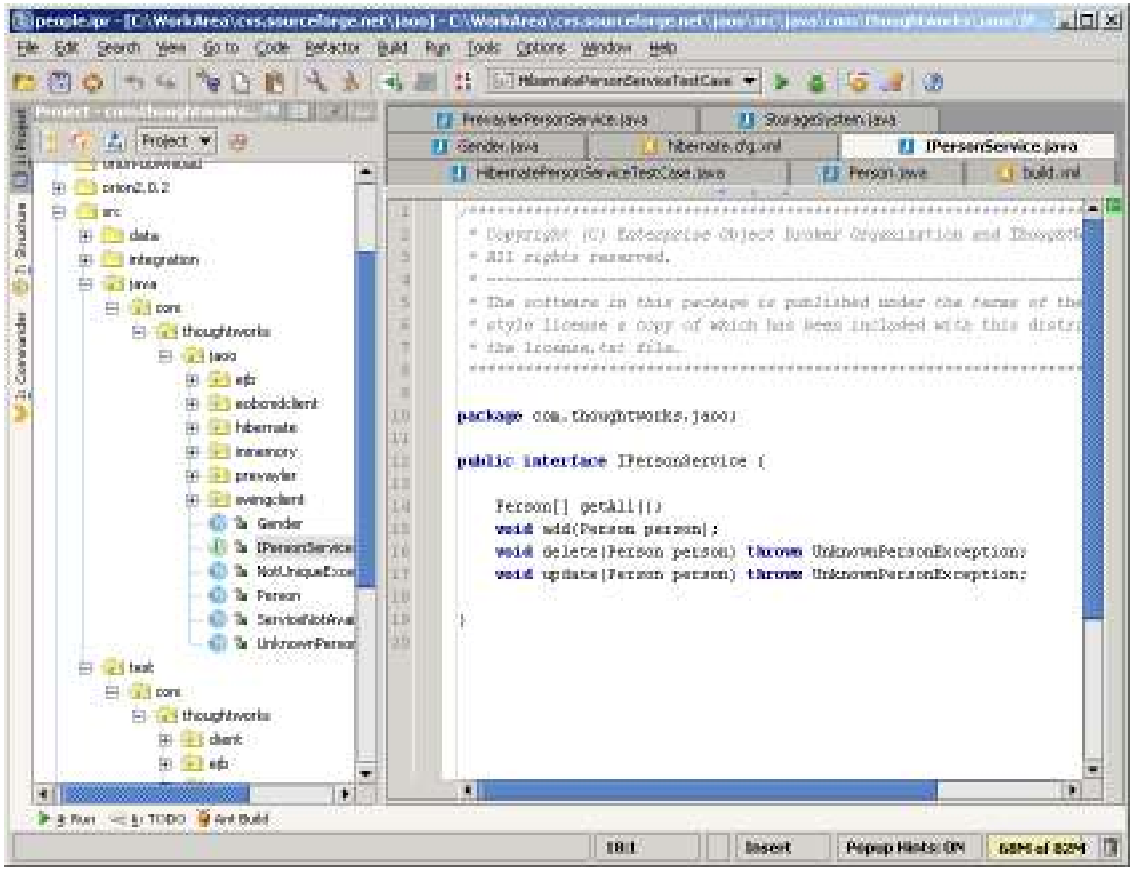
Sample project

- Simple application
- Heavy client
- One business entity
- Basic operations



- Person has **id**, **name**, **gender** with getters and setters as appropriate
- Gender is a custom value class
- PersonService has methods to retrieve all, as well as add, remove and update individual persons

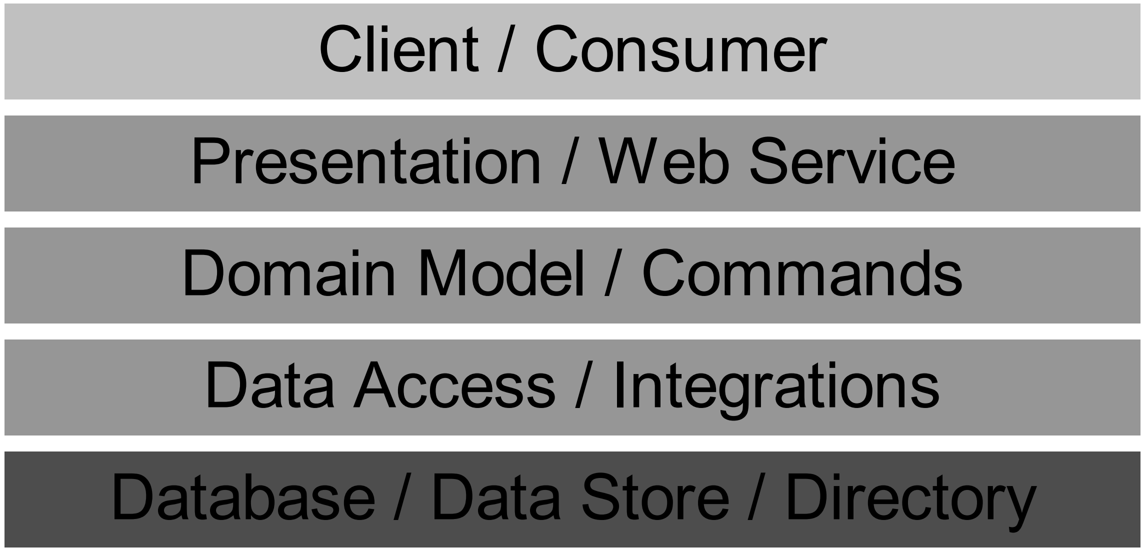
Demo – Explore interfaces in IntelliJ



```java
/*****************************************************************
 * Copyright (C) Enterprise Object Broker Organization and ThoughtW
 * All rights reserved.
 * ---------------------------------------------------------------
 * The software in this package is published under the terms of the
 * style license a copy of which has been included with this distri
 * the license.txt file.
 *****************************************************************


package com.thoughtworks.jaoo;

public interface IPersonService {

    Person[] getAll();
    void add(Person person);
    void delete(Person person) throws UnknownPersonException;
    void update(Person person) throws UnknownPersonException;

}
```

Enterprise Applications

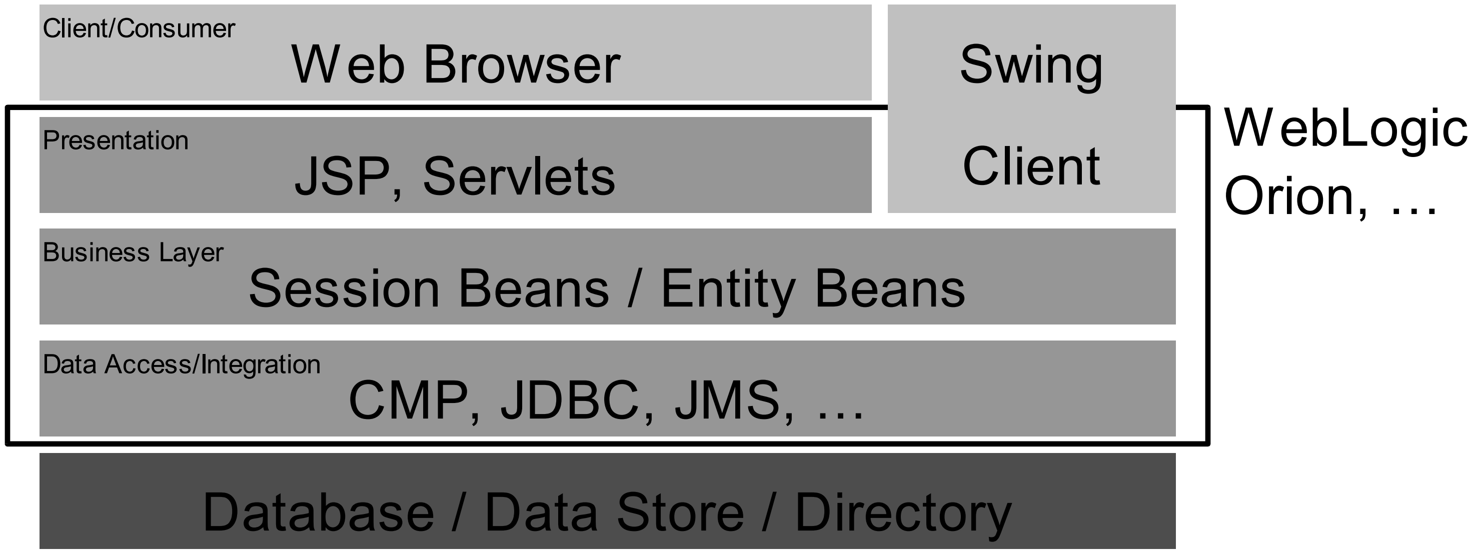
- Modern enterprise applications comprise multiple **layers** and **components**


Client / Consumer
Presentation / Web Service
Domain Model / Commands
Data Access / Integrations
Database / Data Store / Directory

- Technology preferences change: CORBA, EJB, …
- Patterns are explored and crystallised: Façade, IoC

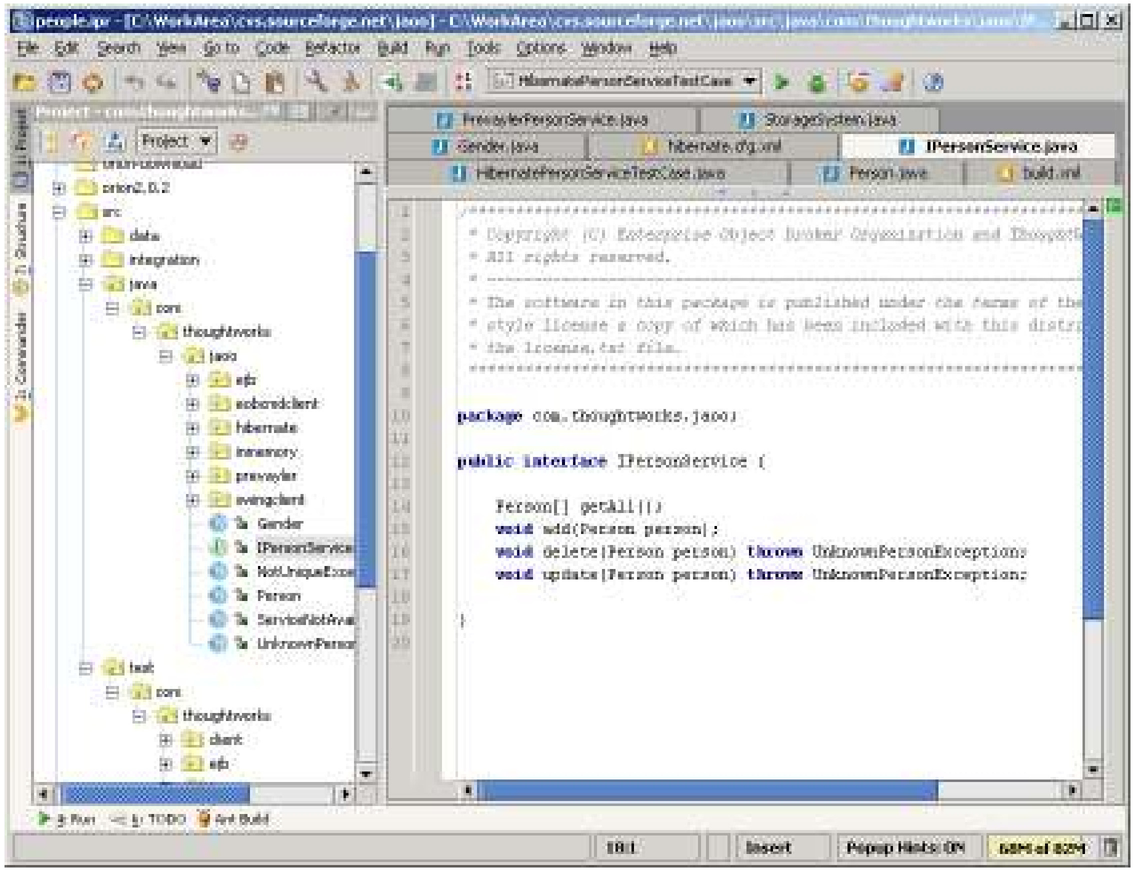
J2EE – Java 2 Enterprise Edition

- Sun standard application server framework for Java



- Application components hosted in containers
- Several complimentary technologies – Mix and match
- Application code tightly coupled to infrastructure

Demo – Explore EJB solution



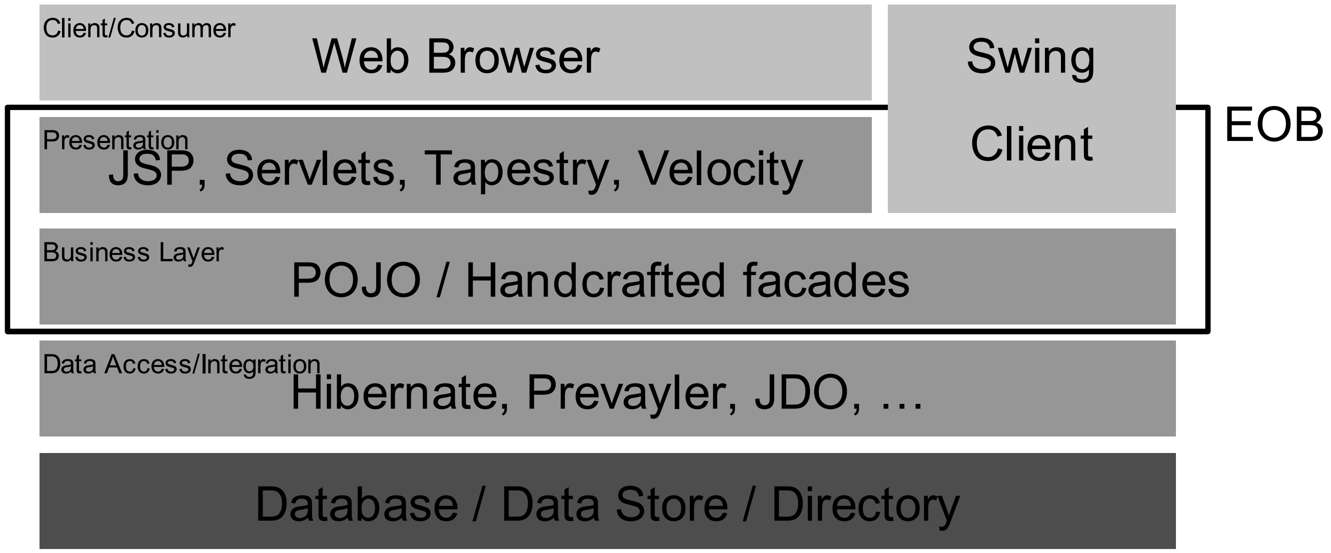
```
/***********************************************************************
 * Copyright (C) Enterprise Object Broker Organization and ThoughtW
 * All rights reserved.
 * ---------------------------------------------------------------------
 * The software in this package is published under the terms of the
 * style license a copy of which has been included with this distr
 * the license.txt file.
 ***********************************************************************

package com.thoughtworks.jaoo;

public interface IPersonService {

    Person[] getAll();
    void add(Person person);
    void delete(Person person) throws UnknownPersonException;
    void update(Person person) throws UnknownPersonException;

}
```

J2EE implementation

- Client side adapter from **PersonService** to Remote interface
- Adapter discovers server using JNDI
- **PersonService** implemented as stateless session bean on server
- Person idea implemented as **PersonBean**
- **PersonBean** is persisted using CMP
- Beans hosted in Orion application server

The Enterprise Object Broker approach

- Mix and match independently developed components


Client/Consumer
Web Browser
Swing Client
EOB
Presentation
JSP, Servlets, Tapestry, Velocity
Business Layer
POJO / Handcrafted facades
Data Access/Integration
Hibernate, Prevayler, JDO, …
Database / Data Store / Directory

- Core values: Transparency and Simplicity
- Utilises AltRMI for remote method invocation
- Application code *not* coupled to infrastructure

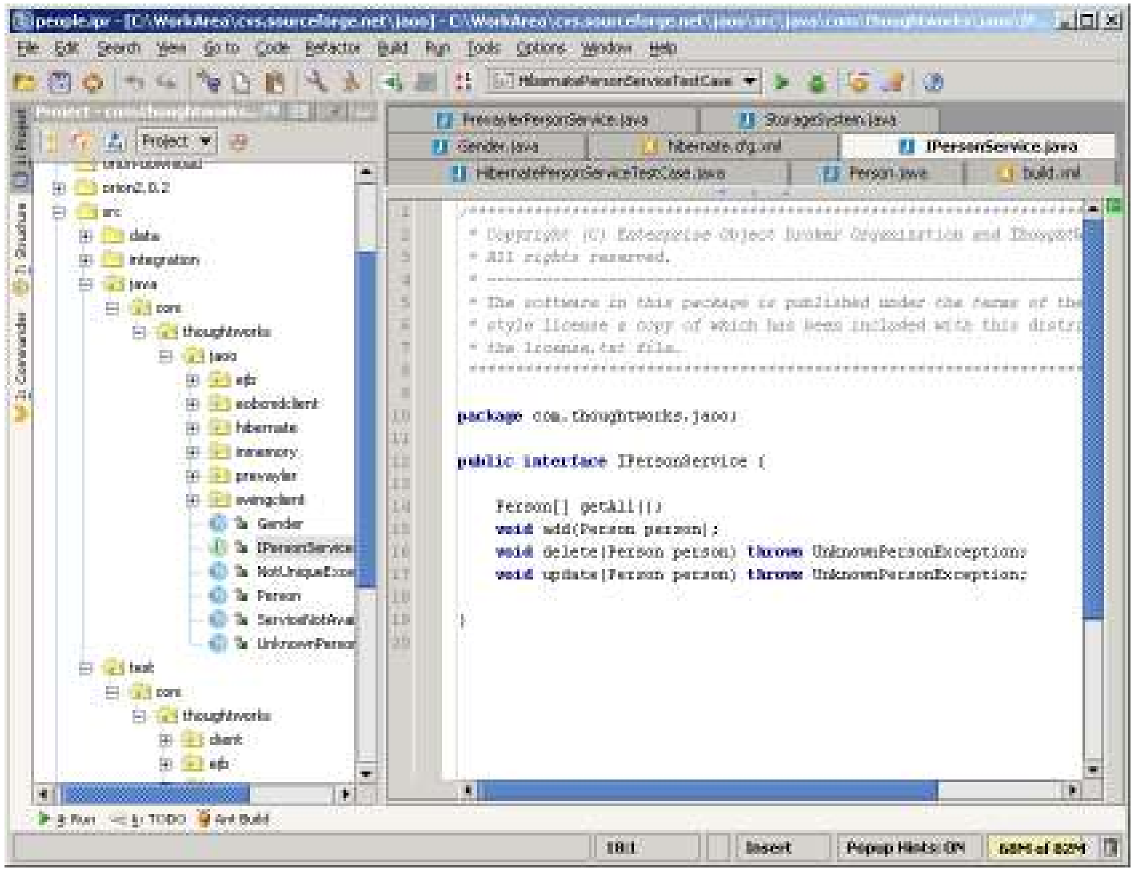
Enterprise Object Broker

- Application beans transparently published remotely
- Blurs differences between ‘Local’ and ‘Remote’ dependant component lookup
- Built-in services
  - Servlet Container (Jetty), facilitates JSP, Tapestry etc.
- Not coupled to an object level persistence service
  - CMP & BMP – no equivalent
  - Can use Hibernate, Prevayler, JDO, XML, serialization, etc.
- Most suitable for Intranet apps presently

AltRMI

- Transparent Remote Procedure Call technology
  - Does not throw **RemoteException** –
    uses **RuntimeException** derived instead
  - No extending of **UnicastRemoteObject**
  - No implementing of **Remote** in interfaces
  - Code with normal Java interfaces
- Multiple and pluggable choices for Transport
  - Native (TCP/pipes), RMI, etc.
  - Interop with SOAP, CORBA, DCOM planned
- In the Incubator at Apache

Demo – Explore Enterprise Object Broker



```
/*********************************************************************
 * Copyright (C) Enterprise Object Broker Organization and ThoughtW
 * All rights reserved.
 * -------------------------------------------------------------------
 * The software in this package is published under the terms of the
 * style license a copy of which has been included with this distri
 * the license.txt file.
 *********************************************************************

package com.thoughtworks.jaoo;

public interface IPersonService {

    Person[] getAll();
    void add(Person person);
    void delete(Person person) throws UnknownPersonException;
    void update(Person person) throws UnknownPersonException;

}
```

ThoughtWorks® The art of heavy lifting.℠

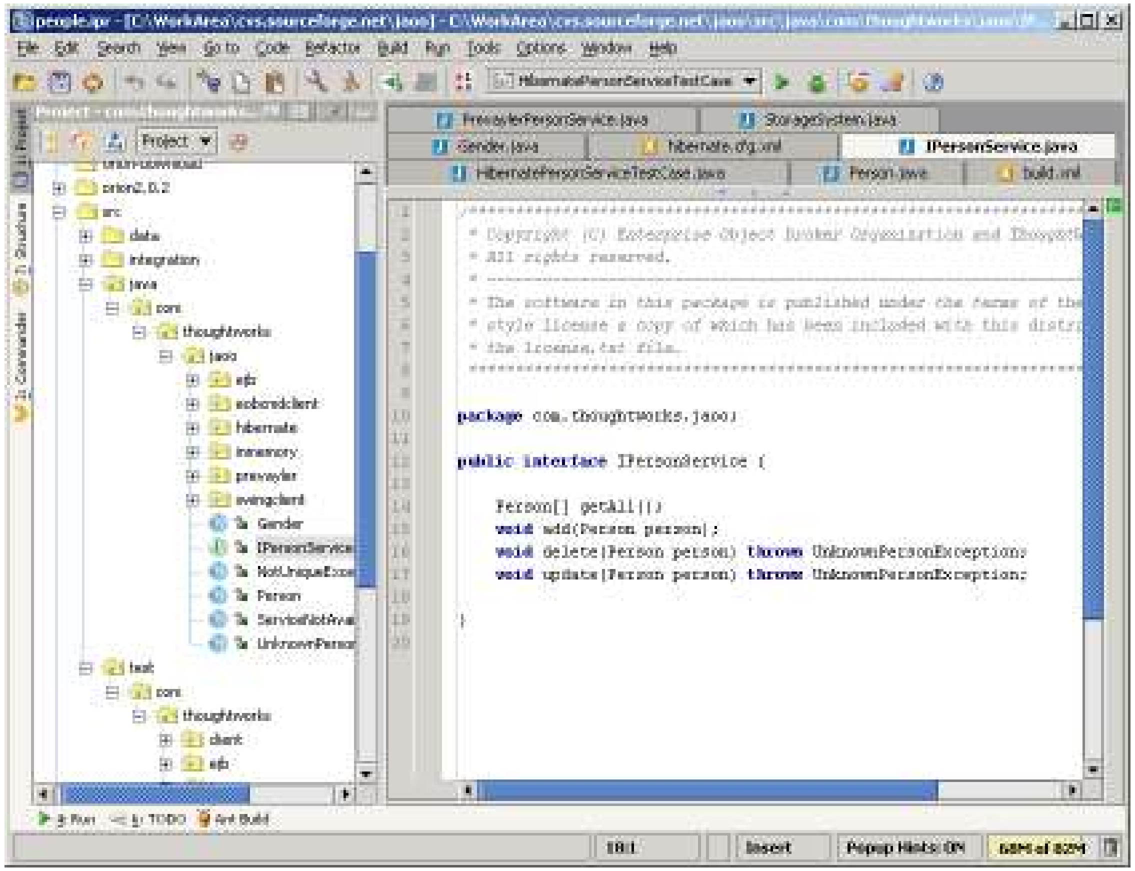
EOB implementation

- InMemoryPersonService
  - No persistence
  - Reference implementation
  - Gives GUI team a head start
  - Good for Unit,Integration and eyeball testing
- beans.xml
  - Describes the façade and its implementation
  - Bundled in JAR with classes
- application.xml
  - Describes the remote publication characteristics
  - Bundled in EOB file with JARs and WARs

Hibernate

- Object-relational persistence and query service
- Developers
  - Write Plain Old Java Objects (POJO)
  - Use idioms like association, inheritance, collection
  - Describe mapping to schema using XML file or Java Doc
- Query language is a 'minimal' OO extension to SQL
- Uses reflection and runtime bytecode generation
- Generates SQL at system startup time

Demo – Explore Hibernate Service



```
/********************************************************************
 * Copyright (C) Enterprise Object Broker Organization and ThoughtW
 * All rights reserved.
 * ------------------------------------------------------------------
 * The software in this package is published under the terms of the
 * style license a copy of which has been included with this distri
 * the license.txt file.
 ********************************************************************

package com.thoughtworks.jaoo;

public interface IPersonService {

    Person[] getAll();
    void add(Person person);
    void delete(Person person) throws UnknownPersonException;
    void update(Person person) throws UnknownPersonException;

}
```

Hibernate implementation

- **Person** is a Plain Old Java Object (POJO)
- No superclass or interface requirements, but
- **Person** must provide a default constructor and a (non-public) setter for its primary key.
- Mapping between the object's properties and database columns is described in an XML file
- **PersonService** *has a* HibernateSession: Composition
- **PersonService** is a lightweight adaptor between app specific requirements and general persistence service

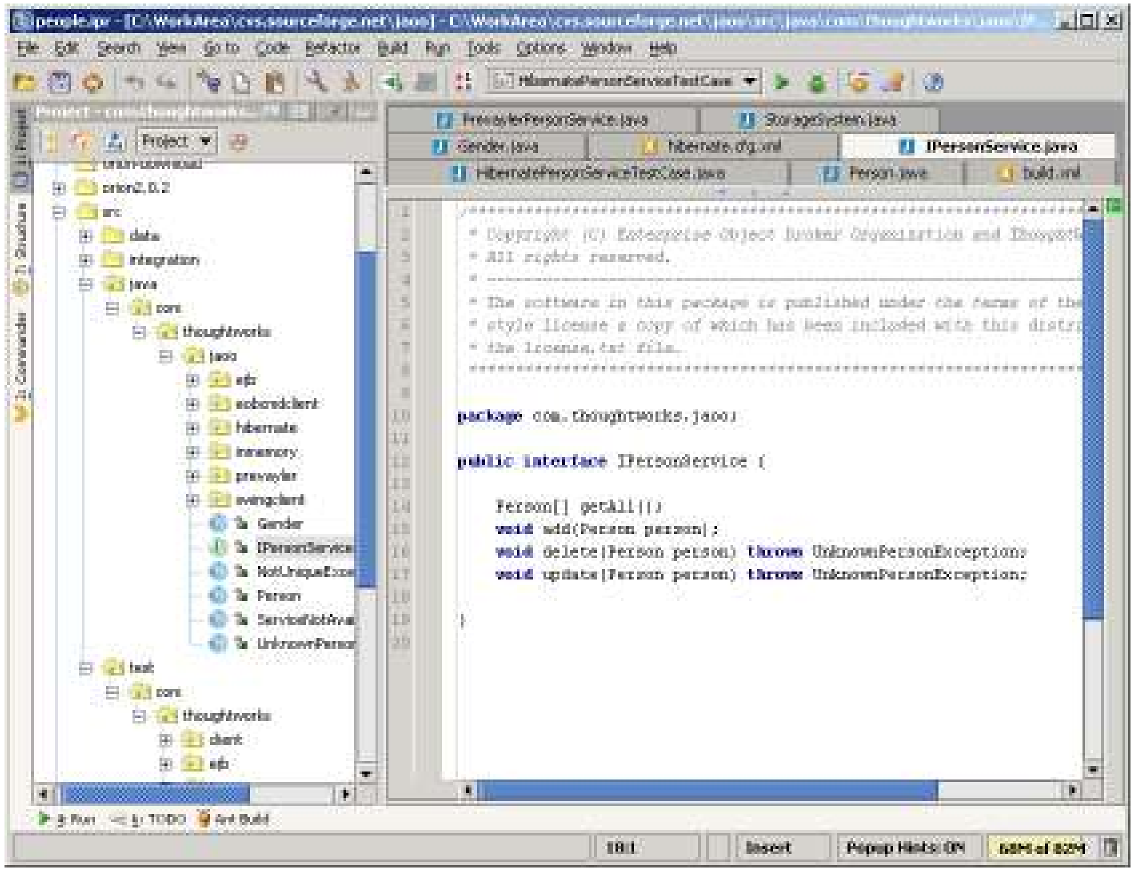
Prevayler

- Persistence without a relational database
- Developer
  - Writes Plain Old Java Objects (POJO)
  - Uses association and other Java idioms
  - Expresses transactions in command objects
  - Does not have access to a set-based query language
- Prevayler
  - Keeps the entire set of 'business objects' is in memory
  - Takes snapshots from time to time
  - Records all commands in a separate log

Prevayler (cnt'd)

- Programming model
  - Queries directly against business objects
  - All modifications to business objects through commands
  - All commands are serialised
- Prevayler performance
  - Queries: orders of magnitude compared to relational databases; even if these have the database in RAM
  - Modifications: on par with comparable database
  - Snapshots: 10 000 objects per second
  - Recovery: 5000 commands per second
  - Uses a replica for writing snapshots

Demo – Explore Prevayler Service



```
/*****************************************************************
 * Copyright (C) Enterprise Object Broker Organization and ThoughtW
 * All rights reserved.
 * ---------------------------------------------------------------
 * The software in this package is published under the terms of the
 * style license a copy of which has been included with this distri
 * the license.txt file.
 *****************************************************************

package com.thoughtworks.jaoo;

public interface IPersonService {

    Person[] getAll();
    void add(Person person);
    void delete(Person person) throws UnknownPersonException;
    void update(Person person) throws UnknownPersonException;

}
```

Prevayler implementation

- **Person** is a Plain Old Java Object (POJO)
- Must be **Serializable**
- No further requirements, not even an id
- Requires a 'prevalent' system to hold the objects
- Uses classes to express transaction on the system
- **PersonService** implemented as a façade which creates instances of commands and executes these on the system.

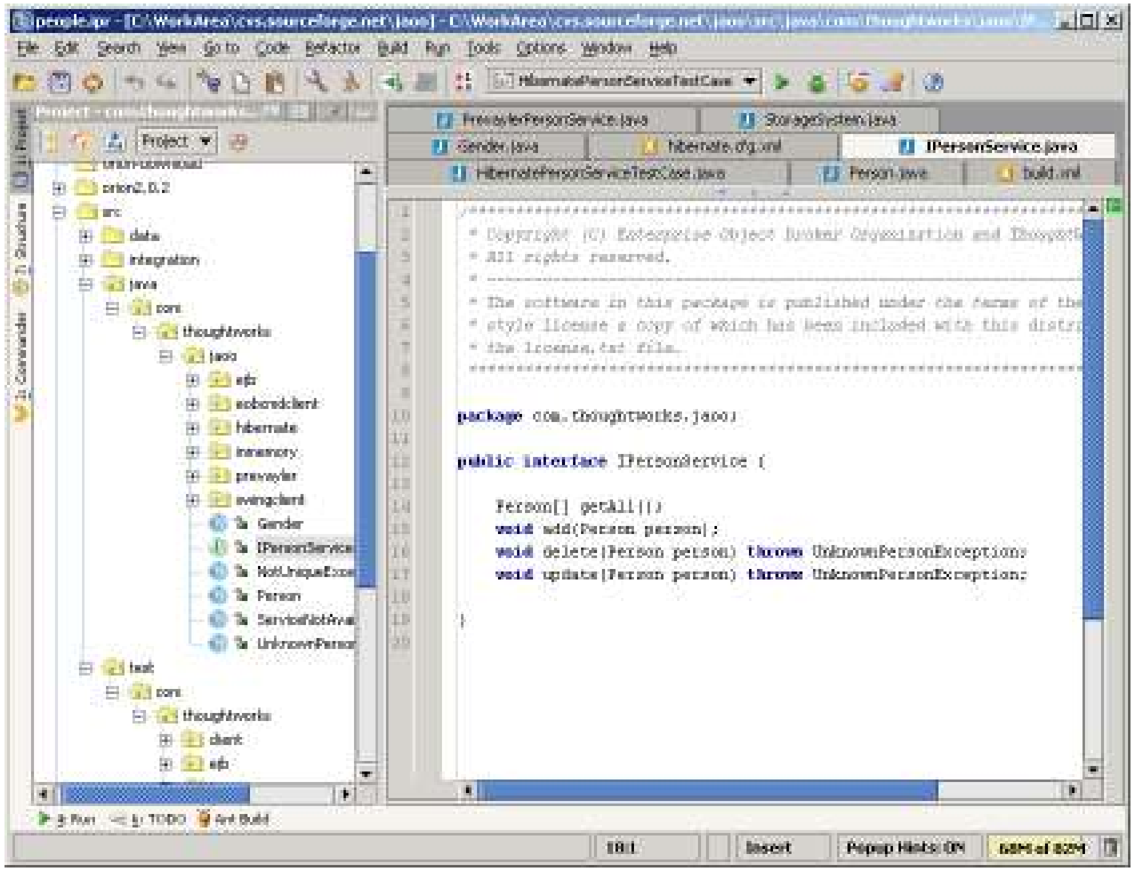
Test-Driven Development & Continuous Integration

- Test-Driven Development (TDD) focuses on the *consumer* view on an object.
- The developer concentrates on the responsibilities of a class by writing unit tests that fail.
- Tests drive the development of the *producer* aspect, the actual implementation of the object.
- Continuous integration means that all tests, unit tests and integration tests, are run on every check-in.
- CI therefore requires very fast test suites; something that can only be achieved by minimising DB hits.

Stubbed implementations & Mock Objects

- Mock objects and stubs are used to
  - simulate interaction
  - help explore responsibilities/interfaces
  - test conditions which are difficult to simulate
  - substitute long-running implementations with dummies
- Stubs are handcrafted implementations of existing APIs that return hard-coded values.
- Mock objects also verify usage of the API through expectations set by the application developer.
- Dynamic Mocks are created automatically at run time.

Live demo – Explore stubbed services



```
/*****************************************************************
 * Copyright (C) Enterprise Object Broker Organization and Thought
 * All rights reserved.
 * ---------------------------------------------------------------
 * The software in this package is published under the terms of the
 * style license a copy of which has been included with this distr
 * the license.txt file.
 *****************************************************************


package com.thoughtworks.jaoo;

public interface IPersonService {

    Person[] getAll();
    void add(Person person);
    void delete(Person person) throws UnknownPersonException;
    void update(Person person) throws UnknownPersonException;

}
```

Summary

- J2EE and EJB require the application developer to add infrastructure code to the domain model and commands.
- Conventional O/R mappers as well as new approaches to persisting objects are constantly evolving.
- Enterprise Object Broker is a lightweight container for application components.
- EOB and AltRMI allow transparent access to the application components.
- Application specific interfaces decouple objects, allowing for substitution and mocking.

Links

Enterprise Object Broker

www.enterpriseobjectbroker.org

Sourcecode for the demonstrations, via cvs

:pserver:anonymous@cvs.sourceforge.net:/cvsroot/eob (module: jaoo)

Hibernate

www.hibernate.org

Prevayler

www.prevayler.org

Mock Objects

www.mockobjects.com

ThoughtWorks

www.thoughtworks.com