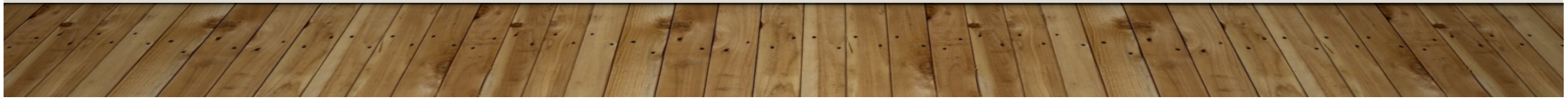
JAVA RMI: CALLBACKS

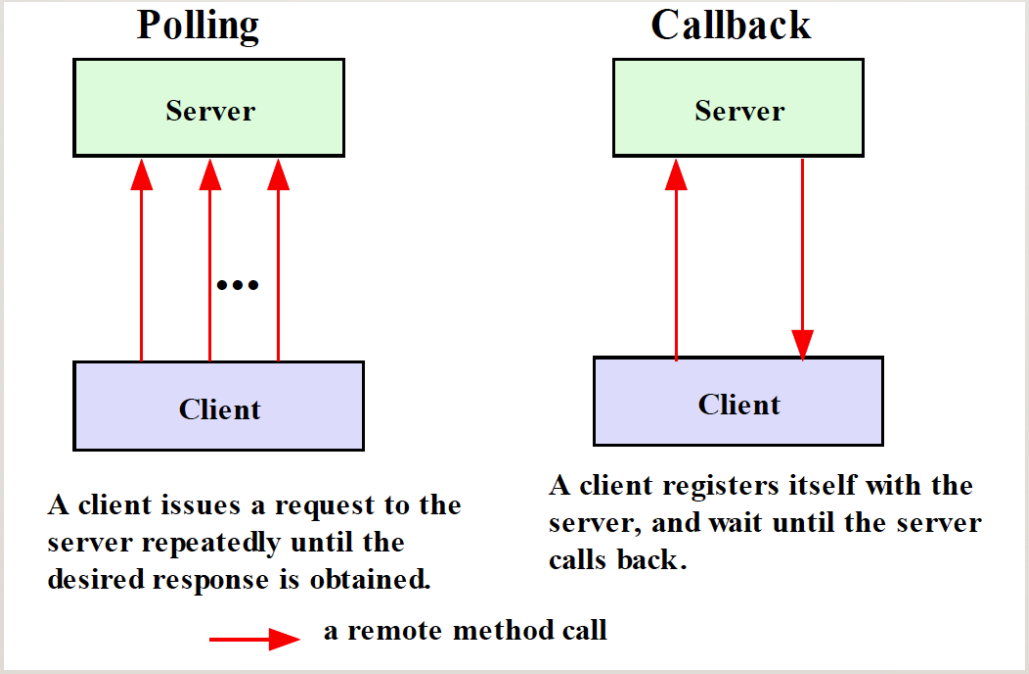
ELOI GABALDON

ELOIGABAL@GMAIL.COM



A NEED FOR CALLBACKS:

- In the client-server model: the communication is initiated by the client; the server waits for the arrival of requests and provides responses.
- Some applications require the server to initiate communication upon certain events.
- With no callbacks, the only option is to poll the server at regular basis. (not efficient)
- With callbacks, the server is able to notify the client when required. (efficient)


Polling
Server
Client
A client issues a request to the server repeatedly until the desired response is obtained.
Callback
Server
Client
A client registers itself with the server, and wait until the server calls back.
a remote method call

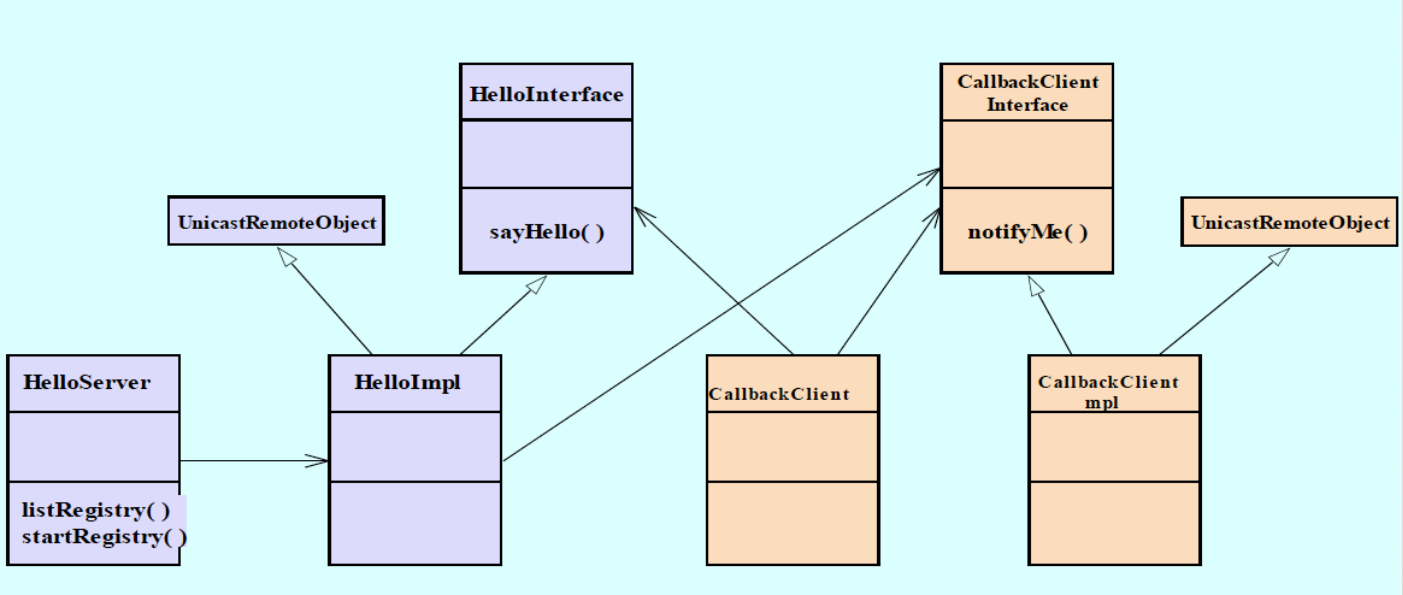
CALLBACK WORKFLOW

- The client registers itself to the server using a remote method.
- The server can then call the remote methods of the registered clients when some events happen.
- Notice the difference with the client-server communication:
  - Client-server: the client searches the server remote objects in the registry directory
  - Server-client: The server knows the client objects by registering them (no registry involved)

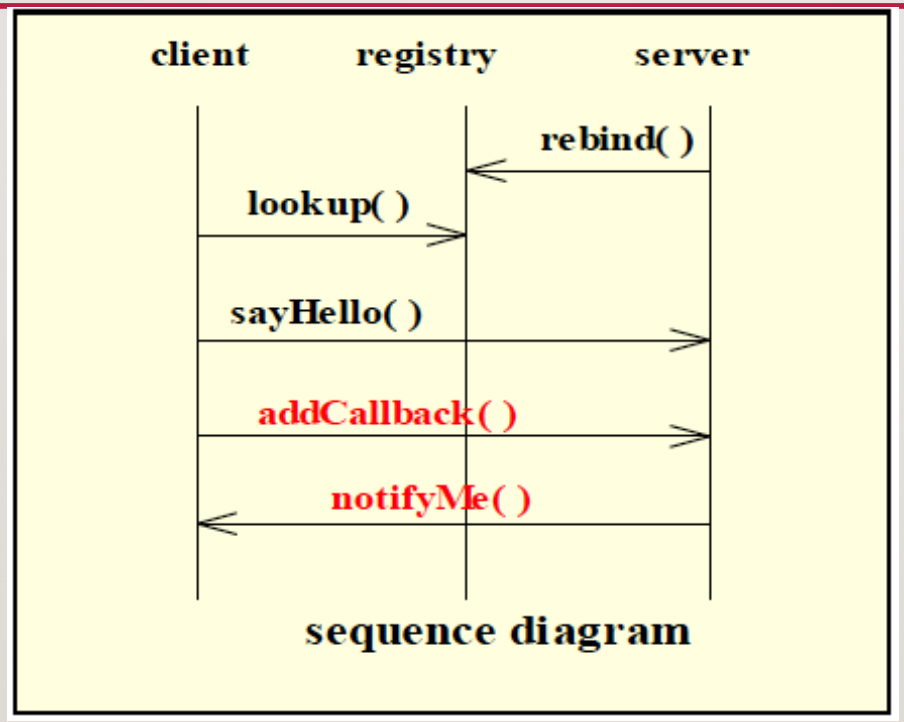
CALLBACK DESIGN

1. Common package:
   - Create a remote interface for the methods of the client object to use as callbacks.
   - Add a register function to the server remote interface that receives a parameter of the client interface.
2. Server side:
   - Implement the register function in the server (should save the client in a list)
   - Also add the information on when to notify the client
3. Client side:
   - Register the client to the server, using the remote "register" function.
   - Implement the client remote interface.

CALLBACK DESIGN



CALLBACK DESIGN



CALLBACK EXAMPLE

- HelloWorld example with Callbacks:
  - Server implements the helloWorldServer to register clients using a function.
  - Client implements the helloWorldClient remote object to receive the callback.
  - Server starts the rmiregistry and exports the server in the rmiregistry.
  - Client obtains the server stub and calls the register function with a helloWorldClient instance.
  - Every 30 seconds, the server calls the notifyHello method on the registered clients.
    - Note this call does not use a registry directory.

CALLBACK EXERCISE

- Define the UML, sequence diagram and program implementation of the following game:
- The server waits for a registration of a client.
- The server selects a random integer in the range 1-10.
- The server notifies the client that the game is about to start.
- The clients then ask the user for input an integer between 1-10, and send this to the server.
- The server collects the number and notifies ”winner” if the client introduced the same number as the server or ”looser” if failed.