

Enterprise Application Integration

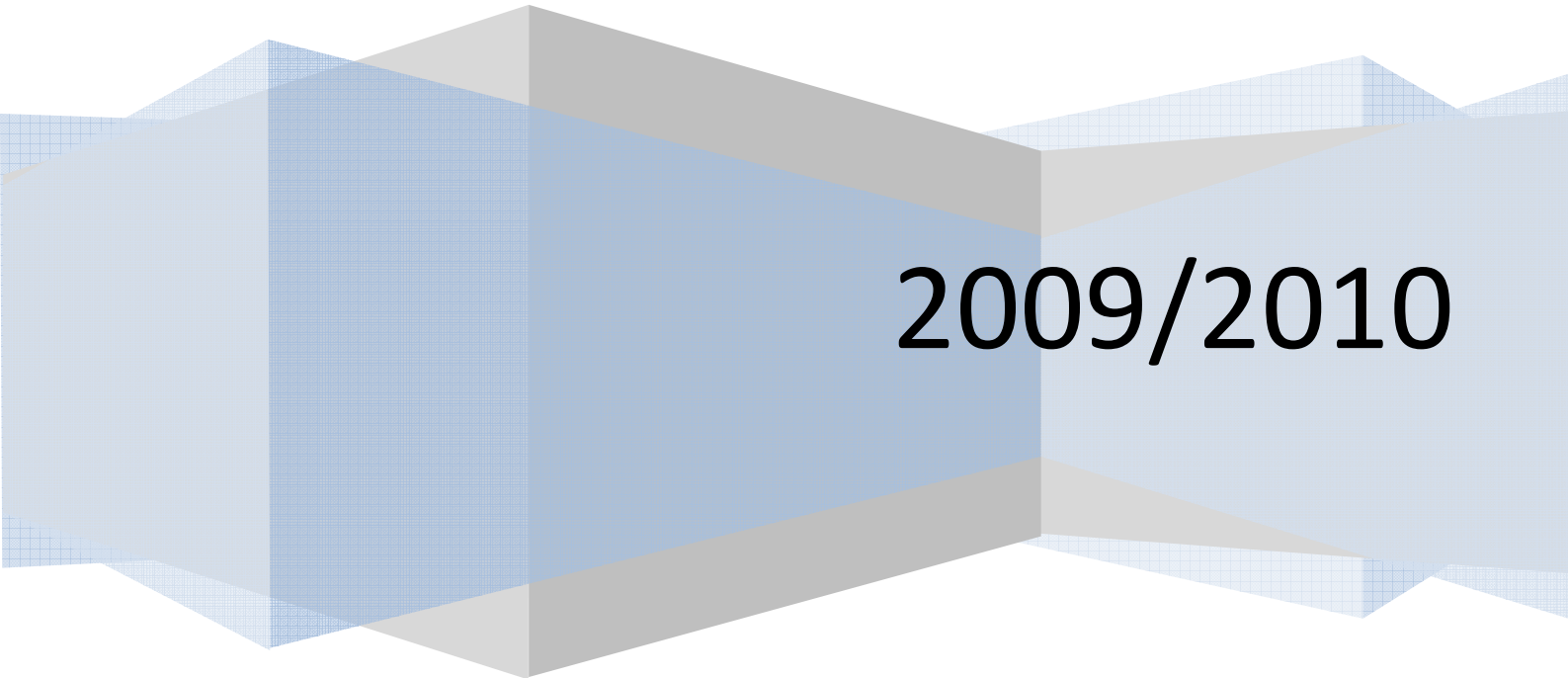
# Low-Price Cameras Online

## Project #2 Report

Carlos Simões

Miguel Oliveira

Pedro Saraiva



2009/2010

## Introduction

During the second project of the course Enterprise Application Integration, named “Low-Price Cameras Online”, decisions, from the point of view of design and implementation, had to be made. The goal of this report is to describe these decisions and the reasons why they were made.

In addition, the report presents the instructions for installing and executing the application and all the web services for the Camera Supplier and Shipping Department

## Time Spent

Student	Mostly involved in	Time spent
Carlos Simões	<ul style="list-style-type: none"><li>• Web Tier</li></ul>	24:50
Miguel Graça Oliveira	<ul style="list-style-type: none"><li>• Business Tier</li><li>• Database Tier</li></ul>	28:41
Pedro Saraiva	<ul style="list-style-type: none"><li>• Web Services</li><li>• Report</li></ul>	35:17

## Installation Instructions

### LPCO

Prerequisites:

- Ant should be installed in the system.
- Java should be installed in the system.
- JBOSS should be installed in the system.

Instructions:

1. Extract the EAIDeploy.zip to the deploy directory of the JBOSS default directory.
2. Run the SQL script in the Hypersonic JBOSS database.

## Implementation and Design decisions

The architecture components that were developed were:

- Client (**CLI**);
- Low-Price Cameras Online (**LPCO**);
- Camera Supplier;
- Shipping Department.

All these components can exist geographical disperse from each other.

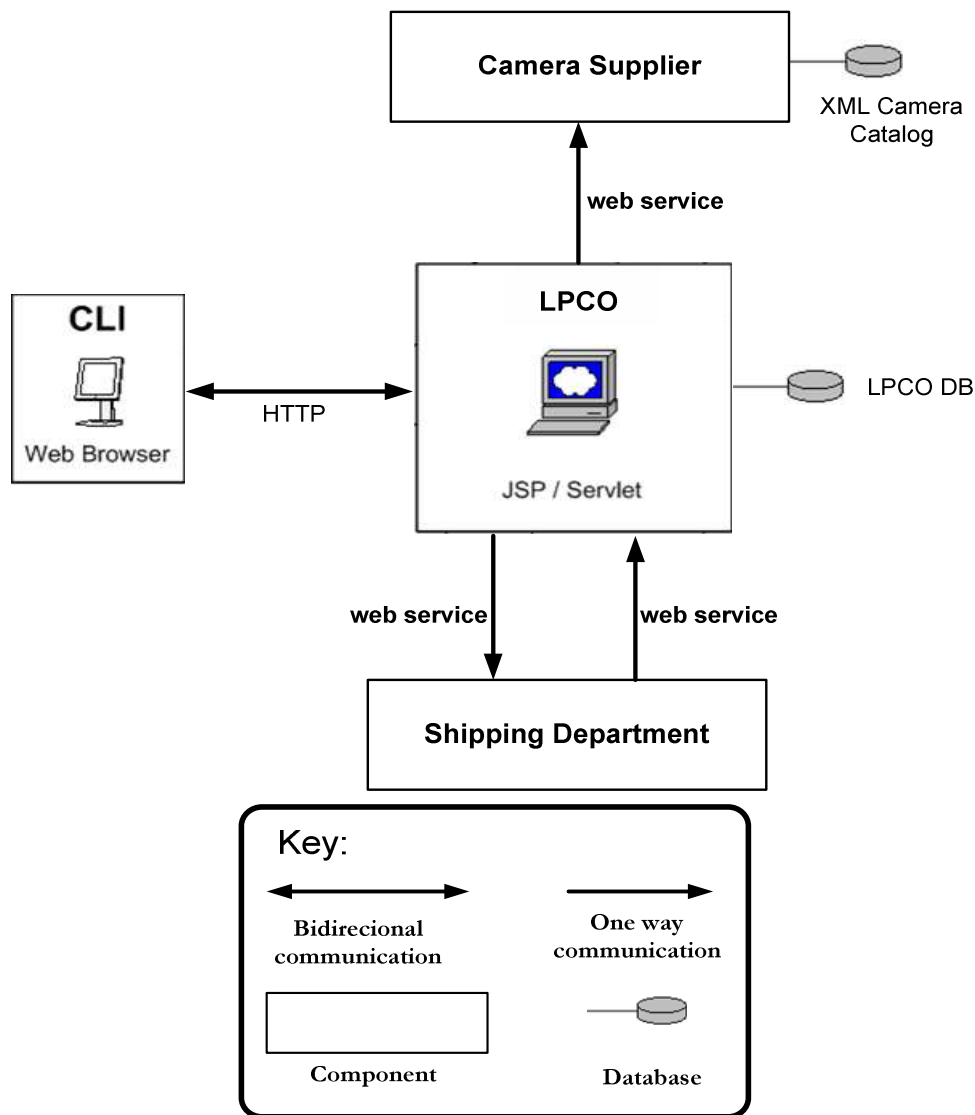


Figure 1 - Dynamic View

The **Client** is a unique application that interacts directly with the user, accepting its commands, and sending them to the server. From the server he receives and presents the results. The client does not save any persistent information. This is, the client does not save any information about the shopping cart. The shopping cart information is saved in the server.

The **LPCO** is responsible for receiving and dealing with user commands, and manage the existing sessions. Besides the user interaction, the **LPCO** is responsible to validate the user's identity and execute the user requests, either obtaining new cameras information from the **Camera Supplier** or by shipping the user products through the **Shipping Department**.

### Main Quality Attributes

3 **Availability** is to be important since we are dealing with a camera purchase online system. In a purchase system, like Amazon, it is important to have the system down the minimum time possible. Security is referred to have into consideration, but not the most important one.

**Portability** is also important, since we assumed that the online system should work in the most wide possible range of web browsers. For this reason and based on our initial experiments we decided to keep the user interface simple.

### **Strategy**

Our first strategy consisted in the centralization of all the business logic of the system, and the decoupling of client and access to data repositories. We adopted a three-tier system, where we have a middle tier (business logic) between the web-tier and data accesses.

This promotes modifiability, since all business rules are centralized, and where all changes occurred in the business logic are completely transparent to the client. In this case, no modification is required for the client. It also helps to provide a centralized point of access to the clients and with future existing services.

Using the tiered architectural pattern has big advantages, since it promotes modifiability and scalability. The modifiability is promoted in this pattern because the system is prepared for coping with some changes. In addition, in terms of scalability, the system will be able to grow on the number of clients, because the components of the system have well defined interfaces, and are independent of each other. The client only needs to be aware of the interface and use it to communicate with the business logic component (logic tier).

### **Database**

In this project, we decided to use Hibernate and HSQLDB.

Hibernate is a powerful, high performance object/relational persistence and query service. Hibernate lets you develop persistent classes following object-oriented idiom - including association, inheritance, polymorphism, composition, and collections. Hibernate allows you to express queries in its own portable SQL extension (HQL), as well as in native SQL, or with an object-oriented Criteria and Example API.

HSQLDB is a relational database management system written in Java.

### **Data Model**

We decided to have three classes to map all the low-price camera online functionalities. We decided to have a user, order and camera classes that correspond to tables in the database. Regarding an order that can include more than one camera, we can have three possible states:

- Waiting for shipping – the cameras have been ordered and paid, but have not yet shipped;
- Shipped – the cameras have been ordered, paid and shipped;
- Not paid – the cameras have been ordered, but the payment could not be executed.

## Web Services

**Camera Supplier** – This web service is responsible to obtain all the cameras where the model name matches with the keyword. The web service is based on the use of a camera catalog xml file. To discover which cameras corresponds to the search criteria we decided to use XPATH. The origin of this web service call is from the LPCO that when does not have cameras that matches the search criteria, tries to obtain information from the Camera Supplier.

The cameras database xml file should have the name “camera\_catalog.xml” and exist in the JBOSS server data folder.

**Shipping Department** – This web service is invoked when the user proceeds to the shopping cart checkout and the payment was executed with success. When this web service is invoked a new thread is created to answer the request. This web service is responsible to call back to the LPCO and with the order id informs that the order has been shipped.

**LPCO** – This web service is responsible to acknowledge that an order has been shipped and inform the user through his existing e-mail.