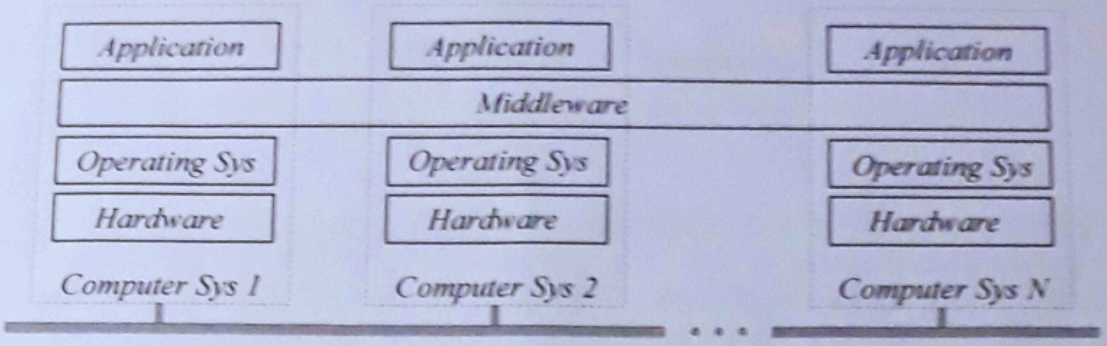
Part A (12 points)

1. The diagram bellow aims to depict in a hierarchical way the different layers of software/hardware modules required to run a distributed application in a parallel machine based on off-the-shelf computer systems.



What is the role played by the middleware layer in this context? Explain why the Java Virtual Machine (JVM) may be thought of as part of the middleware. (***3 points***)

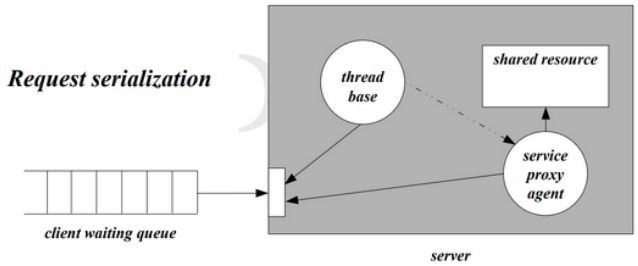
O middleware serve como uma camada de compatibilidade entre os diferentes sistemas, é feita para garantir a portabilidade, compatibilidade e segurança entre esses sistemas. Ao ser implementado cria uma camada de abstração do hardware, sistema de rede e sistema operativo. A JVM pode ser considerada um middleware porque garante que os seus programas compilados são portáveis entre sistemas, gatantindo assim compatibilidade e criando deste modo código independente da plataforma de desenvolvimento.

2. Describe schematically, adding the comments you deem appropriate about their functionality, the advantages and disadvantages, of the three variants of the client-server model that were studied. Can they be implemented using either the message passing or the remote objects paradigms, or are they specific to one of these paradigms? Present clearly your reasoning. (*3 points*)

O modelo cliente-servidor tem múltiplas variantes, entre elas estão:

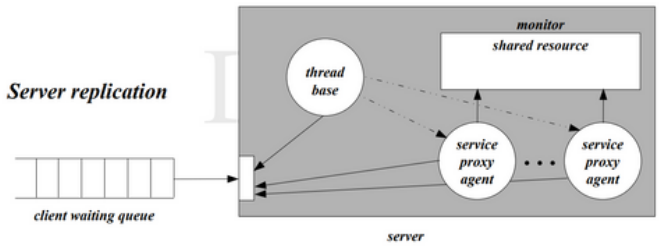
TIPO 1 : Request serialization

Neste modelo a implementação é bastante simplificada, não necessitando de mecanismos de exclusão mútua, no entanto, quando comparado às restantes alternativas não é tão eficaz pois não dá uso aos tempos mortos de processamento, podendo isto causar situações de busy waiting.

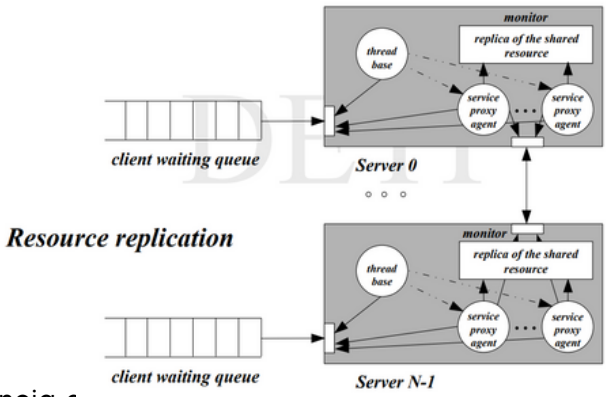


TIPO 2: Server Replication

Neste modelo dada a concorrência o desempenho é melhor comparativamente ao modelo anterior, uma vez que este possibilita o atendimento de múltiplos clientes em simultâneo. No entanto, necessita que sejam implementados mecanismos de exclusão mútua, o que leva a um aumento da complexidade no que diz respeito á gestão das múltiplas instâncias.



TIPO 3: Resource Replication

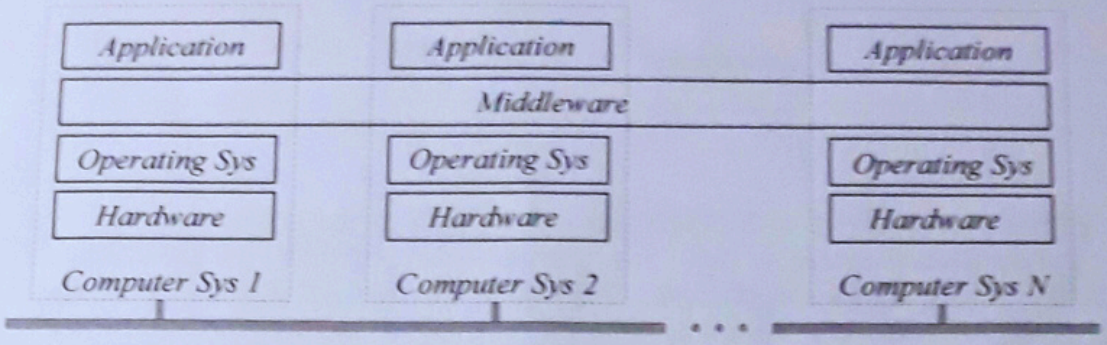


Relativamente a este modelo, é possível verificar uma maior tolerância a falhas, tendo em conta que a falha de 1 ou mais servidores não afetará o serviço, visto que a distribuição da carga é feita por múltiplas máquinas. Por outro lado, esta implementação leva a uma elevada complexidade no momento da replicaçao dos daods, o que pode gerar alguma latência na distribuição dos pedidos.

Tanto o paradigma de Passagem de Mensagens como o de chamada de objetos remotos são versáteis, podendo ser aplicados aos 3 modelos anteriores, no entanto, tendo em conta o objetivo do sistema é possível selecionar a implementação mais adequada, por exemplo:

Num sistema de comunicação assíncrona entre componentes desacopladas, a passagem de mensagens é mais útil. Por outro lado, quando se tratam de sistemas onde são necessárias invocações diretas e síncronas de métodos remotos, o paradigma de chamada de objetos remotos torna-se o mais indicado.

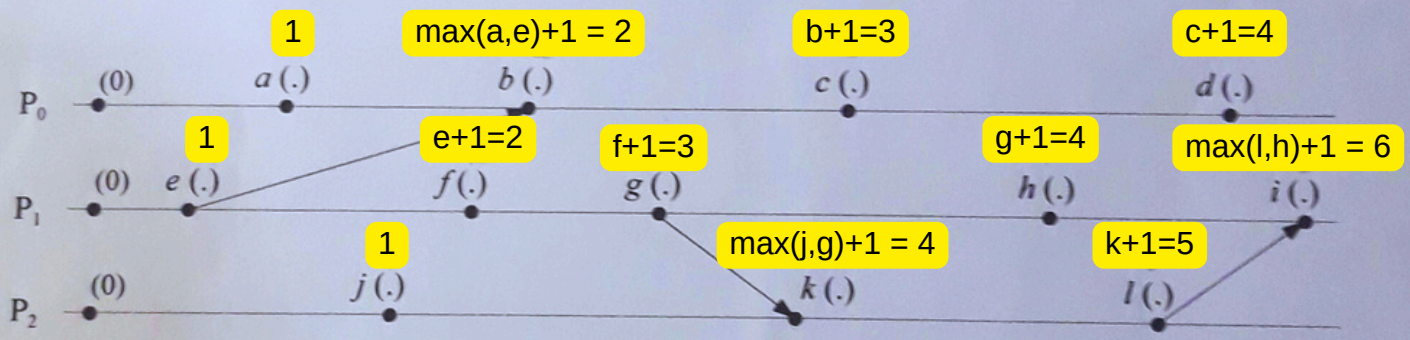
modules required to run a distributed application in a parallel machine based on off-the-shelf computer systems.



What is the role played by the middleware layer in this context? Explain why the Java Virtual Machine (JVM) may be thought of as part of the middleware. (***3 points***)

2. Describe schematically, adding the comments you deem appropriate about their functionality, the advantages and disadvantages, of the three variants of the client-server model that were studied. Can they be implemented using either the message passing or the remote objects paradigms, or are they specific to one of these paradigms? Present clearly your reasoning. (***3 points***)

3. The diagram depicts the *time* evolution of three processes whose local clocks are scalar logical clocks synchronized according to the Lamport algorithm. (***3 points***)



i. Assign to the different events, specified by small letters (*a* ... *l*), their associated time stamp.

ii. Consider the following pairs of events: *a-h*, *a-l*, *e-i*, *i-j*, *d-e* and *f-l*. Representing concurrent events by '| |' and events connected by a causality nexus by '–>', show how would you classify each pair.

Pairs: (a-h), (a-l), (e-i), (i-j), (d-e)

ts(a) = 1
ts(h)= 4
ts(a)<ts(h)
Como não há a -> h, então a || h

ts(a) = 1
ts(l)= 5
ts(a)<ts(l)
Como não há a -> l, então a || l

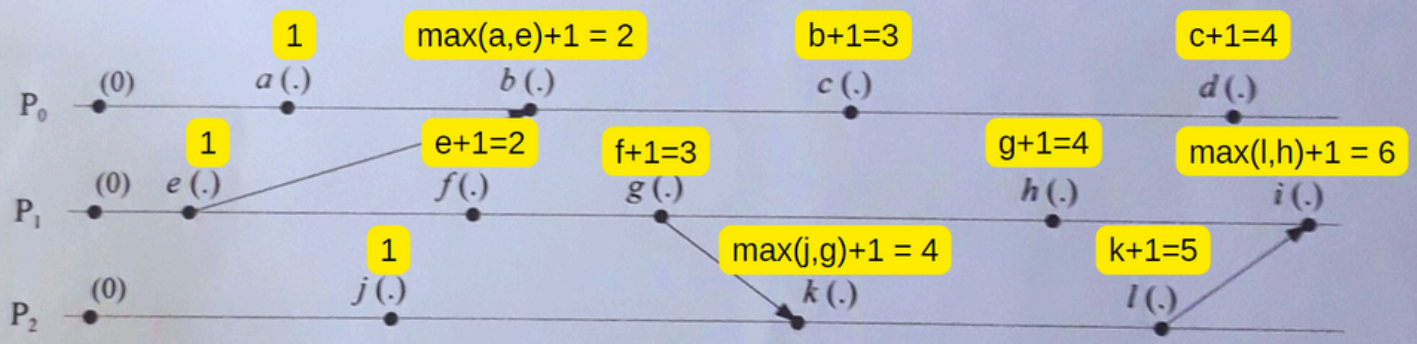
ts(e) = 1
ts(i)=6
ts(e)<ts(i)
Como existe e->i então, e->i

ts(i) = 6
ts(j)= 1
ts(i)>ts(j)
Então j->k->l->i, existe i->j

ts(d) = 4
ts(e)=1
ts(d)>ts(e)
Então e->b->c->d, existe d->e

ts(f) = 2
ts(l)= 5
ts(f)<ts(l)
Então f->g->k->l, existe f->l

3. The diagram depicts the *time* evolution of three processes whose local clocks are scalar logical clocks synchronized according to the Lamport algorithm. (***3 points***)



i. Assign to the different events, specified by small letters (*a* ... *l*), their associated time stamp.

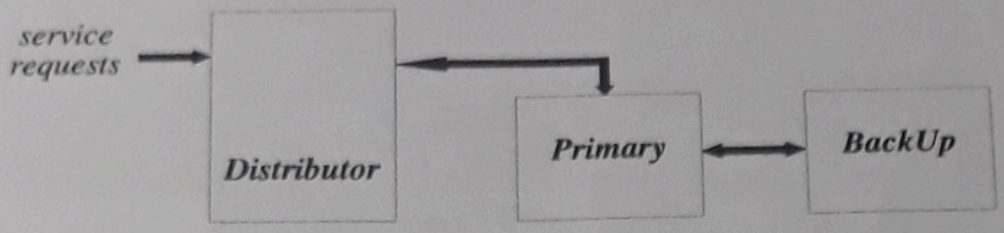
ii. Consider the following pairs of events: *a-h*, *a-l*, *e-i*, *i-j*, *d-e* and *f-l*. Representing concurrent events by '| |' and events connected by a causality nexus by '–>', show how would you classify each pair.

iii. What is required to total order a group of events time stamped by their logical scalar clocks synchronized according to Lamport algorithm? Present clearly your reasoning.

Para que seja possível ordenar totalmente um grupo de eventos previamente marcados com timestamps de acordo com o algoritmo de Lamport, já estando estes parcialmente ordenados, estes eventos são passados a um processo de 'desempate' com os ids de processo respetivos onde todos os eventos são comparados. O desempate é feito da seguinte forma:

Se 2 eventos possuem timestamps iguais, ao comparar os ids dos processos é possível ordenar os mesmos. Consideremos os eventos A(ta, Pa) e B(tb,Pb):
ta = tb
mas Pa é obrigatoriamente diferente de Pb, pelo que, por exemplo, se Pa>Pb, então A sucede antes de B e assim se torna possível ordenar totalmente um grupo de eventos.

4. The picture bellow depicts a simple solution to improve *service availability* in a client-server model that is known as a *primary-backup topology*.



The role of the *distributor* is to forward service requests to the *primary server*. The *backup server* is updated by the *primary server* so that their internal states are continuously synchronized. Assume that the reliability of the hardware platform that takes the role of *distributor* is much higher than the reliability of the hardware platforms that that take the role of *primary* or *backup servers*.
Which hardware platform is *primary* is decided in a dynamical way: i) it is the first to register in the *distributor* after reboot; ii) it is the *backup server*, thereby changing its role, if *primary* fails.
What kind of data the *distributor* and the *primary* have to keep internally in order for operations to be carried out as expected? How do the *distributor* and the *backup server* become aware of a *primary* failure? What operations have to be carried out to overcome this problem? Assume that no messages are lost and message transmission time is limited above. (***3 points***)

O distributor e o backup server necessitam de manter tanto os dados do estado do sistema quanto os dados relativos aos pedidos, juntamente com os metadados e os dados de configuração do sistema que regem o comportamento do sistema.
O backup server pode monitorizar a disponibilidade do servidor primario através de mensagens periodicas de heartbeat, assumindo como sendo uma falha caso exceda o timeout predefinido.
Tanto o distributor como o backup tomam conhecimento de uma falha no Primary server através do heartbeat transmitido ou verificações periódicas.
Para contornar uma falha no sistema, existem varias abordagens:

- **Failover**
  - O backup server assume o papel de primary server e começa a processar os pedidos dos clientes, ao utilizar a sua réplica atualizada do estado do sistema e dos dados.
- **Restaurar o Primary Server**
  - Uma vez que o Primary Server é restaurado, é necessário que ocorra uma sincronização do mesmo com os seus dados e estado com o servidor de backup de modo a garantir que possui as informações mais recentes.
- **Failback**
  - Depois do Primary server ter sido restaurado e sincronizado com base nos dados retidos pelo Backup server, ele pode então retomar o seu papel de Primary server, enquanto que o backup server volta ao seu modo wait.