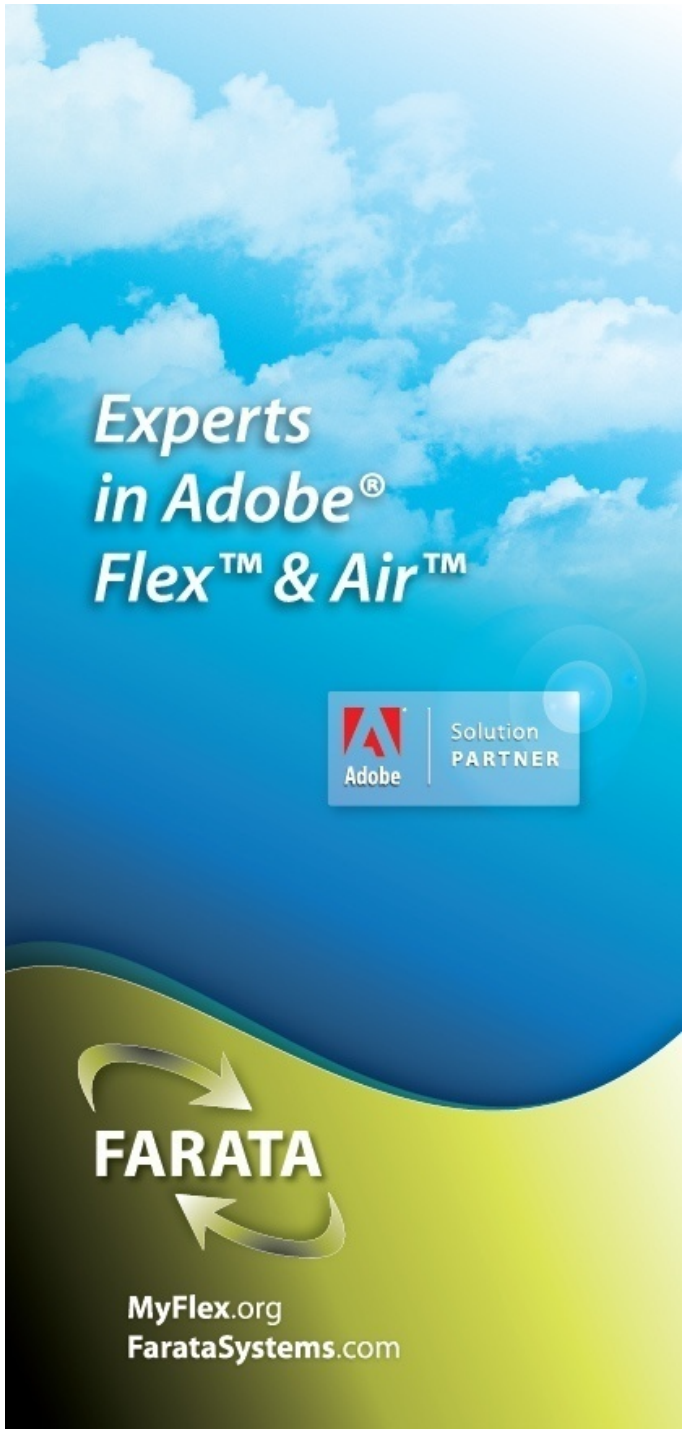
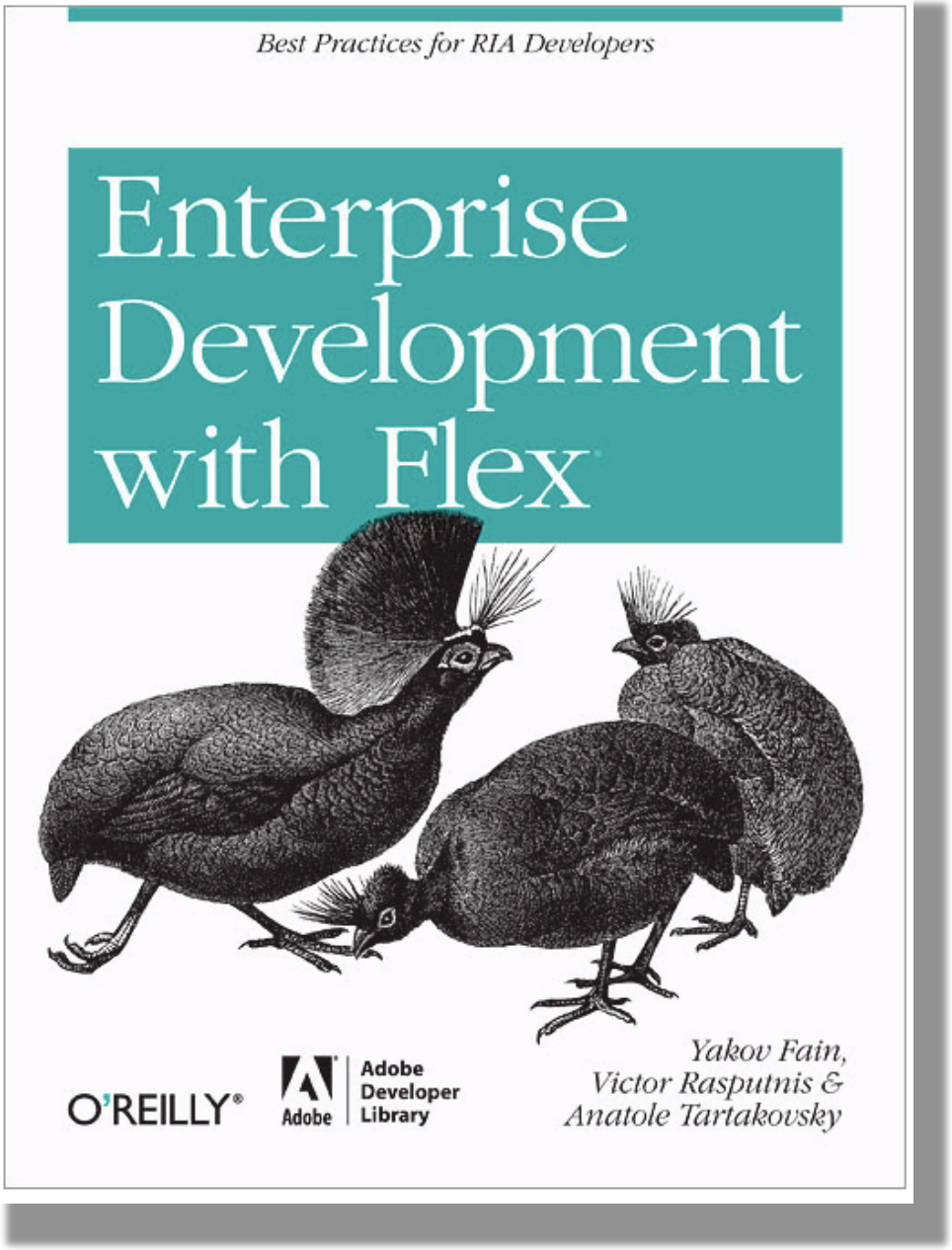


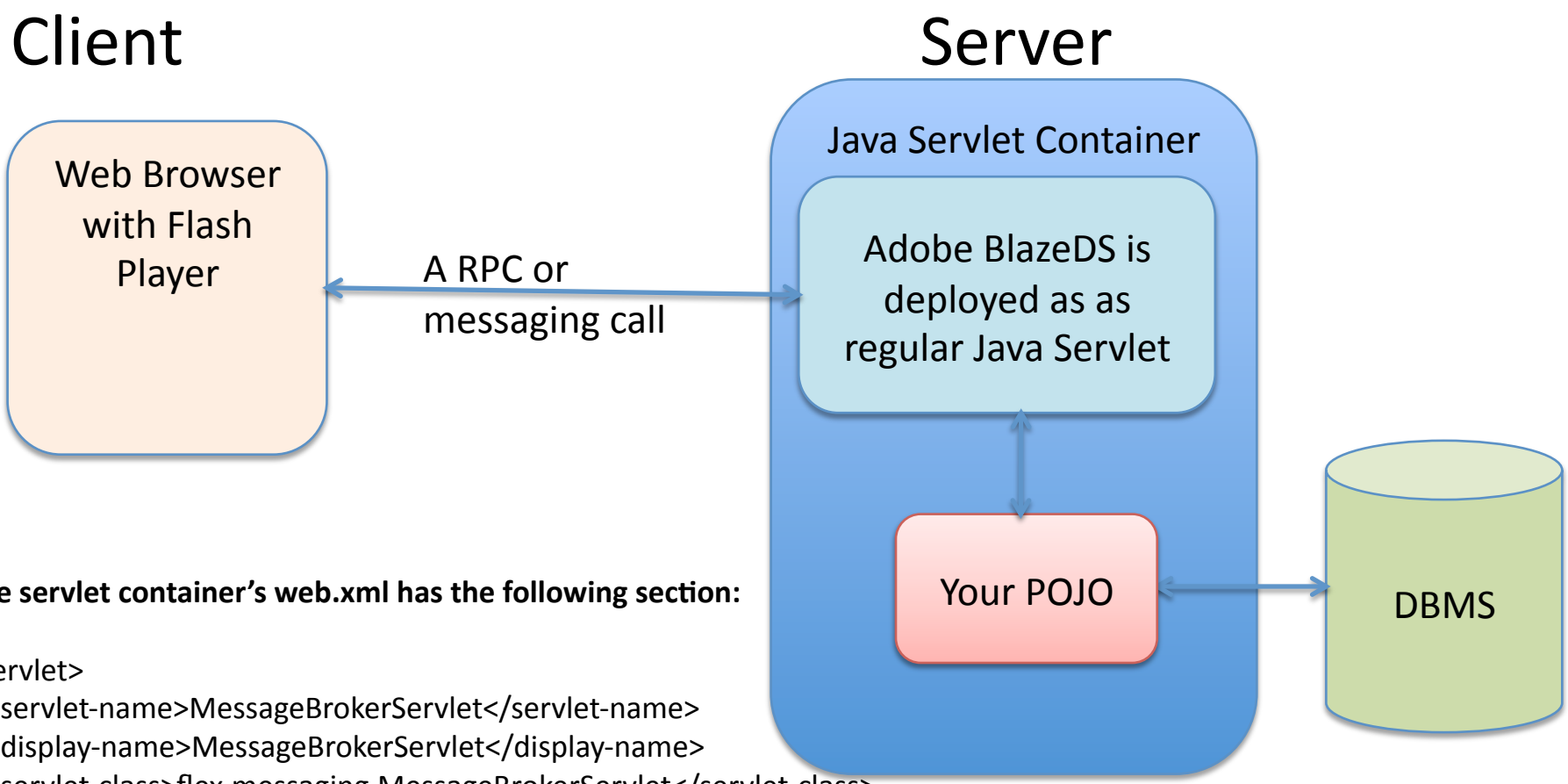
Bringing together Flex, Java, and DBMS

Download this slide deck form
http://myflex.org/presentations/FlexJavaDBMS.pdf



While this is a 101-level presentation, you can find a lot of advanced Flex/Java materials in our latest book "Enterprise Development with Flex" from the series Adobe Developer Library

Flex/BlazeDS/DBMS communications



The servlet container's web.xml has the following section:

```
<servlet>
  <servlet-name>MessageBrokerServlet</servlet-name>
  <display-name>MessageBrokerServlet</display-name>
  <servlet-class>flex.messaging.MessageBrokerServlet</servlet-class>

  <init-param> <param-name>services.configuration.file</param-name>
   <param-value>/WEB-INF/flex/services-config.xml</param-value>
  </init-param>
</servlet>
```

Flex and JSP communication with HTTPService object
Problem: all the data arrive with the type String

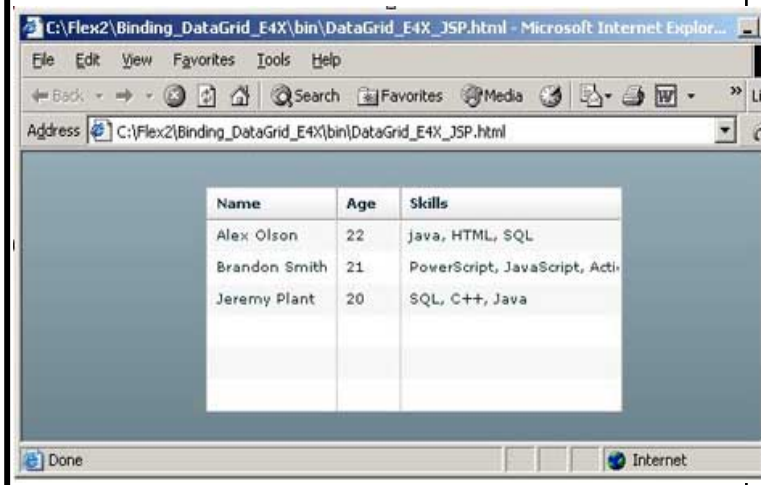
XML

```
<people>
 <person>
    <name>Alex Olson</name>
    <age>22</age>
    <skills>java, HTML,    SQL</skills>
 </person>
</people>
```

JSP

```
<% out.println("<?xml version=\"1.0\" encoding=\"UTF-8\"?
><people><person><name>Alex Olson</name><age>22</
age><skills>java, HTML, SQL</skills></person><person><name>Brandon
Smith</name><age>21</age><skills>PowerScript, JavaScript,
ActionScript</skills></person><person><name>Jeremy Plant</
name><age>20</age><skills>SQL, C++, Java</skills></person></
people>"); %>
```

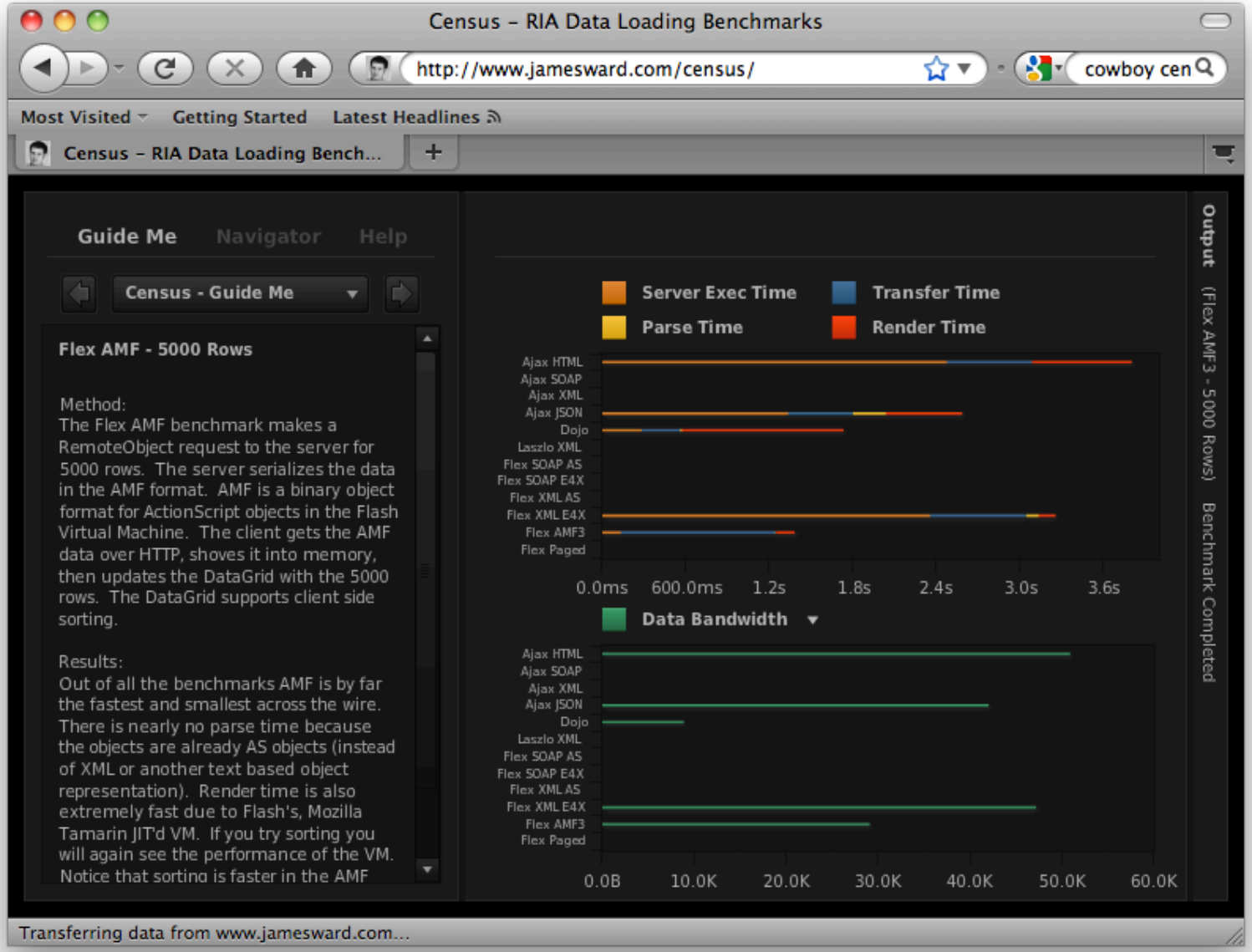
Flash Player in HTML



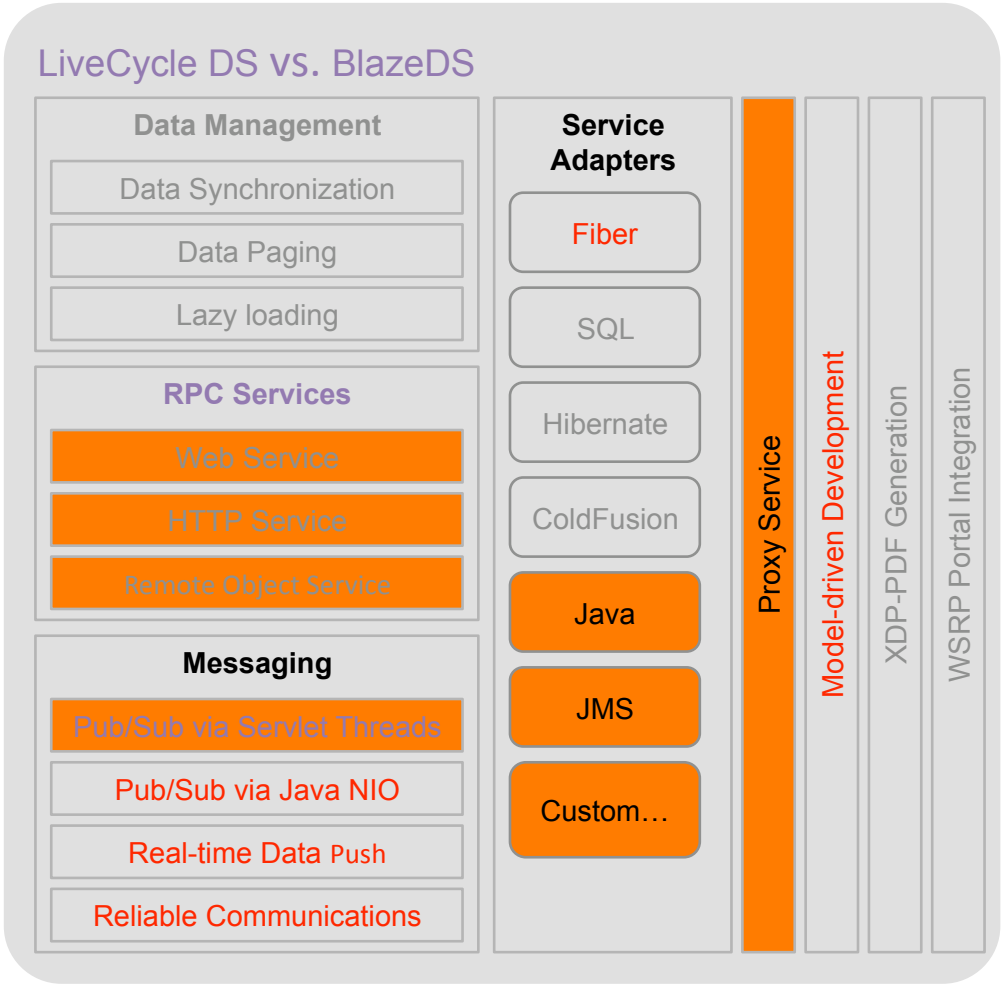
MXML

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
                applicationComplete="employees.send()">
 <mx:HTTPService id="employees"
      useProxy="false"  method="POST"
      url="http://localhost:8080/test/employees.jsp" />

 <mx:DataGrid  dataProvider="{employees.lastResult.people.person}">
    <mx:columns>
      <mx:DataGridColumn dataField="name" headerText="Name"/>
       <mx:DataGridColumn dataField="age" headerText="Age"/>
       <mx:DataGridColumn dataField="skills" headerText="Skills"/>
    </mx:columns>
  </mx:DataGrid>
</mx:Application>
```

James Ward “Census” Benchmark

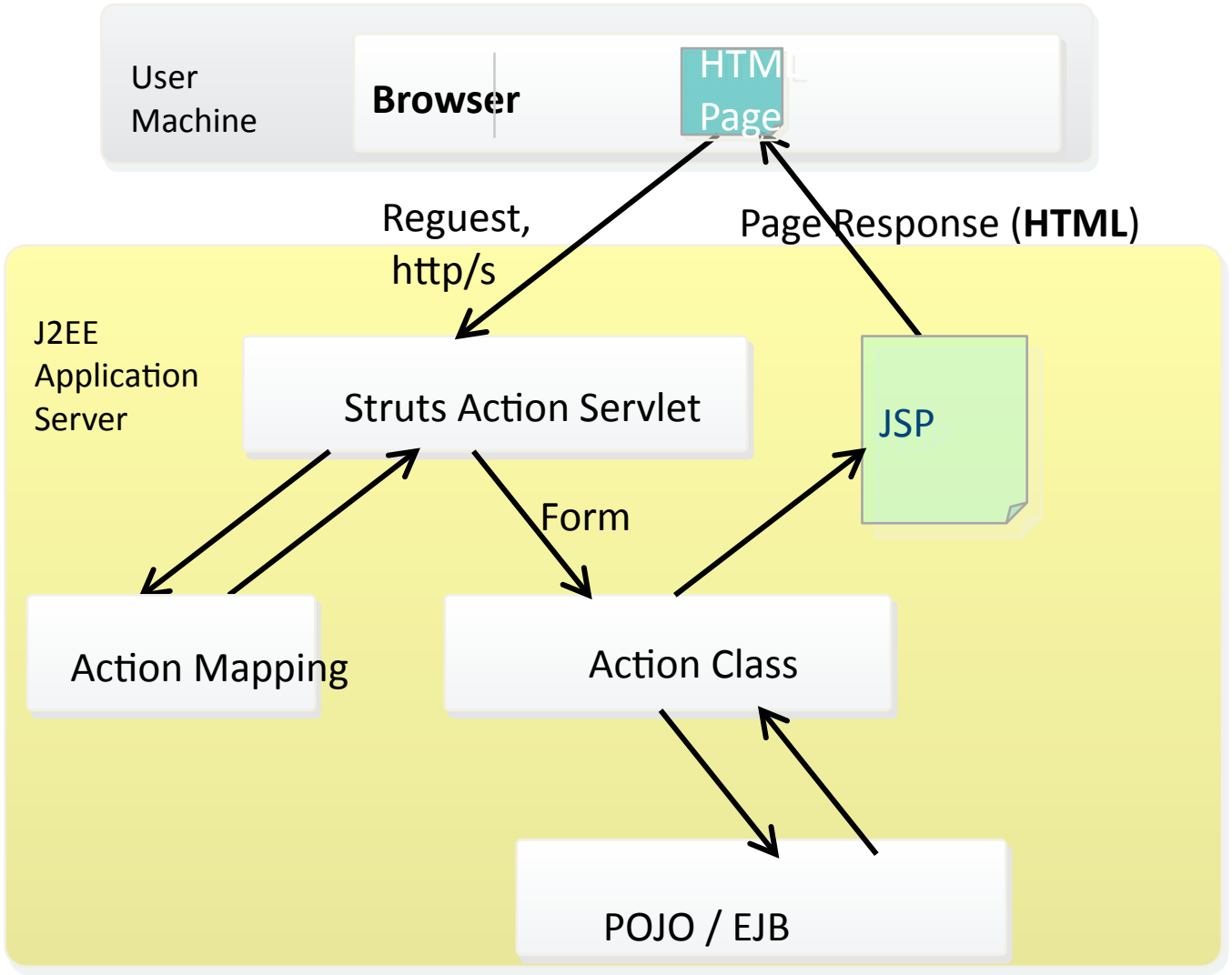


Integration With Server: LCDS vs. BlazeDS



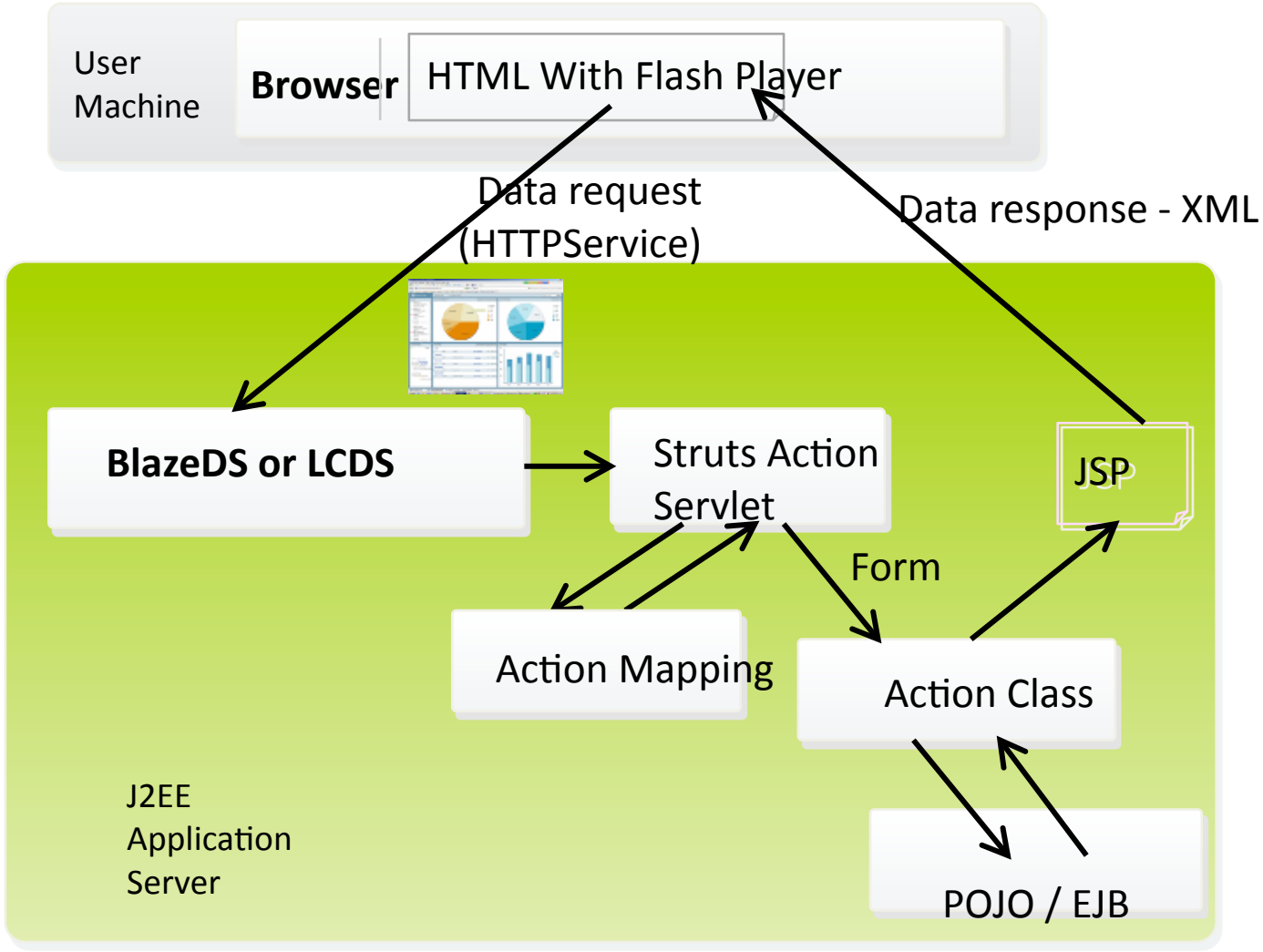
- LCDS advantages:
  - Scalability (10000 users /CPU)
  - Reliability
  - Developers' Productivity
- BlazeDS advantage:
  - Free and open source

Data Workflow with Java Struts Framework



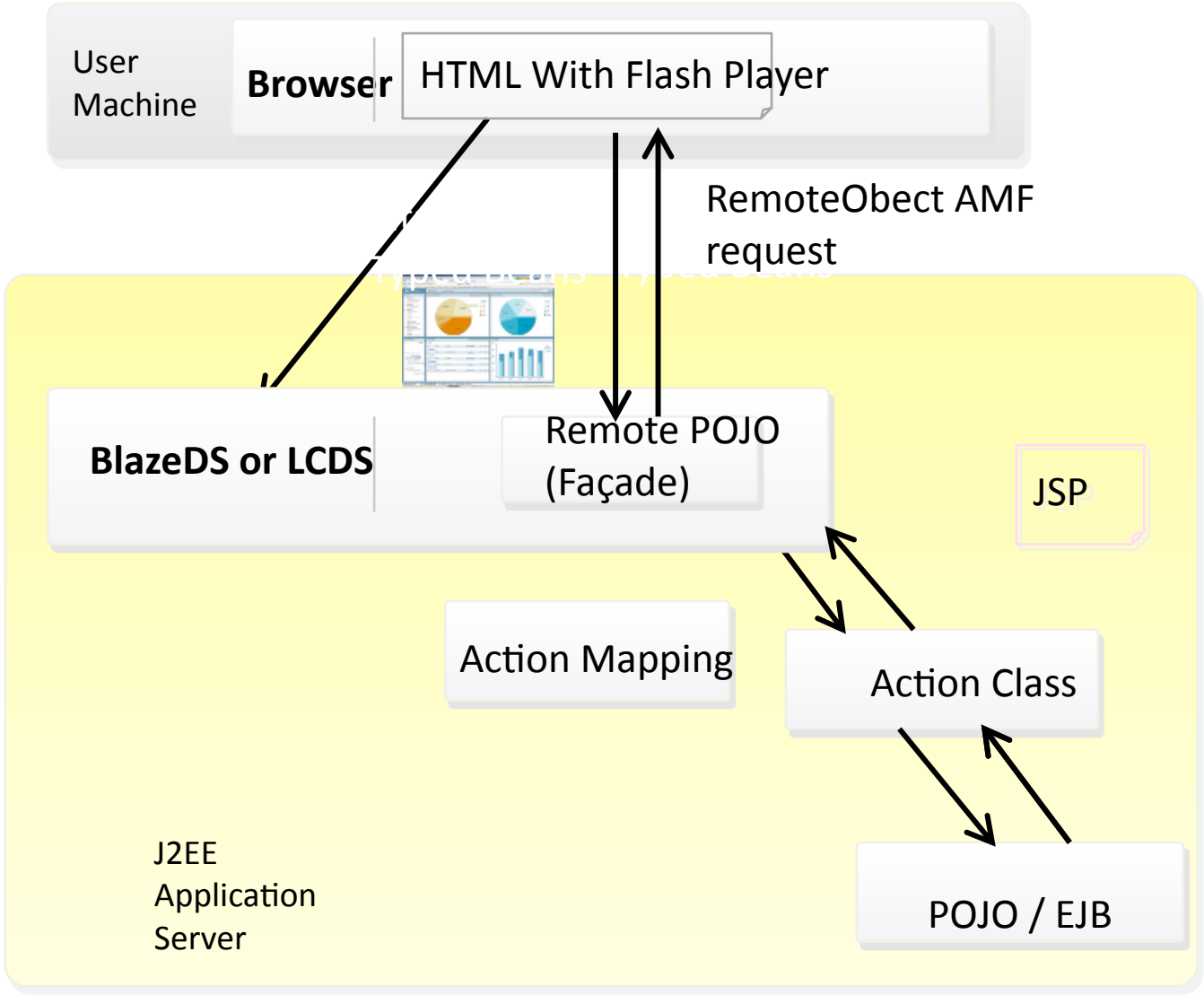
Flex-Struts Integration, Take 1

Approach 1: Add Flex on top of existing Struts app as is.



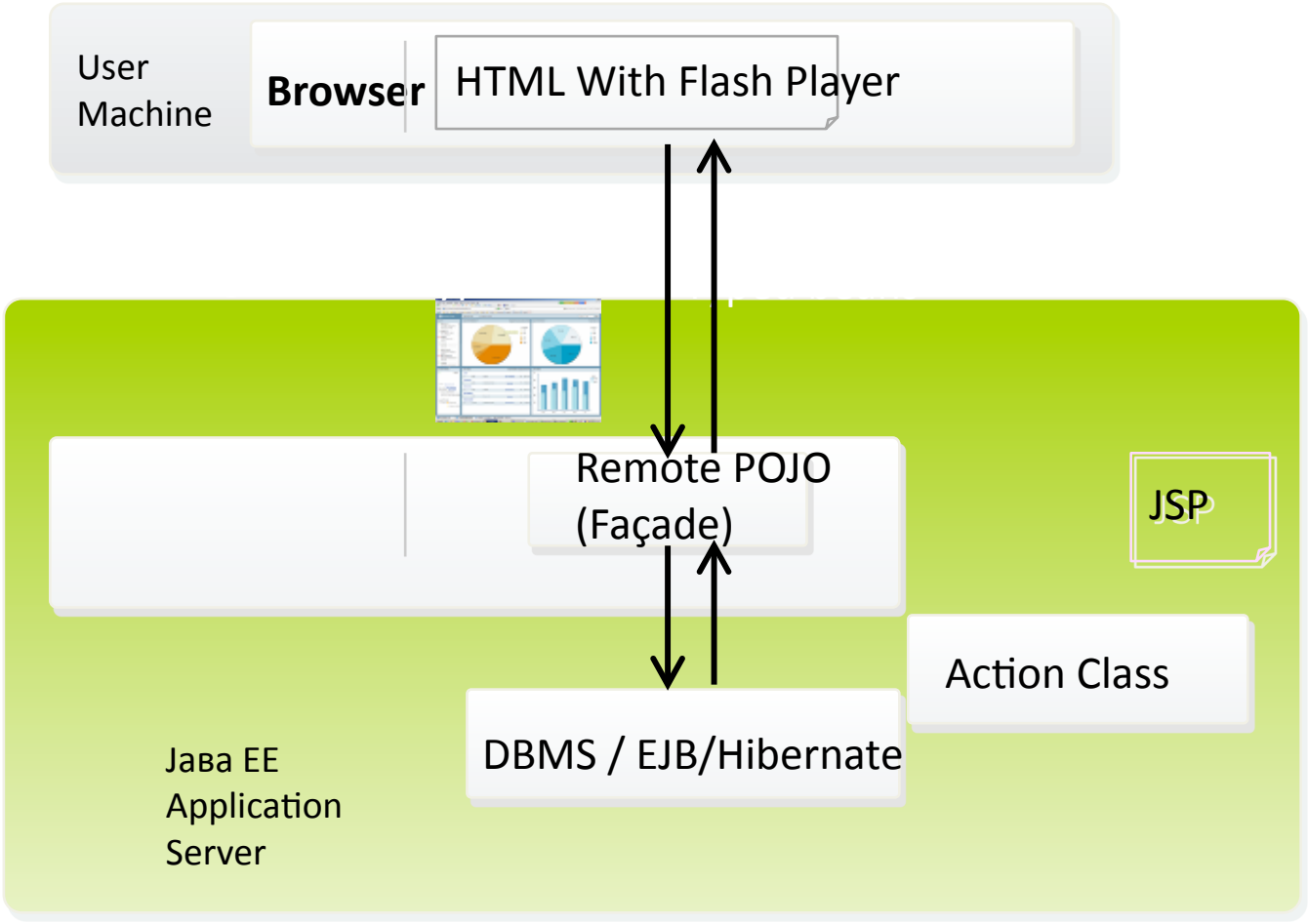
Flex-Struts Integration: Take 2

Do we really need the request-form maze of Struts?



Flex-Struts: Take 3, No Struts

As RIA handles UI navigation on the client, the entire notion of Flow-based Web frameworks becomes obsolete.



Manual coding of retrieval from Java

- Develop Java Data Transfer Object
- Develop Java Data Access Object (DAO) that knows how run SQL queries select, update, insert, delete) or call stored procedures
- Configure JDBC driver as per documentation of your Java servlet container and DBMS
- Configure a **destination** in remoting-config.xml pointing at the Java DAO.
- Make a RemoteObject call to a function using the above **destination**.
- Put the received data into an ArrayCollection

Code fragment from EmloyeeDAO.java

```
public class EmployeeDAO extends Employee implements IEmployeeDAO{

 public List <EmployeeDTO>  getEmployees(){

    String sql = "select * from employee";
    ArrayList<EmployeeDTO> list = new ArrayList<EmployeeDTO>();
    ResultSet rs = null;
    PreparedStatement stmt = null;
    Connection conn = null;
    try{
      conn = JDBCConnection.getConnection("jdbc/test");
      stmt = conn.prepareStatement(sql);
      rs = stmt.executeQuery();

       while( rs.next() ) {
        EmployeeDTO dto = new EmployeeDTO();
        dto.emp_id = rs.getLong("emp_id");
        dto.manager_id = rs.getLong("manager_id");
        dto.emp_fname = StringUtil.truncate(rs.getString("emp_fname"));
        list.add(dto);
       }
      return list;

    } catch(Throwable te) {
         te.printStackTrace();
         throw new DAOException(te);
    } finally {
      try {rs.close(); rs = null;} catch (Exception e){}
      try {stmt.close(); stmt = null;} catch (Exception e){}
      JDBCConnection.releaseConnection(conn);
    }
 }
```

Manual coding of data persistence

- Keep track of all changes of ArrayCollection objects caused by the user's action (i.e. create an array of changed objects)
- When the user hits Save, make a remote call to a Java DAO passing the changed objects to it
- In Java, loop through the collection of changed objects and and apply SQL insert, update, delete based on some flag.

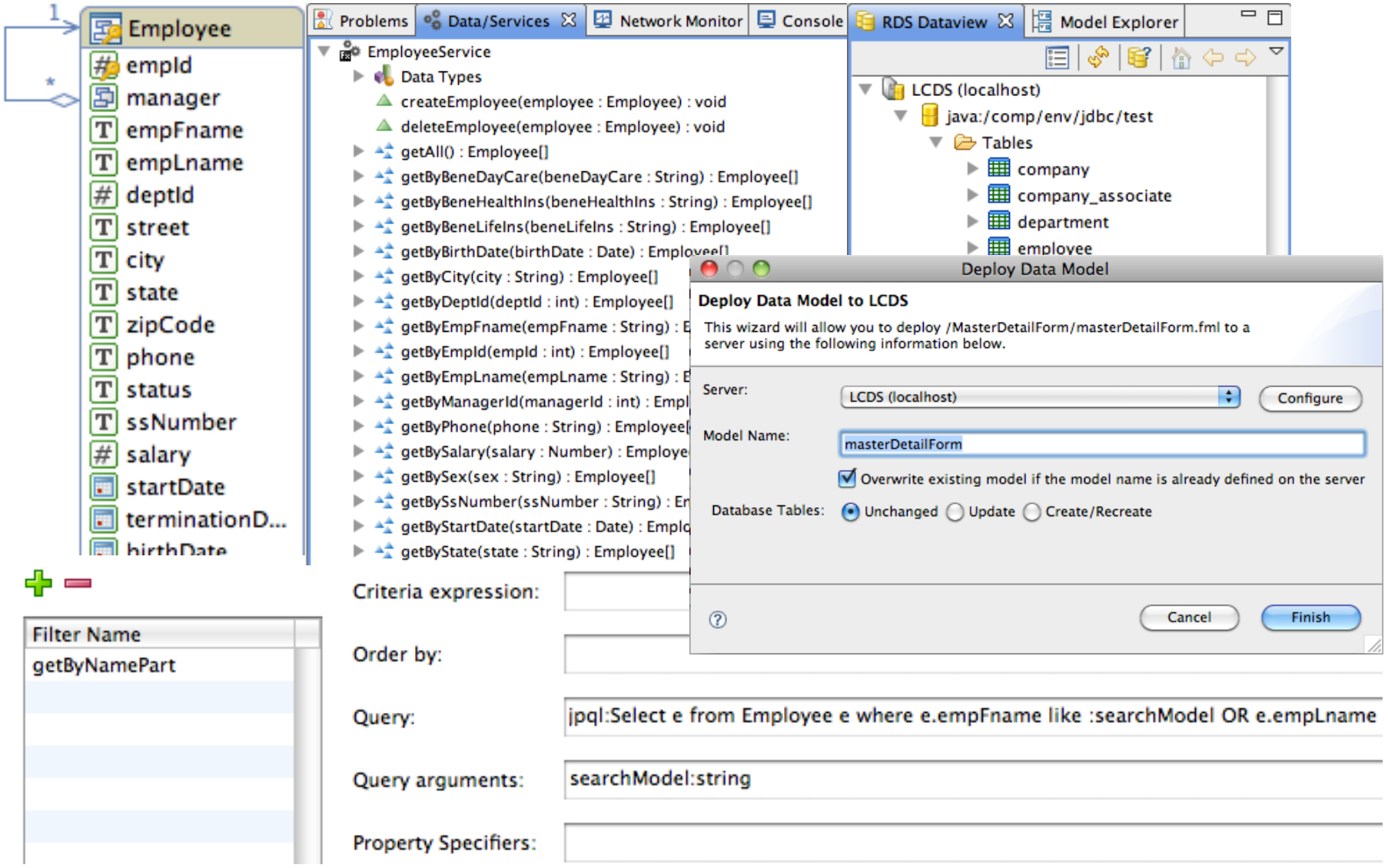
A fragment from EmployeeDAO to insert a new record

```
private ChangeObject doCreate_getEmployees(Connection conn, ChangeObject co) throws SQLException{
        PreparedStatement stmt = null;

        try {
            String sql = "INSERT INTO employee " +
            "(emp_id,manager_id,emp_fname)"+
            " values (?,?,?)";

            stmt = conn.prepareStatement(sql);

            EmployeeDTO item = (EmployeeDTO) co.getNewVersion();

            stmt.setLong(1, item.emp_id);
            stmt.setLong(2, item.manager_id);
            stmt.setString(3, item.emp_fname);

            if (stmt.executeUpdate()==0)
                throw new DAOException("Failed inserting.");
            return co;
        } finally {
            try { if( stmt!=null) stmt.close(); stmt = null;} catch (Exception e){}
        }
    }
```

Alternatives to manual coding

- Use open source dpHibernate Flex library and adapter for BlazeDS http://code.google.com/p/dphibernate/
- If you can afford, use LCDS DataServices that support automatic Flex/data synchronization (ChangeObject), but you'd need to write Java manually
- If you can afford, use LCDS ES2 Model-driven development that comes with Hibernate adapter. Code is generated in memory, and it's not Java
- Use Clear Data Builder plugin that works with BlazeDS and automatically generated AS3 and Java code for retrieval, change objects, and SQL for persistence.

Model-Driven Development with LCDS ES2



LCDS 3.0 MDD DEMO

Open Source Clear Toolkit includes CDB

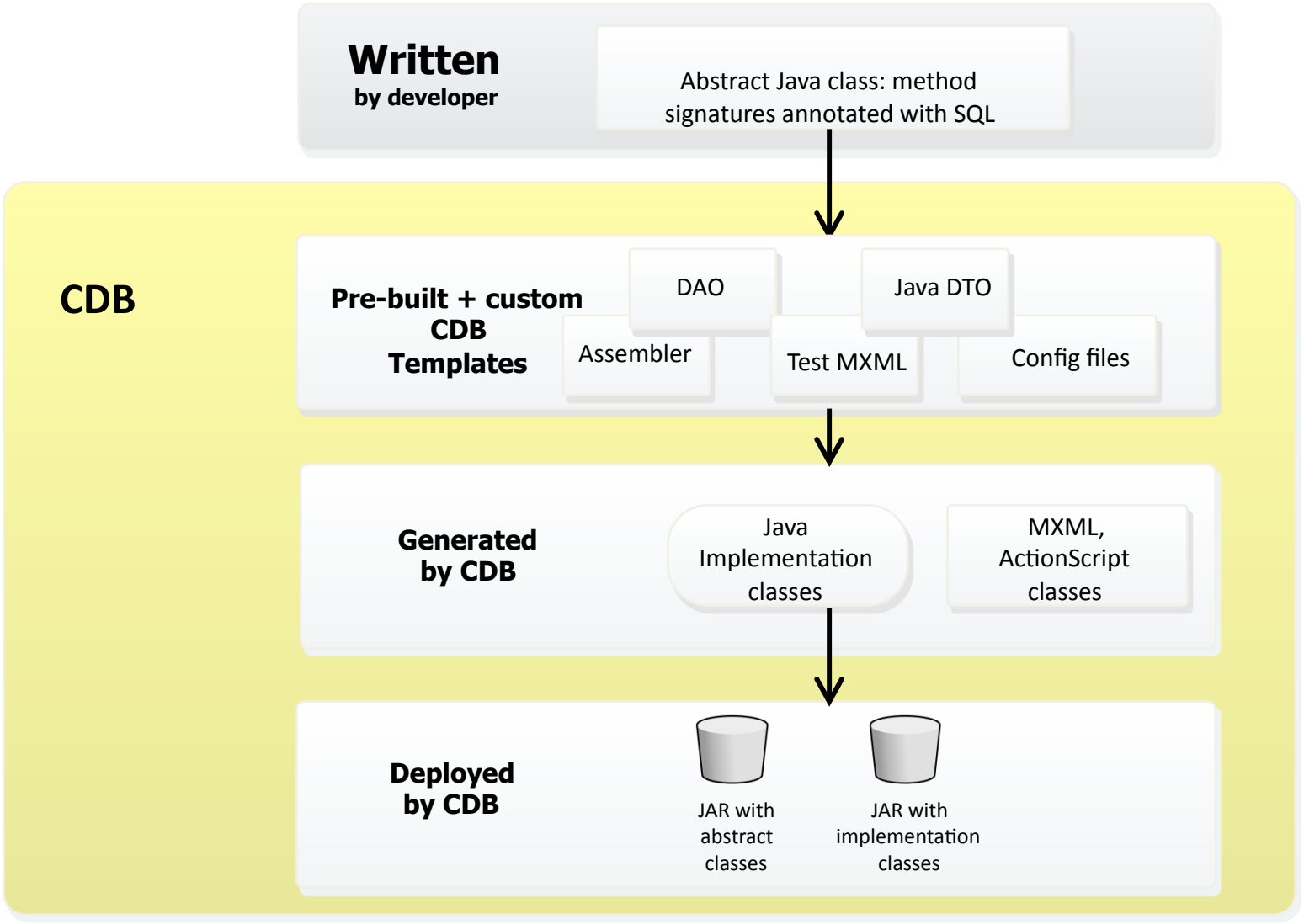
Clear Data Builder is an Eclipse plugin that generate a CRUD application based on either SQL or Java DTO:

http://sourceforge.net/projects/cleartoolkit

(Downloads, documentation, free license)

A screencast that shows the CRUD generation is here:
http://www.myflex.org/demos/CDB_blazeds_db2/CDB_blazeds_db2.html

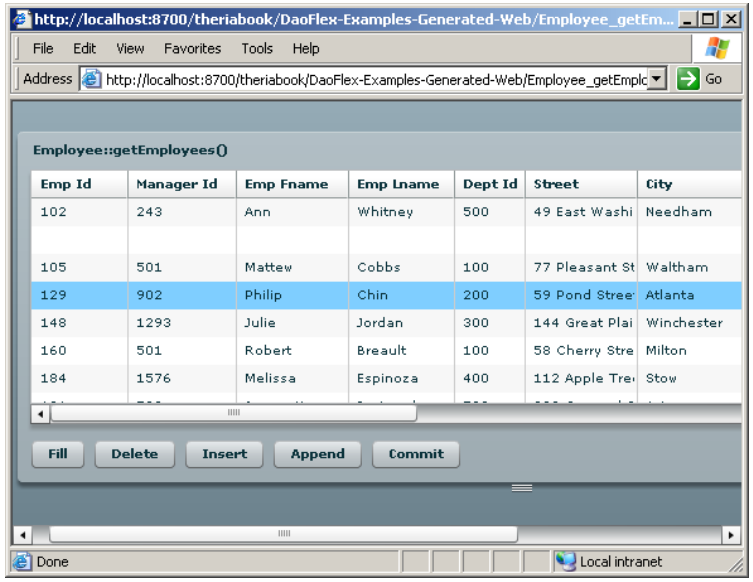
CDB SQL Mode: Complete Automation



Clear Data Builder - SQL Mode

```
package com.theriabook.datasource;
/**
 * @daoflex:webservice
 *   pool=jdbc/theriabook
 */
public abstract class Employee {
/**
* @daoflex:sql
*  sql=  select * from employee where start_date < :startDate
*  transferType=EmployeeDTO[]
*  keyColumns=emp_id
*/
public abstract List getEmployees(Date startDate);
}
```

>



- Code generation solution that completely automates manual Java coding for CRUD (create, retrieve, update, delete) tasks
- Integrates with Eclipse or works standalone as Ant task
- Goes all the way from simple annotated Java class to deployed JARs in the WEB-INF/lib
- Generates all required Java, ActionScript, MXML, XML

DTO2Fx Plugin Creates AS3 DTO's from Java

http://sourceforge.net/projects/cleartoolkit/files/ browse for DTO2Fx.pdf

```
package com.farata.test;
import com.farata.dto2fx.annotations.FXClass;
import com.farata.dto2fx.annotations.FXIgnore;

@FXClass
public class EmployeeDTO {
        public firstName;
        private String sex;
        . . .
        private String uid;

        public long getSex() {
          return sex;
        }
        public void setSex(String sex) {
          this.sex = sex;
        }

        @FXIgnore
        public String getUid() {
          return uid;
        }
        @FXIgnore
        public void setUid(String uid) {
          this.uid = uid;
        }
}
```

>

```
package com.farata.test {
public class _EmployeeDTO extends flash.events.EventDispatcher
        implements mx.core.IPropertyChangeNotifier,
        mx.core.IUID {

@FXClass
public class EmployeeDTO {
        private  var _firstName:String;
        private var _sex:String;
        . . .

        private var uid:String;
 [Bindable(event="propertyChange")]
  public function get firstName():String {
   return _firstName;
  }
  public function set firstName(value:String):void {
   const oldValue:String = this._firstName;
   if (oldValue != value) {
     this._firstName= value;
     dispatchUpdateEvent("firstName", oldValue, value);
   }
  }
 [Bindable(event="propertyChange")]
  public function get sex():String {
   return _sex;
  }
  public function set sex(value:String):void {
   const oldValue:String = this._sex;
   if (oldValue != value) {
     this._sex = value;
     dispatchUpdateEvent("sex", oldValue, value);
   }
  }
```

Data Management with CDB

- Clear Data Builder supports data synchronization over RemoteObject and Consumer (BlazeDS is ENOUGH)
- Enables client-side managed transactions:
  - transactional commit from several array collections
  - transactional batching of any number of remoting operations
  - supports hierarchical nesting of collections
- Major objects:
  - DataCollection
  - BatchService

What's DataCollection?

- *com.farata.collections.DataCollection* is included in Clear component library (clear.swc).
- It's a subclass of Flex ArrayCollection
- DataCollection features:
  - destination aware, knows which destination (Java object) to use to *fill* itself with data
  - change tracking, keeps track of all interactive and programmatic changes (automatically creates ChangeObject's)
  - *sync*-able – knows the Java class method to use for submitting all accumulated changes to the server
  - supports server data push – it has an internal Consumer object

Demo of the CRUD generation with Clear Data Builder

Further reading on using Spring and Hibernate with Flex

- Spring and Hibernate with BlazeDS: http://bit.ly/d30Q0C
- Basics of Flex/Hibernate integration by Shashank Tiwari: http://bit.ly/aVi16D
- Flex 4/Spring/BlazeDS integration by Christophe Coenraets: http://bit.ly/945PqJ
- LCDS Hibernate assemblers: http://bit.ly/ba7d2x

Contact info and useful links

Email: info@faratasystems.com

Web site: http://www.faratasystems.com

Flex Blog: http://flexblog.faratasystems.com

Clear Toolkit Framework: http://sourceforge.net/projects/cleartoolkit/

O'Reilly Book "Enterprise Development with Flex":
http://bit.ly/bDg7IR