Homework assignment two, 5.1, 5.3, 6.9, 6.14, 6.16, 6.21

5.1 The Election interface provides two remote methods:
vote: with two parameters through which the client supplies the name of a candidate (a string) and the 'voter’s number' (an integer used to ensure each user votes once only). The voter’s numbers are allocated sparsely from the range of integers to make them hard to guess.

result: with two parameters through which the server supplies the client with the name of a candidate and the number of votes for that candidate.

Which of the parameters of these two procedures are input and which are output parameters?

Reference Answer
vote: input parameters: name of candidate, voter’s number;
result: output parameters: name of candidate, number of votes

5.3 Define the interface to the Election service in CORBA IDL and Java RMI. Note that CORBA IDL provides the type long for 32 bit integers. Compare the methods in the two languages for specifying input and output arguments.

5.3 Ans.

CORBA IDL:

```idl
interface Election {
    void vote(string name, in long number);
    void result(out string name, out long votes);
};
```

Java RMI

```java
We need to define a class for the result e.g.

class Result {
    String name;
    int votes;
};
```

The interface is:

```java
import java.rmi.*;

public interface Election extends Remote {
    void vote(String name, int number) throws RemoteException;
    Result result () throws RemoteException;
};
```

This example shows that the specification of input arguments is similar in CORBA IDL and Java RMI. This example shows that if a method returns more than one result, Java RMI is less convenient than CORBA IDL because all output arguments must be packed together into an instance of a class.

6.9 Compare the worker pool multi-threading architecture with the thread-per-request architecture.

Reference Answer:
The worker pool architecture saves on thread creation and destruction costs compared to the thread-per-request architecture but (a) the pool may contain too few threads to maximize performance under high workloads or too many threads for practical purposes and (b) threads contend for the shared work queue.

6.14 Explain the factors that motivate the hybrid scheduling approach of the ‘scheduler activations’ design (instead of pure user-level or kernel-level scheduling).

Reference Answer:
A hybrid scheduling scheme combines the advantages of user-level scheduling with the degree of control of allocation of processors that comes from kernel-level implementations. Efficient, custom scheduling takes place inside processes, but the allocation of a multiprocessor’s processors to processes can be globally controlled.
6.21. A client makes RMIs to a server. The client takes 5ms to compute the arguments for each request, and the server takes 10ms to process each request. The local OS processing time for each send or receive operation is .5ms and the network time to transmit each request or reply message is 3ms. Marshalling or unmarshalling takes .5ms per message. Estimate the time taken by the client to generate and return from 2 requests i) if it is single-threaded ii) if it has two threads which can make requests concurrently on a single processor. Is there a need for asynchronous RMI if process are multi-threaded?

Solution : For single threaded client

For each request :

Client :

Process Arguments : 5 ms
marshal args = 0.5 ms
send = 0.5 ms

receive = 0.5 ms
unmarshal = 0.5 ms

Message Transmission = 3 ms

Server :
Receive = 0.5 ms
unmarshal = 0.5 ms
process request : 10 ms
marshal = 0.5 ms
send = 0.5 ms

Message transmission = 3 ms
So total time for a single request is the sum of all the above times = 25 ms
Time for finishing processing two such messages = 2 * 25 = 50 ms