

Distributed Systems

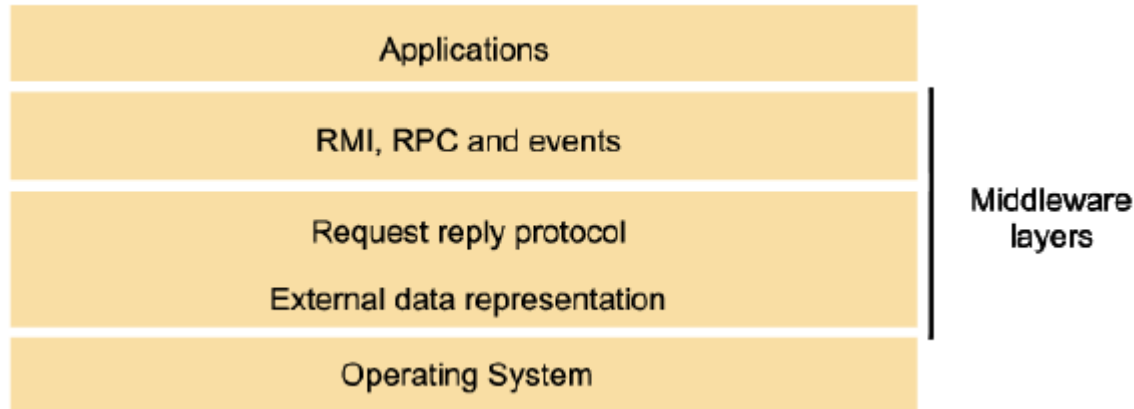
COMP90015 2018 Semester 2
Tutorial 07

Today's Agenda

- Questions of Remote Invocation
- Demonstration of Remote Method Invocation (RMI)

Remote Invocation

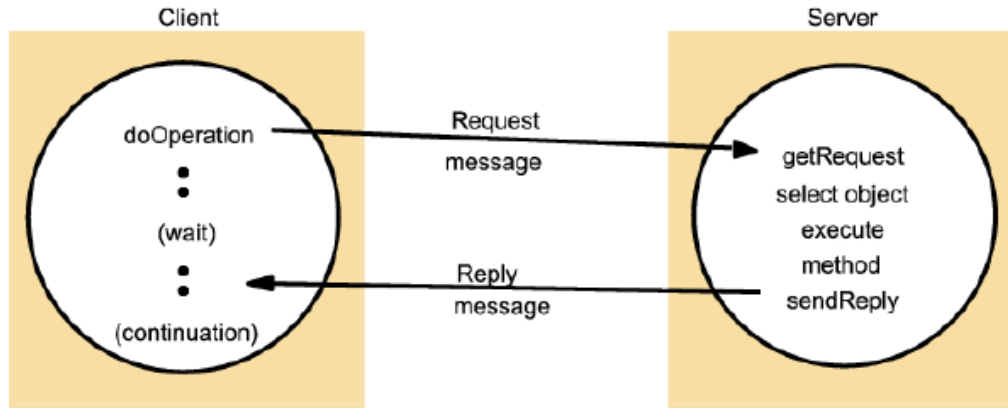
- A set of information exchange protocols at the Middleware layer



- Can you define and explain the procedure of the Request-Reply protocol?

Request-Reply protocol

- Can you describe the procedure of the Request-Reply protocol?



Possible Issues in Request-Reply

- What are the issues that may arise in a request-reply process?
 - Request timeouts
 - Retry request message after timeout
 - Do nothing
 - Reply timeouts
 - Retransmit results after timeout
 - Do nothing
 - Receiving duplicate requests/replies
 - Discard duplicate messages
 - Perform the same action again if the logic is idempotent

Invocation Semantics

- Different issue handling strategies lead to different invocation semantics
- Invocation semantics
 - Define what the client can assume about the execution of the remote procedure
 - Offer different reliability guarantees in terms of the number of times that the remote procedure is executed
- Classification of Request/Reply protocols based on invocation semantics

	Name	Messages sent by		
○ The request protocol (R)		Client	Server	Client
○ The request-reply protocol (RR)	R	Request		
○ The request-reply-acknowledge protocol (RRA)	RR	Request	Reply	
	RRA	Request	Reply	Acknowledge reply

Can you explain the following invocation semantics?

- *Maybe semantics*
- *At-least-once semantics*
- *At-most-once semantics*

Different Invocation Semantics

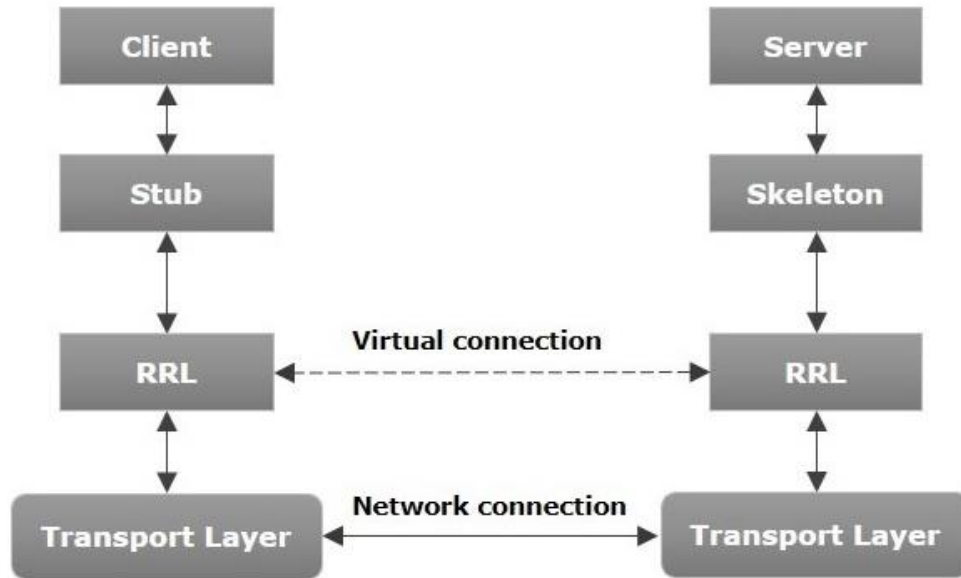
- **Maybe:**
 - The remote procedure call may be executed once or not at all.
 - Unless the caller receives a result, it is unknown as to whether the remote procedure was called.
 - Suitable for applications in which occasional failed calls are acceptable
- **At-least-once:**
 - Either the remote procedure was executed at least once, and the caller received a response, or
 - The caller received an exception to indicate the remote procedure was not executed at all.
 - Suitable for idempotent operations.
- **At-most-once:**
 - The remote procedure call was either executed exactly once, in which case the caller received a response, or
 - It was not executed at all and the caller receives an exception.
 - Suitable for non-idempotent operations.

Figure 5.5
Invocation semantics

<i>Fault tolerance measures</i>			<i>Invocation semantics</i>
<i>Retransmit request message</i>	<i>Duplicate filtering</i>	<i>Re-execute procedure or retransmit reply</i>	
No	Not applicable	Not applicable	<i>Maybe</i>
Yes	No	Re-execute procedure	<i>At-least-once</i>
Yes	Yes	Retransmit reply	<i>At-most-once</i>

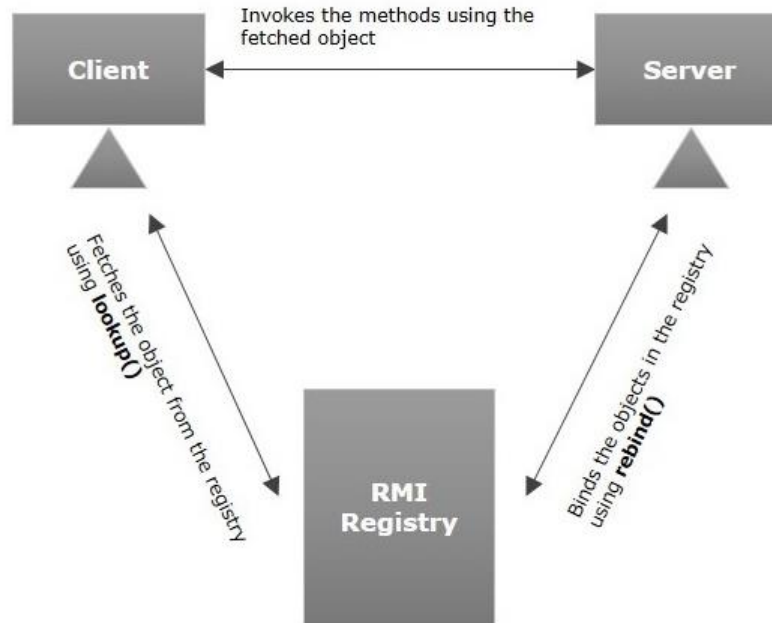
Java RMI

- Explain the steps involved in Java RMI to build a distributed system.

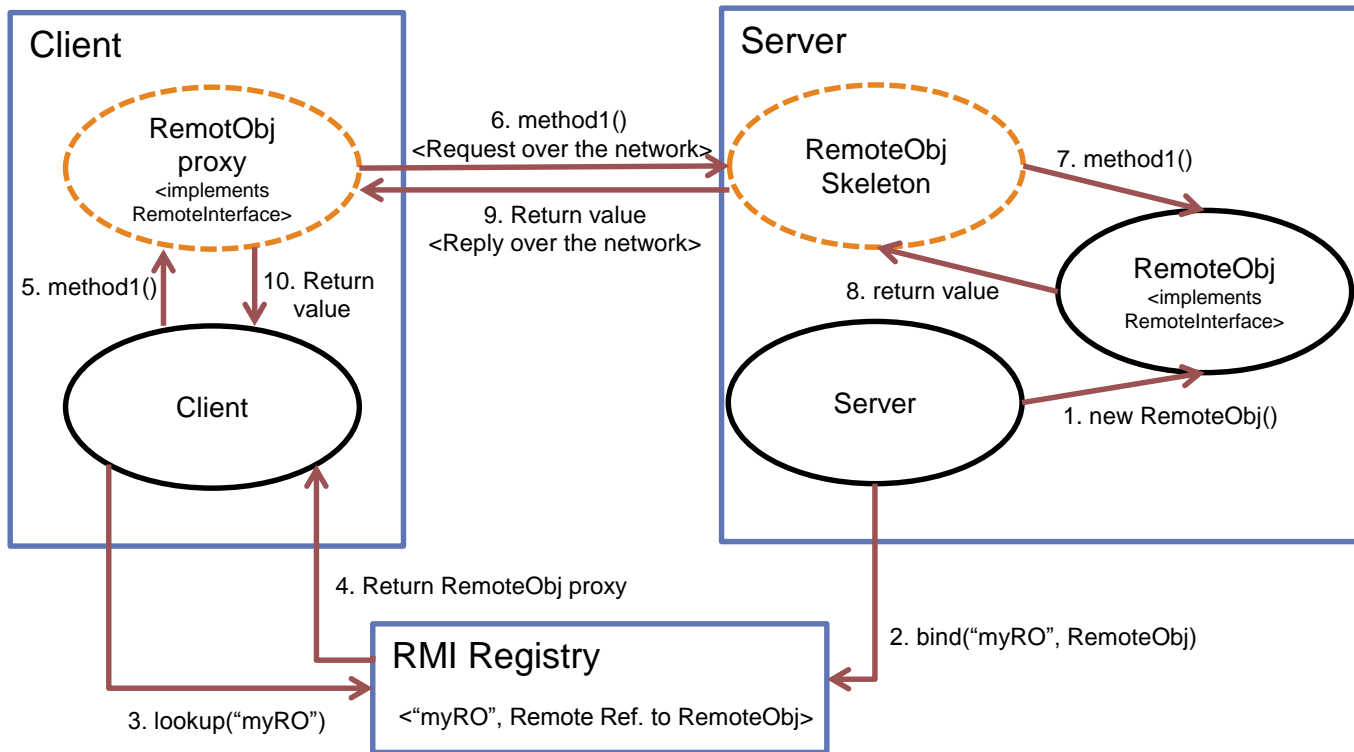


Java RMI

- Explain the steps involved in Java RMI to build a distributed system.



Java RMI Overview



Explain the steps involved in Java RMI to build a distributed system.

- Define the remote interface

- Defines the methods that can be remotely invoked
- Extends `java.rmi.Remote` interface
- All methods throw `java.rmi.RemoteException`
- Needs to be included in the client and in the server programs

- Server program

- Create the remote object
- Implements the remote interface
- Extends `UnicastRemoteObject`
- Implement the actual server
 - Export the remote object into the Java RMI runtime so that it can receive remote calls
 - Locate the `RMIRegistry` and publish the remote object on it (the registry stores <name, remoteObjRef> pairs).

- Client program

- Locate the `RMIRegistry` and lookup the remote object by name
- Invoke methods on the remote object as if it was a local one

Demo Time!

RMI Demo