**Student's name:** Hoang Thuy Ha - 20184251

**Class:** ICT02-K63

**Theoretical Exercises**

**Module: Distributed Systems**

**Chapter 1: Introduction and Architectures of Distributed Systems**

**Question 1:** What is the role of middleware in a distributed system?

To increase the distribution transparency that is missing in network operating system.

**Question 2:** Explain what is meant by (distribution) transparency, and give examples of different types of transparency.

Distribution transparency is the phenomenon by which distribution aspects in a system are hidden from users and applications.

Examples include:

* access transparency (hide differences in data representations and how a resource id accessed)
* migration transparency (hide that a resource may move to another location)
* location transparency (hide where a resource is located)
* relocation transparency (hide that the resource may be moved to another location while in use)
* failure transparency (hide the failure and recovery of a resource)
* persistence transparency
* concurrency transparency (hide that a resource may be shared by serveral competitive users)
* replication transparency.

**Question 3:** Why is it sometimes so hard to hide the occurrence and recovery from failures in a distributed system?

It is generally impossible to detect whether a server is actually down, or that is simply slow in responding. Consequently, a system may have to report that a service is not available, although, in fact, the server is just slow.

**Question 4:** Why is it not always a good idea to aim at implementing the highest degree of transparency possible?

Because it may be lead to considerable loss of performance. Users are not willing to accept it.

**Question 5:** What is an open distributed system and what benefits does openness provide?

An open distributed system offers services according to clearly defined rules.

An open system is capable of easily interoperating with other open systems but also allows applications to be easily ported between different implementations of same systems.

**Question 6:** Describe precisely what is meant by a scalable system.

A system is scalable with respect to either its number of components, geographic size, or number and size of administrative domains, if it can grow in one or more of these dimensions without an unacceptable loss of performance.

**Question 7:** Scalability can be achieved by applying different techniques. What are these techniques?

These techniques are distribution, replication, and caching.

**Question 8:** If a client and a server are placed far apart, we may see network latency dominating overall performance. How can we tackle this problem?

A synchronous communication, with a signal to the client that the message has arrived once it reaches the server end.

**Question 9:** What is a three-tiered client-server architecture?

A three-tiered client-server architecture consists of three logical layers, where each layer is, in princible, implemented at a seperate machine. The highest layer consists of a client user interface, the middle layer contains the actual application, and the lowesr layer implements the data that are being used.

**Question 10:** What is the difference between a vertical distribution and a horizontal distribution?

Vertical distribution refers to the distrubution of the different layers in a multitiered architectures across multiple machines. In princible, each layer is implemented on a different machine.

Horizontal distribution deals with the distribution of a single layer across multiple machines.

**Question 11:** In a structured overlay network, messages are routed according to the topology of the overlay. What is an important disadvantage of this approach?

By defination "An overlay network is a virtual network built on top of a physical (underlay) network". While nodes in underlay network are essentially routers but in overlay network node may be a router, a host, a server or even an application. When a message is routed across such network the shortest path between source and destination is chosen. But this logical shortest path many not be the physical best path. The source and recievers may be logically next to each other but physically at the remotest part of the network. So it can unnecessarily delay the message delivery even the other better path is available. Peer to peer applicatons are example of overlay networks.

**Question 12:** Consider a chain of processes P\_1 , P\_2 , ..., P\_n implementing a multitiered client-server architecture. Process P\_i is client of process P\_i+1, and Pi will return a reply to P\_i−1 only after receiving a reply from P\_i+1 . What are the main problems with this organization when taking a look at the request reply performance at process P\_1?

Performance can be expected to be bad for large n. The problem is that each communation between two successive layers is, in princible, between two different machines. Consequently, the performance between P\_1 and P\_2 may also determined by n-2 request\_reply interations between the other layers. Another problem is that id one machine in the chain performs badly or is even temporarily unreachable, the this will immediately degrade th performance at the highest level.

**Question 13:** Considering that a node in CAN knows the coordinates of its immediate neighbors, a reasonable routing policy would be to forward a message to the closest node toward the destination. How good is this policy ?



Consider the route from node (0.2, 0.3) to node (0.9, 0.6). There are several possibilities, but if we want to follow the shortest path according to a Euclidean distance, we should follow the route (0.2,0.3) =>(0.6,0.7) => (0.9,0.6), which has a distance of 0.882. The alternative route (0.2,0.3) => (0.7,0.2) => (0.9,0.6) has a distance of 0.957.

In our example above, it can already be seen that it need not lead to the best route. If node (0.2,0.3) follows this policy for the message destined for node (0.9,0.6), it would send it off to node (0.7,0.2).

**Question 14:** What are the benefits of Microservices architecture compared to monolithic architecture?

Strengths of the Microservice Architecture compared to monolithic architecture.

* **Independent components.** Firstly, all the services can be deployed and updated independently, which gives more flexibility. Secondly, a bug in one microservice has an impact only on a particular service and does not influence the entire application. Also, it is much easier to add new features to a microservice application than a monolithic one.
* **Easier understanding.** Split up into smaller and simpler components, a microservice application is easier to understand and manage. You just concentrate on a specific service that is related to a business goal you have.
* **Better scalability.** Another advantage of the microservices approach is that each element can be scaled independently. So the entire process is more cost- and time-effective than with monoliths when the whole application has to be scaled even if there is no need in it. In addition, every monolith has limits in terms of scalability, so the more users you acquire, the more problems you have with your monolith. Therefore, many companies, end up rebuilding their monolithic architectures.

**Question 15:** Design yourself an e-commerce system using Microservices architecture

