

Excercise Sheet 3

Solution

Lecture Distributed Systems

Winter Term 2024/25

Exercise 1: RPC and unions

If the runtime system is unable to determine which value type is in the field, it cannot pack it correctly. This means that unions cannot be permitted in an RPC system, unless there is an identifying field that uniquely describes what is contained in the variant field. The identifying field must not be under the control of the user.

Exercise 2: Remote object references

Using Java we can express such an implementation e.g. by the following class:

```
public class Object_reference {
    InetAddress server_address;    // Network address of the object server
    int server_port;              // Port that identifies the server
    int object_id;                // Identifier for this object
    URL client_code;              // (Remote) file containing the code of the
                                // client stub
    byte[] init_data;             // Possible additional initialization
                                // data for the stub
}
```

The object reference should contain at least the transport layer address of the server on which the object is located, including the port. We also need an object ID, because the server can contain several objects. In our implementation, we use a URL to point to a (remote) file that contains all the required client-side code. An alternative implementation would be to directly use a serialized client stub as an object reference. For example, this approach is followed in Java RMI, where proxies are passed as a reference.

Exercise 3: Exceptions

Because the exceptions are originally thrown on the server side, the server stub can only catch the exception, package it as a special error response, and return it to the client. The client stub, on the other hand, must unpack the message and throw the same exception to keep the access to the server transparent. Consequently, the exceptions must also be described in an interface definition language.

Exercise 4: Threads (**Mandatory exercise for 6 CP, submit via moodle!**)

Exercise 5: RMI basics and architecture (**Mandatory exercise for 6 CP, submit via moodle!**)

See chapter 2 and 3 of the lecture.

Exercise 6: Programming: Java RMI - Hello-World!

You will find the solution to this problem in the archive [103eFiles.zip](http://www.bs.informatik.uni-siegen.de/web/wismueller/v1/vs/103eFiles.zip)¹ on the lecture's web page.

¹<http://www.bs.informatik.uni-siegen.de/web/wismueller/v1/vs/103eFiles.zip>