Prof. Dr. A. Poetzsch-Heffter
Dipl.-Inform. Kathrin Geilmann

University of Kaiserslautern
Department of Computer Science
Software Technology Group

Advanced Aspects of Object-Oriented Programming (SS 2010)

Practice Sheet 11 (Hints and Comments)

Exercise 1 Asynchronous vs synchronous Calls

a) Synchronous calls: all methods are executed in the order they appear in the snippet. Output:

```
i=5
i=9
```

Asynchronous calls: it is unknown, when the methods are executed:

```
i=5
i=9
```

or, if C handles the message send by B before the one of A

```
i=9
i=5
```

b) With synchronous communication methods are executed completely before the control returns to the caller. That means that the order of execution of methods corresponds to the order of the send messages. In asynchronous contexts, the order of messages may change. Depending on the language, certain guarantees about the message order can be given, but in most cases the order will not be total but some partial order. E.g. in JCoBox it is guaranteed, that if an object a of CoBox a sends two messages to objects in another CoBox, these messages will be handled in the same order as they have been send, but the system makes no guarantees about messages send to or by other CoBoxes.

In multi-threaded contexts the usage of synchronous communication is easier because the behaviour is "easier" to predict. But synchronous communication is a source for deadlocks, which are less likely in asynchronous settings, but for the price of even less predictable behaviour.

Exercise 2 RoboWorld

a) -

b) The robot runs concurrently to the controller, i.e. the robot continuous to move while the controller handles the message. If by chance, the time between the robot sending the sensed-message and receiving the turn-message is too long, the robot already crashed into the wall. This is not a special problem of asynchronous communication but this way to communicate makes it even more difficult to handle this situation.

c) -

d) Idea: use an additional CoBox to handle a datastructure that receives the gold events and distributes them to the robots.

Exercise 3 Distributed Programming with RMI

a) -

b) In general callbacks are supported, but as RMI copies non-remote objects to the remote side, it is difficult to see wether a callback will occur or not. Direct callbacks can only occur if the first object is also a remote object, but indirect callbacks can occur if the controlflow passes via another remoteobject back to the first object

According to the W3C: *A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.*

RMI is directly supported by the language and uses real java-objects as parameters and message receivers. With some exceptions they behave like standard objects, whereas a webservice uses its own message format and it transfers messages not objects. Webservices cannot provide callbacks, because they don't have access to the object system.

c) As remote objects are not copied during transfer, the usage of a remote object as parameter or return value exposes the remote object, even if it has not been registered before. The name service only serves as a central repository to find objects by their name, it is has to be used to establish the first connection between two machines. After that, references to remote objects may be passed freely around.