

Enterprise Computing (WS 2014)

Exercise 1 (3 Portfoliopunkte)

Info:

- The solution to this exercise must be handed in by Tuesday, Nov 4th 2014, 2PM to Markus Klems.
- The solution must be printed out. Please write your name on the solution sheet.

Task 1 – Layers & Tiers (20%)

What do you think of this statement: „There is no problem in system design which cannot be solved by adding a level of indirection”? Write a short explanation of the pros and cons that are the consequence of adding indirection into a system.

Solution:

The statement above is indeed true in system architecture. By adding more level of indirection, the system will become more modular which means more opportunities for distribution and parallelism. It also allows encapsulation, component based design and reuse. But it also has disadvantages, such as: the system becomes more complex to monitor and manage. Furthermore, it can also make the system performance to suffer considerably. So, a system designer has to try to balance the flexibility of modular design with the performance demands of applications.

Task 2 – Middleware (20%)

Shortly describe the main differences between synchronous and asynchronous communication.

Solution:

In synchronous communication, the caller sends a request to a receiver and waits for a response of the receiver to come back before continue doing its work. The interaction requires both the caller and the receiver to be alive at the same time. On the other hand, in asynchronous communication, the caller sends a message that gets stored somewhere and does not wait for a response of the receiver. After having sent the message, it returns immediately without waiting for a response and continue doing its work.

Task 3 – JMS (60%)

Budi Yanto (308819)

Set up the Eclipse project with JMS sample source code like this:

- Clone the project from <https://gitlab.tubit.tu-berlin.de/klems/messaging>
- Open the project in Eclipse.
- Build the project with Maven.
- Change the `java.naming.security.credentials` in `jndi.properties` from `secret` to `AKZZ91s72TRa`

a) PubSub Chat Example (20%)

In `Chat.java`, change the following line

```
TopicSubscriber subscriber =  
    subSession.createSubscriber(chatTopic, null, true);
```

to

```
TopicSubscriber subscriber =  
    subSession.createSubscriber(chatTopic, null, false);
```

Now open two Chat windows with users “Angie” and “Edward” (start the Java Chat application twice) and type:

Angie: How is the weather in Moscow?

Edward: Pretty cold.

Angie: Bye

What is the actual output that you see on the console after having made the slight code change above? Write the output of Angie’s Chat window here:

Solution:

After having made the slight code change above, Angie and Edward can see their own message on their chat windows, which otherwise will be inhibited if the `noLocal` attribute is set to `true`.

Output of Angie’s chat windows:

How is the weather in Moscow?

Angie: How is the weather in Moscow?

Edward: Pretty cold

Bye

Angie: Bye

b) Point-to-Point QBorrower and QLender Example (40%)

The example code is in the project package *com.jms.p2p*. First, you must set the *jndi.properties* settings as follows because we are in a P2P messaging scenario here:

```
connectionFactoryNames = QueueCF
queue.LoanRequestQ = jms.LoanRequestQ
queue.LoanResponseQ = jms.LoanResponseQ
```

Modify the example as follows and add your code snippets in the blanks below:

- Add an additional String value “Name” to the MapMessage
- The borrower must enter her/his name in addition to salary and requested amount of money
- The lender declines the request, no matter which salary or amount, if the borrower’s name equals to “Donald Duck”

Solution:

```
// QBorrower.java
```

```
private void sendLoanRequest(double salary, double loanAmt, String name) {
```

```
...
```

```
    msg.setDouble("LoanAmount", loanAmt);
```

```
    msg.setString("Name", name);
```

```
    msg.setJMSReplyTo(responseQ);
```

```
...
```

```
}
```

```
public static void main(String argv[]) {
```

```
...
```

```
    // Parse the deal description
```

```
    StringTokenizer st = new StringTokenizer(loanRequest, ",") ;
```

```
    double salary = Double.valueOf(st.nextToken().trim()).doubleValue( );
```

```
    double loanAmt = Double.valueOf(st.nextToken().trim()).doubleValue( );
```

```
    String name = st.nextToken().trim();
```

```
    borrower.sendLoanRequest(salary, loanAmt, name);
```



```
...
```

```
}
```

```
// QLender.java
public void onMessage(Message message) {
    ...
    double loanAmt = msg.getDouble("LoanAmount");
    String name = msg.getString("Name");

    ...
    if (      name.equals("Donald Duck")      ) {

        accepted = false;

    } else {
        if (loanAmt < 200000) {
            accepted = (salary / loanAmt) > .25;
        } else {
            accepted = (salary / loanAmt) > .33;
        }
        
    }
    
    System.out.println("" +
        "Percent = " + (salary / loanAmt) + ", loan is "
        + (accepted ? "Accepted!" : "Declined"));

    ...
}
```

—