CS 5204

Term Project Report

Design and Implementation of an
Electronic Commerce System using Java RMI

Yuxin Chen

Rao Shen
Design and Implementation of an
Electronic Commerce System using Java RMI

By

Yuxin Chen, Rao Shen

ABSTRACT

We have designed and implemented a simple electronic commerce system as a term project of CS5204. The system allows the client to view and purchase items from either store of the two stores and provides asynchronous interaction between the client and the store. User-friendly Graphic User Interfaces (GUIs) are provided to facilitate the usage of the system.

To implement this distributed simple commerce system, Java Remote Method Invocation (RMI) mechanism was employed. We use Monitor to synchronize the access to the database in store sides and Java multi-thread technology to process concurrent orders at store side.

The primary objectives of this project are to master the concepts of the distributed system (especially object model) and synchronization, to gain some practical experiences of using Java’s RMI mechanism, and to explore Java’s GUI programming techniques.

SYSTEM DESCRIPTION

Our system is a simple electronic commerce system designed for solving a specified distributed commerce problem. The system consists of clients, two "stores", and one "bank". The system allows the client to view and purchase items from either store. The client provides an "account number" which the store will verify by contacting the "bank". Before authorizing the expenditure, the bank will seek confirmation from the
client that the amount and the store are acceptable. The client does not block waiting for
the store to reply to its requests to view or purchase items. This asynchronous feature
allows the client to conduct business with several stores simultaneously.

**DESIGN OF CLIENT**

A graphic user interfaces (GUI) using *Java Swing* has been developed to facilitate the interaction between clients and stores.

Revoking the remote method at the store server after the query is submitted will return query results. Then custom can select the stuff to order, which will revoke the confirmation methods on the remote site.

There are two main windows in the client site. The first one is where the user can select a store to visit, input his query, and then submit the query to the store site. After submitting a query, a new thread is forked to looks up the store site and invokes remote method to process the query and this window is ready to take in selections form the user again. The other main window then will display the query result. The user can concurrently go to visit a different store and conduct other queries and orders.

Before authorizing the expenditure, the bank needs to communicate with the client site to get the user’s confirmation about the transaction. This is implemented by providing a remote method at the client side. On the client side locates a remote object implementing two remote methods.

**DESIGN OF STORE SERVER**

On the store side locates a remote object implementing the remote interface. This remote object can be invoked to perform queries against the database on the store server and to interact with the bank server.
Access to the database on the store server is synchronized to ensure the consistency of the database by using the Monitor.

When the order button is clicked, one of the remote methods on the store server will be revoked to check whether the selected items are in stock. If at least one of the items is not available, the order is aborted and the client is informed. If all items are available, the store server deducts the number of each item from the database on the store server and contacts the bank to verify the client’s account. Then the remote method on the bank server will be revoked.

The bank server checks to see if the account is valid. If the account is invalid, the bank informs the store to abort the order. Otherwise, the bank checks to see if there is enough money in the account for the order. If money is not enough, the bank informs the store site to abort the order. The bank site seeks confirmation from the client before authorizing the expenditure. If the client confirms the order, the bank will deduct the specific amount of money form the account and informs the store site to commit the order, otherwise asks the store site to abort the order.

If the store receive abort message, the databases on the store server and the bank server will not be modified. Otherwise, the client will be informed that the selected stuffs have been ordered and the amount of money has been spent.

**DESIGN OF BANK SERVER**

On the bank side locates a remote object implementing the remote interface. This remote object can be revoked to perform authentication of the customer and to interact with the client.
Synchronized methods are used to implement the lock mechanism to ensure consistency of database on the bank server. To access an account, a lock needs to be acquired first. Once an account is locked, any subsequent requesting processes will be blocked to wait for the lock to be released.

The bank site uses the stored address to contact the client to seek authentication and authorization from a client. If a fake account number is encountered, the bank will fail to get contact with the true account owner and an exception is raised, resulting in the order to be aborted.

**SUMMARY**

Through the design and implementation of this simple electronic commerce system, we have reached our pre-set objectives. We thoroughly explored the procedures of building a distributed system using Java’s RMI mechanism. Also, we obtained significant insight of Java’s Swing.