Micro Project

Sophie Chabridon

Revision : 545

MDS

Télécom SudParis—INF septembre 2018

1 Introduction

The aim of the project is to design and implement an e-shopping system. The system allows users to connect, browse the catalog of articles and order articles. When articles are not immediately available, users can register in order to be informed of the availability of the articles later on.

A user is characterized by a profile with a unique pseudonym, first name and last name, postal address, and email address.

An article is characterized by a unique identifier, a short description, and a category.

Architecture The general Architecture of the system is presented in Figure 1. It follows a client/server architecture. Clients and servers are distributed over the network.

- A *Directory Manager* manages the users and the articles and can create/update/browse and delete them.
- An *Administration Client* accesses the *Directory Manager* services to add, update, browse or remove users. The *Administration Client* also accesses the *Directory Manager* services to add, update, browse or remove articles. Some specific methods are proposed to manage the availability the number of articles. When an article is not currently available, use some random processing time to simulate the time necessary to get new items of this article.
- An *Order Manager* manages the orders made by clients.
- An *Order Client* accesses the *Order Manager* services. This client represents the actions made by users who want to order articles, get information on articles and be informed of the availability.

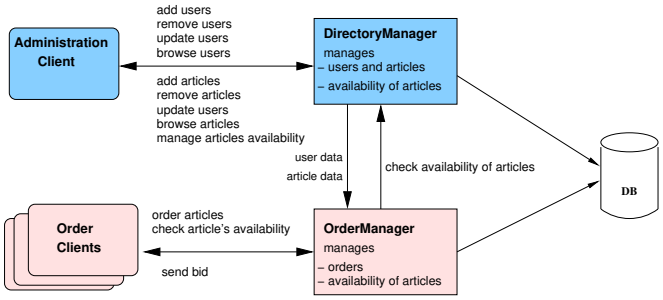

FIGURE 1 – General architecture of the system

2 Middleware technologies involved in the project

- The *Directory Manager* and the *Order Manager* are developed using the *JavaEE* technology. Information on users and articles are stored in a relational database.
- You must provide security mechanisms to ensure that your system is not vulnerable to SQL injection and provide user authentication, control of user rights.
- In a first step, the communication between the clients and the Managers will be implemented using Java RMI for synchronous communication.
- Add asynchronous communication using the *Future* mechanism to deal with articles not currently available and to inform interested users when they become available again.

3 Micro Project steps

The project will be realized in several steps.

1. Architectural choices :
   - With the material in the course "Introduction to middleware through design patterns", identify the design patterns involved in the application. For each design pattern, draw a picture presenting it and give adequate explanations. These pictures have to be included in the final report.
   - Design the persistent data structures : Do not use SQL keywords in the description of these data structures. For instance, *user* and *right* are reserved SQL keywords and should not be used in your own data structures.

   - Describe the facade of the services. Implement first synchronous communication and then add asynchronous communication.
2. Design and implement the *Directory Manager* and the *Administration Client*.
3. Design and implement the *Order Manager* and the *Order Client*.
4. Clients should use a simple textual interface. Clients with a predefined scenario should be provided. Additionally, interactive clients can be developed.
5. Add an asynchronous interface to the *Order Manager* to deal with not currently available articles.
6. Write a script to automate the demonstration.

Database You are advised to use the *Derby* database as during the labs and administrate it using the *ij* tool.

Clients and user interfaces This micro-project is about middleware for distribution not about graphical user interfaces ! We ask for simple clients, *i.e.,* test clients with no interactions and with predefined users and messages are absolutely sufficient. You are asked to write either test clients with predefined messages or command line clients (to specify all the parameters on the command line).

4 What is expected at the end of the project

The results of the micro project will be :

Report A small report (max. 10 pages) describing your solution and the associated design patterns, the architecture and technology choices, the persistence solution, a short user manual so that we can test your project, the encountered difficulties and all information you will judge necessary.

Sources All the material (report, slides of the oral presentation, program SOURCE files) has to be returned by email before the deadline and in one archive. The archive will contain a root directory with the name of the students who worked on the project (*e.g., nameA-nameB.tgz* archive contains *nameA-nameB/* directory). Be careful :

- Do not include files that can be generated (e.g. .class) ;
- Use ascii7 characters only in the name of files (no accent, no white etc.) ;
- Include the *pom.xml* (for maven) used to automate the generation of classes and also to launch the demonstration ;
- If you develop your application with the Eclipse Integrated Development Environment (IDE), make sure that it can be run as a standalone application ;
- The code of a demonstration with a script showing your solution in action.

Plagiarism Plagiarism is forbidden. An anti-plagiarism tool will be used to detect similarities in the delivered code. If too many similarities are found, the grade will be 0.

Grading Report : 7 points, Project implementation : 7 points (including quality of source code), Code of an operational demonstration : 6 points

Good work !