MOVE-12: OpenEdge®, ADOBE FLEX, and OpenEdge Reference Architecture

A Look at a Transformation

Alvacir Reinaldo Schulze

Flávio Eduardo de Córdova

PROGRESS SOFTWARE Exchange 2006

Agenda

- What is Datasul ?
- A Brief Overview Of Datasul Webdesk
- Main Features of New Release
- Main Challenges
- Main Goals of New Architecture
- Elements of New Architecture
- Sample Code
- Adobe Flex
- Integrating Flex and OpenEdge
- Webdesk BPA
- Lessons Learned
- Questions & Answers

What is Datasul ?

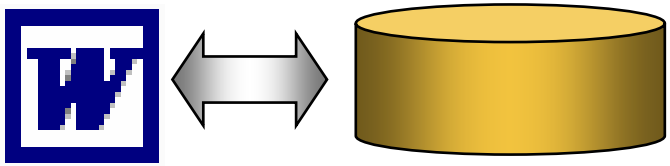
- Largest Progress ISV in Latin America
- +60 Thousand Modules Installed
- 80.000 users
- More than 120 Million Source Code Lines
- More than 680 developers
- US$ 10 Millions invested in Product Development in 2005

A Brief Overview Of Datasul Webdesk

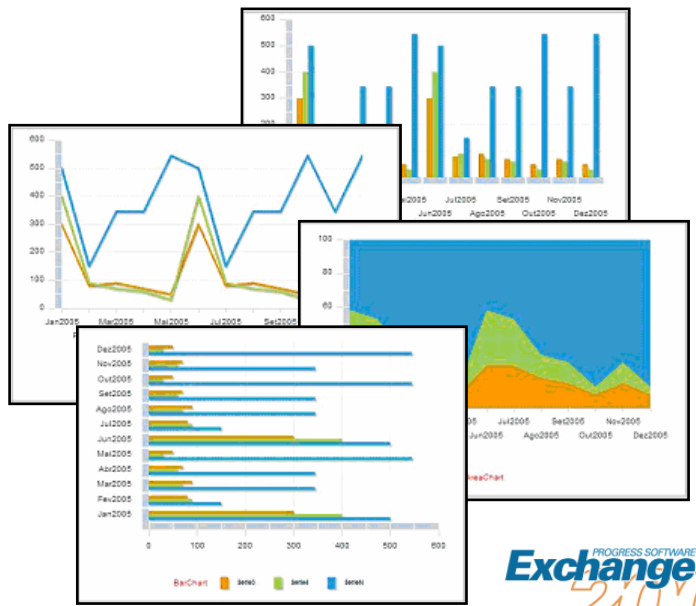
- Enterprise Content Manager
- Workflow Engine
- Portal Manager
- Progress® WebSpeed®
- Java™ and Microsoft Technologies Integration

Main Features Of New Release

- WebDAV interface (Java)
- Business Process Analysis (Flex)



Webdesk Repository



Main Challenges

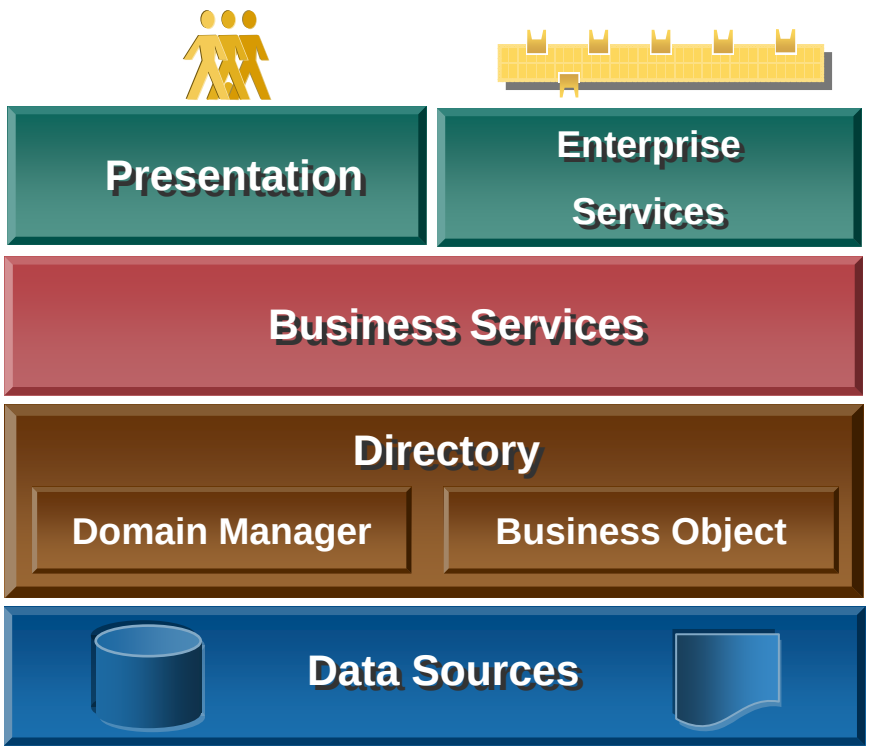
- Reduce Duplicated Code
- Reduce New Features Costs
- Integration to Java
- Rich User Interface

Main Goals Of New Architecture

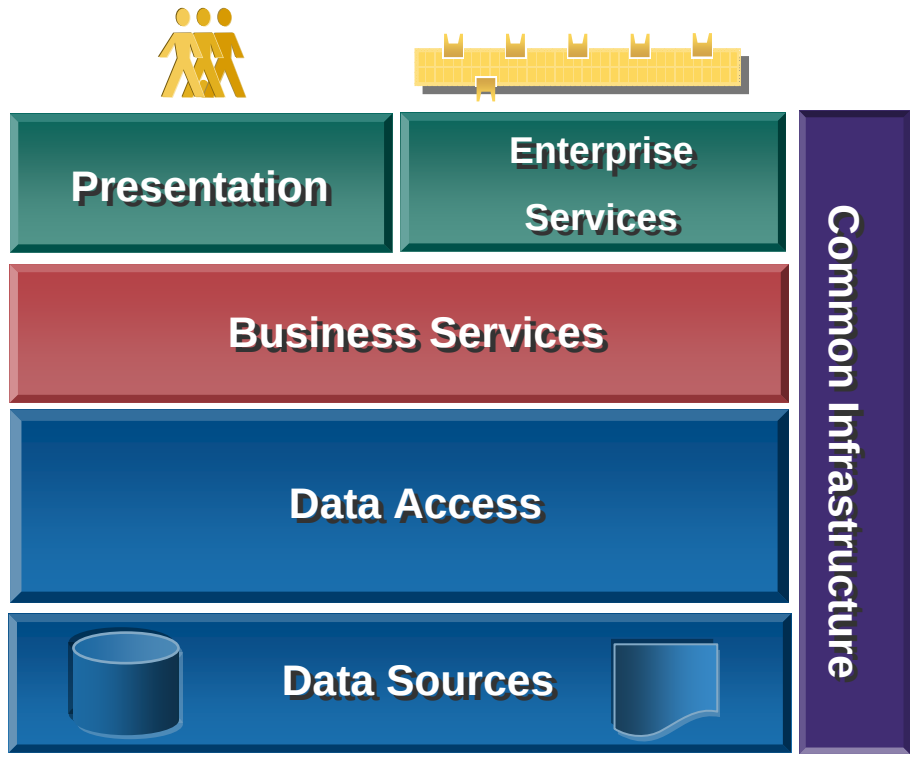
- Separate Business Logic and Interface
- Lose Coupling Between Layers
- Enable Heterogeneous Teams
- Adoption of Object-Oriented concepts
- Enable Better Testing Practices
- Enable Continuous Integration

Elements Of New Architecture

Used Architecture



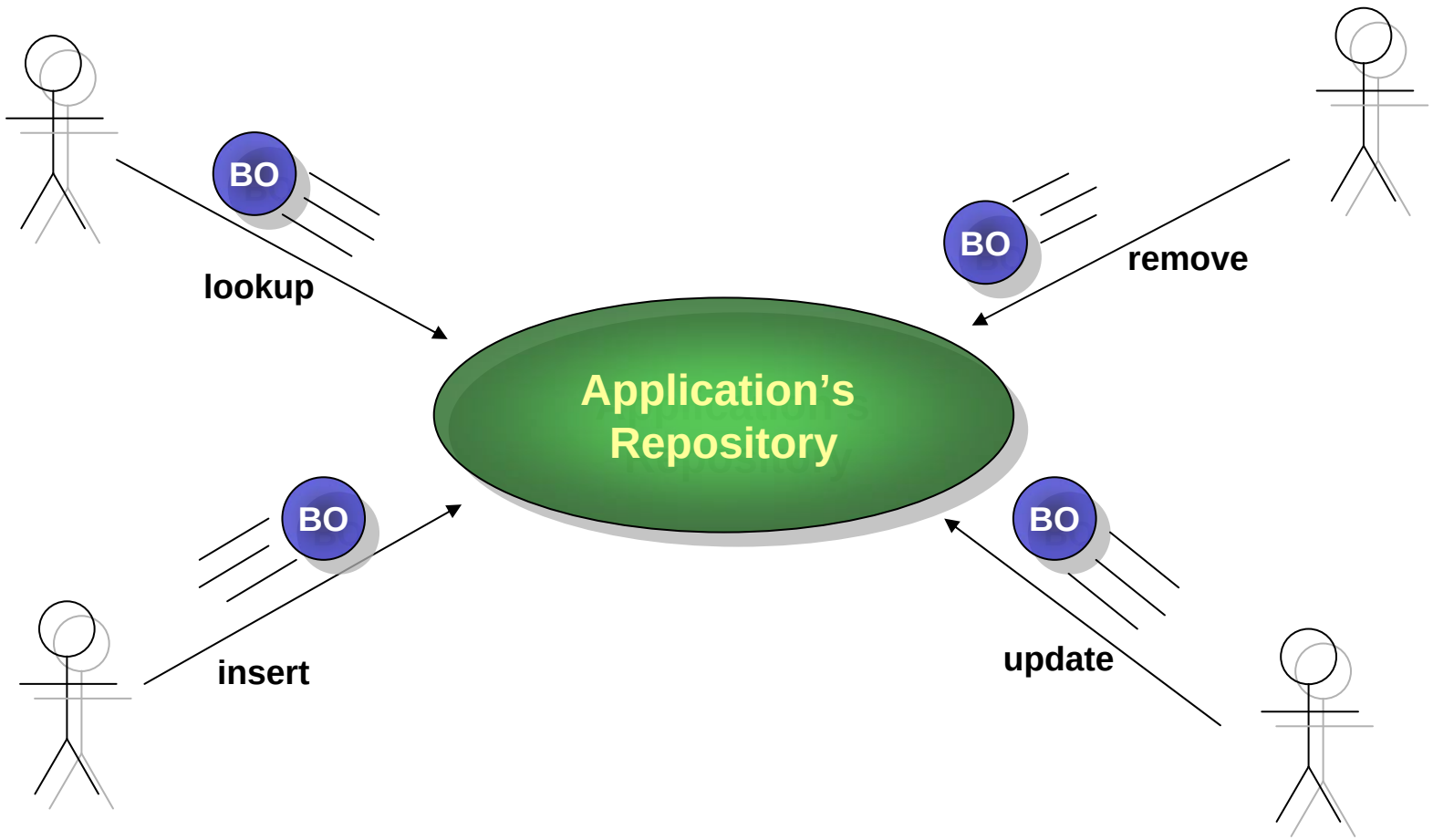
Open Edge Reference Architecture



Elements Of New Architecture Directory

- Represents the application's repository where all objects are stored
- Manages all CRUD operations
- Hides Business Objects' implementation from the client (using logical names to find them)
- Optimizes queries in database

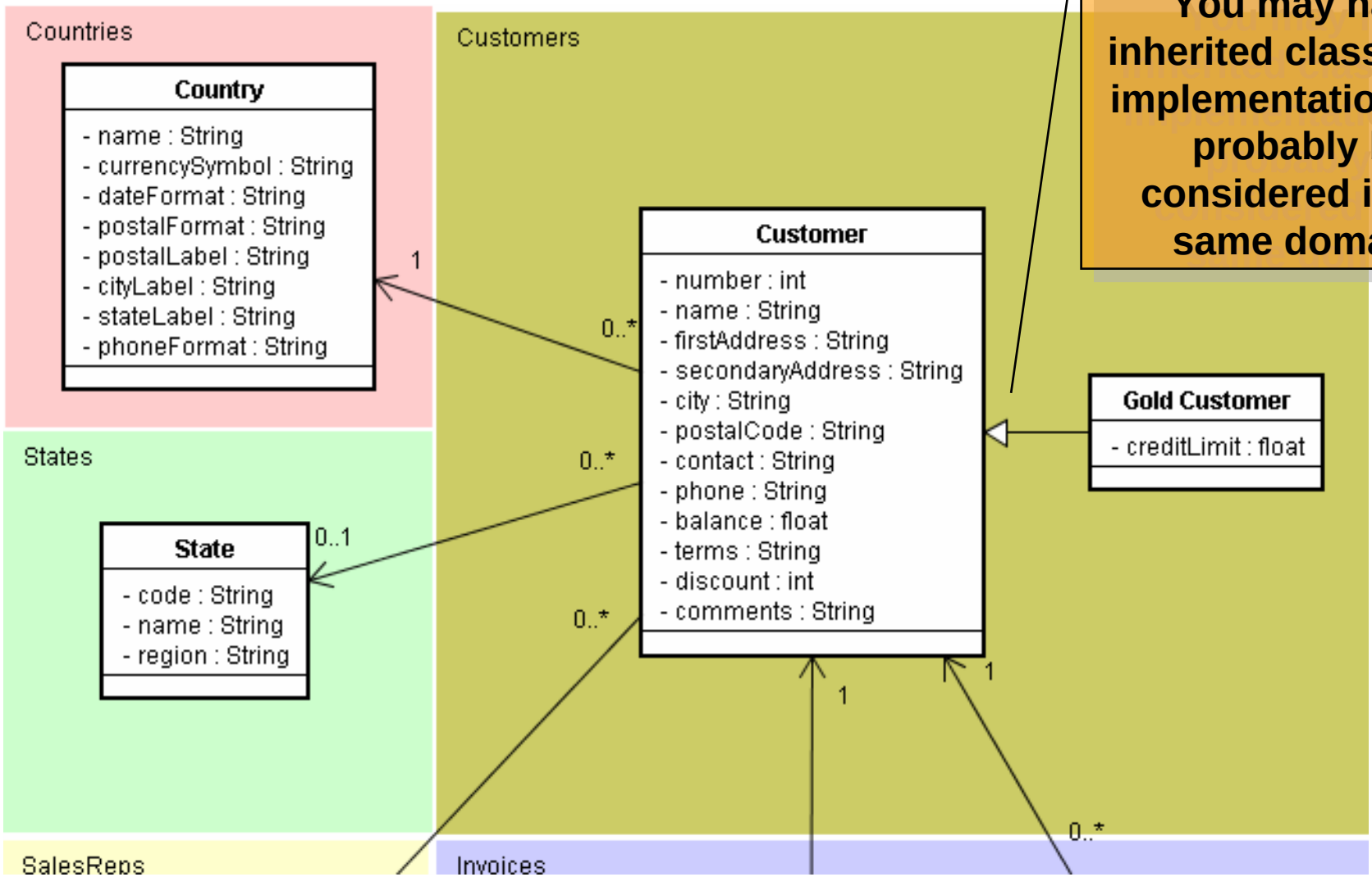
Elements Of New Architecture Directory



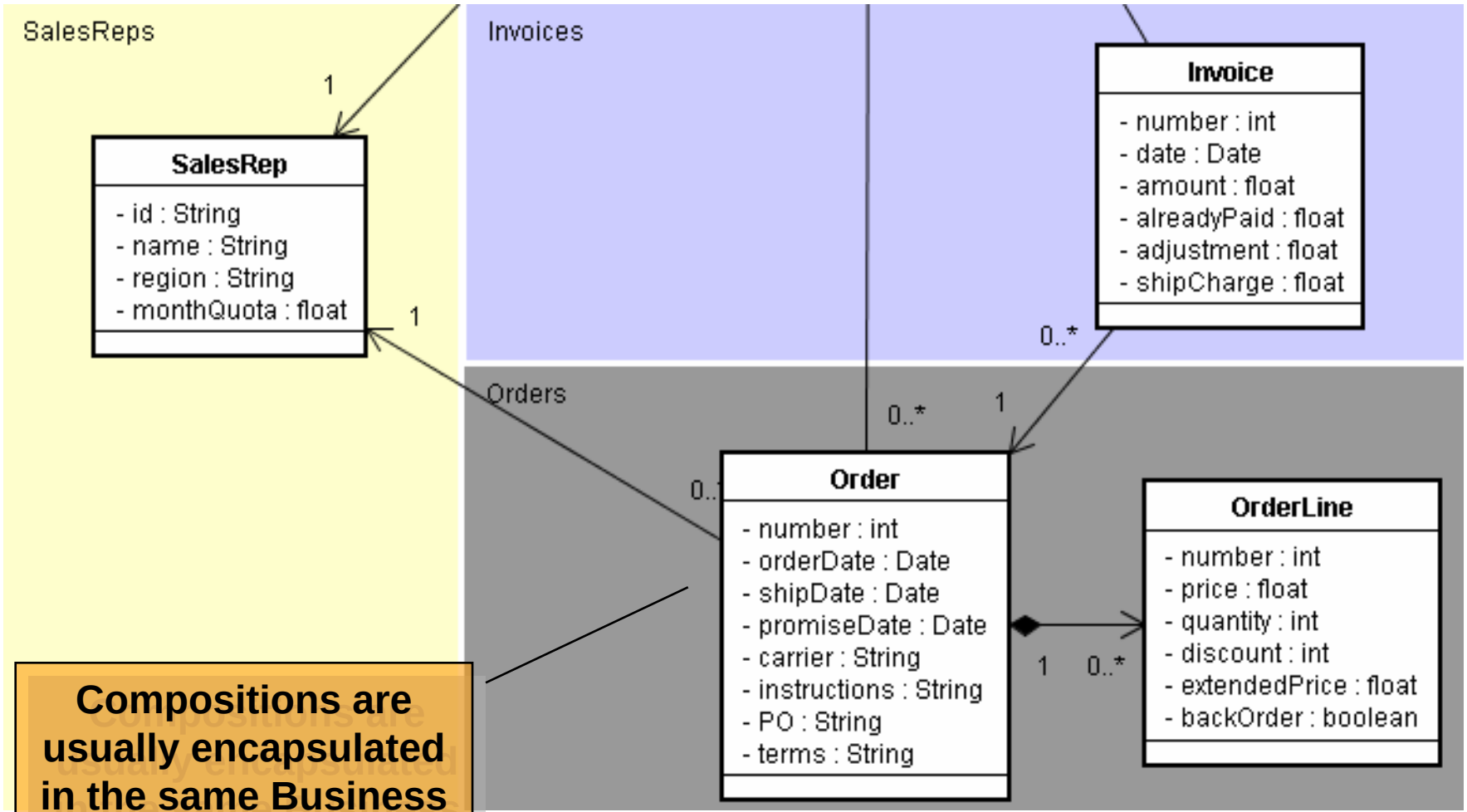
Elements Of New Architecture
Domain Manager

- Publishes what type of business object are available to the directory
- Gives directory meta-information of how to look for objects (table name, field mapping, query constraints) when they are persisted in the database
- Initializes business objects before they are made available to clients

Elements Of New Architecture
Domain Manager



Elements Of New Architecture
Domain Manager



Compositions are usually encapsulated in the same Business Object.

Elements Of New Architecture
Business Object

- Encapsulates business logic of a specific entity
- Usually access database to persist data (insert, update and remove), even though may use DAOs if needed.
- May encapsulate more than one entity (main entity plus compositions)
- Provides accessor methods to entity's attributes
- Provides a well-defined interface to be implemented by the developer.

Elements Of New Architecture
The Service Façade

- Exposes Business Object's logic as services to user interface or to external applications
- Do not encapsulate any business logic or database access. Only make use of Business Objects to complete a specific task.
- May be accessed through different technologies (Java Proxies, .Net™ Proxies, WebServices, Local 4GL, AppServer™ 4GL, etc.).
- Usually stateless

Sample Code
Looking for objects

```
DEFINE TEMP-TABLE ttReferences
    FIELD namingRef    AS CHARACTER.

hDirectory = instantiateDirectory().

RUN lookup IN hDirectory
    (INPUT "webdesk://localhost/users/*?Active=true",
     INPUT-OUTPUT TABLE ttReferences).
FIND FIRST ttReferences.

RUN getBO IN hDirectory
     (INPUT BUFFER ttReferences:HANDLE,
      OUTPUT hBO).
RUN getName IN hBO (OUTPUT cName).

releaseDirectory().
```

Get reference to the Directory

Search in the directory for objects in the "users" domain

Get the BO instance

Call BO's method

Sample Code
Inserting Objects

```
hBO = getBOById("Order").

RUN setCustomer IN hBO (10);
RUN addItem ("Item 1", 123.20).
RUN addItem ("Item 2", 33.00).

hDirectory = instantiateDirectory().

RUN insert IN hDirectory (INPUT hBO).

releaseDirectory().
```

Instantiate a "blank" Order

Set new Order's attributes.

Insert new object in the repository.

Sample Code
Updating Objects

```
DEFINE TEMP-TABLE ttReferences
    FIELD namingRef    AS CHARACTER.

hDirectory = instantiateDirectory().
RUN lookup IN hDirectory
  (INPUT “webdesk://localhost/orders/*?Number=12345”,
   INPUT-OUTPUT TABLE ttReferences).
FIND FIRST ttReferences.
RUN getBO IN hDirectory
  (INPUT BUFFER ttReferences:HANDLE,OUTPUT hBO).

RUN addItem IN hBO (“Item 3”, 1.99).
RUN addItem IN hBO (“Item 4”, 199.00).

RUN update  IN hDirectory (INPUT hBO).

releaseDirectory().
```

Change Order’s attributes (in this case, adding new Items to an existing Order).

Update object in the repository.

Sample Code
Removing Objects

```
DEFINE TEMP-TABLE ttReferences
    FIELD namingRef    AS CHARACTER.

hDirectory = instantiateDirectory().
RUN lookup IN hDirectory
    (INPUT
  “webdesk://localhost/orders/*?Number=12345”,
     INPUT-OUTPUT TABLE ttReferences).
FIND FIRST ttReferences.
RUN getBO IN hDirectory
(INPUT BUFFER ttReferences:HANDLE, OUTPUT hBO).

RUN remove  IN hDirectory (INPUT hBO).

releaseDirectory().
```

Remove object from the repository.

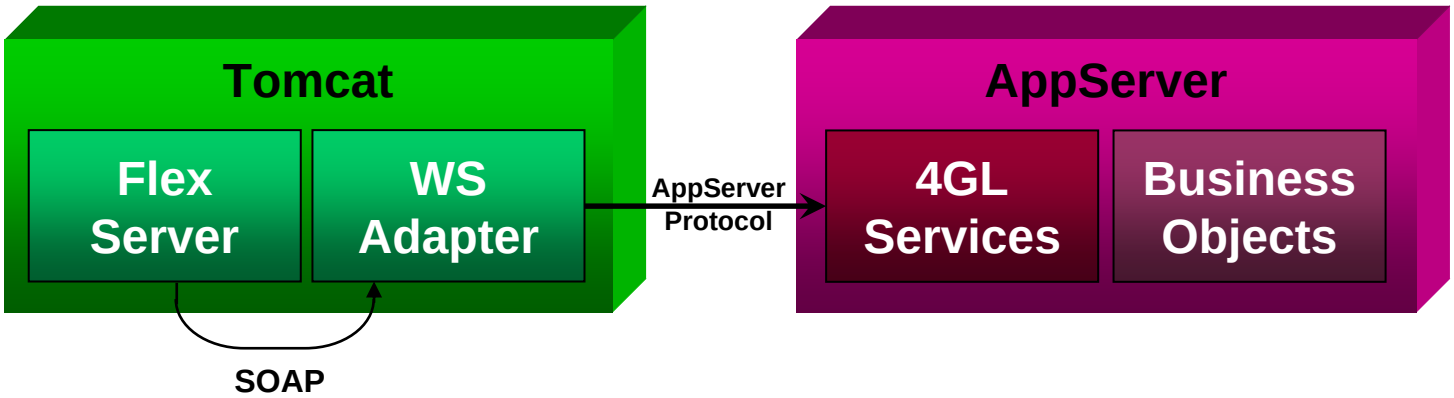
Adobe Flex

http://www.macromedia.com/software/flex

- Adobe's platform for Rich Interface Applications development
- Flash components running on the client
- Java classes running on a Web Container
- Web-service enabled
- Easy integration between Java classes and ActionScript
- Multiplatform

Integrating Flex and OpenEdge

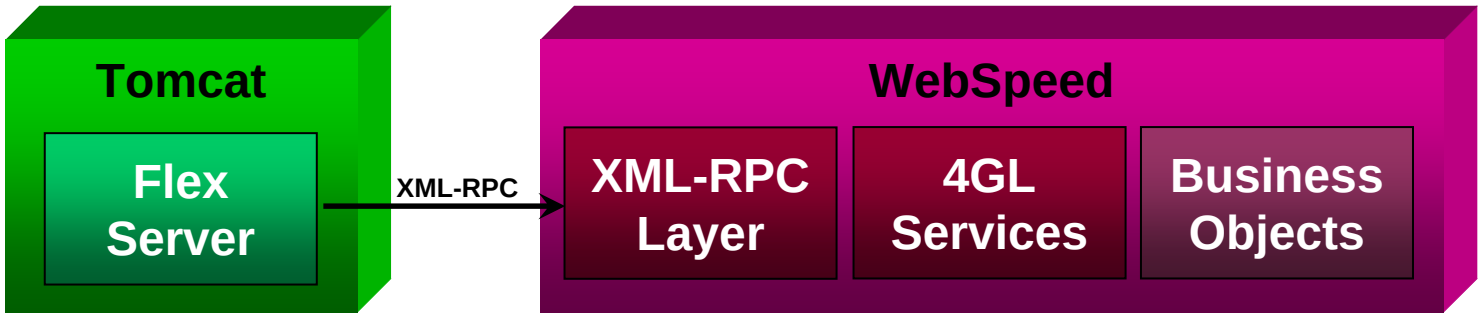
Use of Web Services



- ☺ Ready-to-use
- ☹ Requires AppServer configured and running
- ☹ XML packing/unpacking overhead

Integrating Flex and OpenEdge

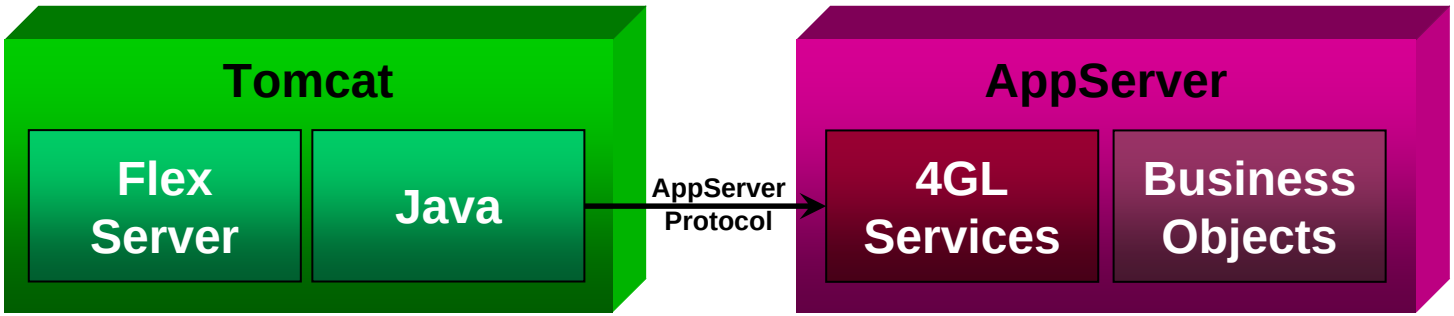
Use of XML-RPC



- ☹ Require Implementing XML-RPC server (already implemented)
- ☹ Requires WebSpeed
- ☹ XML packing/unpacking overhead (less than SOAP)

Integrating Flex and OpenEdge

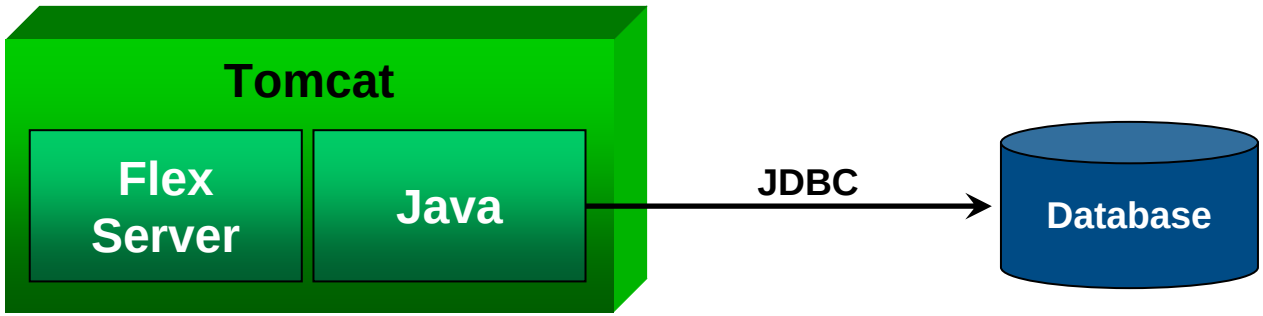
Use of Java Classes and AppServer



- ☹ Requires Java classes to get data from AppServer and serve Flex
- ☹ Requires AppServer configured and running
- ☺ Good performance

Integrating Flex and OpenEdge

Use of Java Classes and JDBC

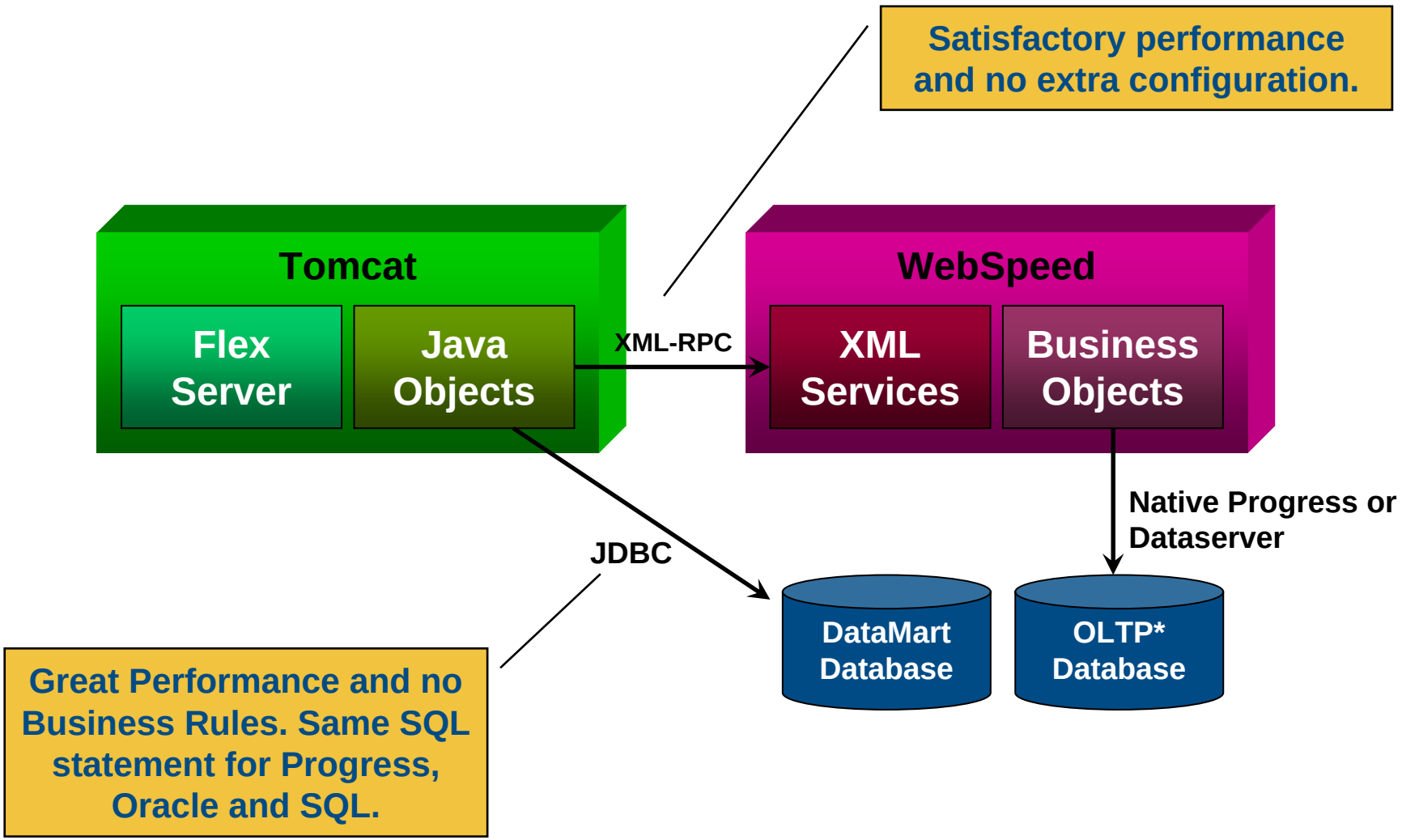


- ☹ Requires Java classes to get data from AppServer and serve Flex
- ☹ “Duplicated” code in Java and 4GL
- ☺ Best performance

Webdesk BPA

- On Webdesk BPA, Flex layer has to access two different types of data: Meta information about available indicators (about 100 records) and Data Mart database (potentially millions of records).
- WebSpeed is already configured and running (actually, most of the application runs under WebSpeed)
- AppServer is not required today (unless some other features are enabled)

Webdesk BPA


Satisfactory performance and no extra configuration.
Tomcat
Flex Server
Java Objects
XML-RPC
WebSpeed
XML Services
Business Objects
Native Progress or Dataserver
JDBC
DataMart Database
OLTP* Database
Great Performance and no Business Rules. Same SQL statement for Progress, Oracle and SQL.

Lessons Learned
Moving to OpenEdge

- Automate Tests on Business Layer
- Continuously Integrate
- Pay attention on Design
- Concentrate on the Interfaces
- Always consider Refactoring

Lessons Learned Using Adobe Flex

- Asynchronous Programming Model tricks developers
- [Try to] Avoid “small code replications” temptation
- Don’t forget it should be Rich from user’s perspective
- Be careful of how much information you show
- Combine Flex Technology to Ergonomics Best Practices
- Consider Waiting for Flex 2.0

Live Demo

Q & A

Thank You