Sistemas Distribuídos
Teste 2

9 de Julho de 2018

Part A (12 points)

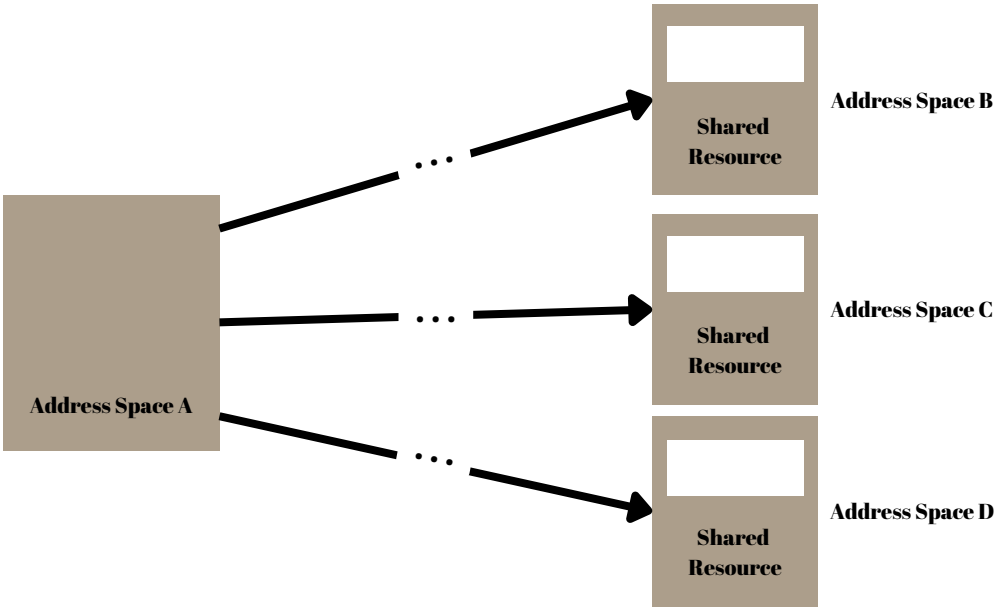
1. What is meant by *transparency* in the context of distributed systems? Give examples, at least five, of the modes transparency may assume. Justify clearly your claims.

Num contexto de sistemas distribuidos o conceito de transparencia é referente á capacidade que o sistema apresenta de esconder a complexidade da distribuiçao dos seus componentes aos users e das aplicaçoes. Ou seja, um sistema que se considera transparente faz com que os users e as apps interajam com ele como se fosse um unico sistema independente da localizacao fisica dos componentes que integram o sistema em causa ou rede subjacente.

O grau de transparencia de um sistema distribuido é um recurso que expressa o maior ou menor sucesso em mascarar a complexidade subjacente. Existem então 5 tipos de graus de transparencia:

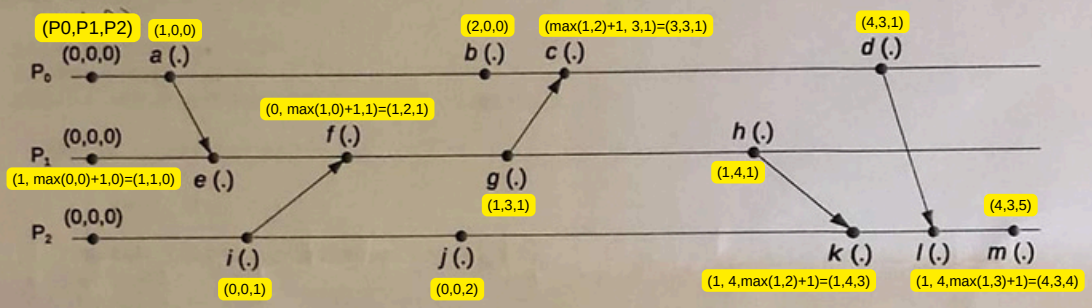
- **Access Transparency:**
  - Sucede quando as mesmas operações são executadas para aceder a recursos locais e remotos.
- **Position Transparency:**
  - Sucede quando a transparência de acesso a recursos é efetuada sem o conhecimento da sua localização fisica precisa ou da rede.
- **Network Transparency:**
  - Ocorre quando a transparencia de acesso e posição existem em simultâneo.
- **Mobility Transparency:**
  - Acontece quando o acesso da localização dos recursos pelo cliente pode alterar dentro do sistema sem que a operação que está a ser executada seja prejudicada.
- **Migration Transparency:**
  - Dá-se quando os recursos são movidos sem que os acessos a esses mesmos recursos sejam afetados.

2. Take *access* to *remote objects* as the communication paradigm which was chosen to establish a client-server model among processes residing in different nodes of a distributed system. Draw a diagram that describes the distribution of the components of such a model, at the server side, by different computer systems, identifying each of them and explaining their role in the interaction. Assume that there are three remote objects positioned in geographically distant regions.
Refer in this context to what one means by *network transparency*. How can it be enforced in Java?



O Java aplica o modo de transparência na rede privada devido ao RMI Registry, onde armazena referencias para objetos remotos sem que o user tenha essa perceção ou sequer saiba a sua localização especifica, ou se está armazenado local ou remotamente (transparência de rede).

3. The schematics below depicts the *temporal* evolution of three processes whose local clocks are vector logical clocks synchronized according to the vector algorithm.



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

ii. Take from the schematics three examples of pairs of concurrent events (||) and list the longest sequence of sequential ones (->).

3.ii)

Concorrent Pairs:

Não existe conexão direta entre o par (a,j) , pelo que se trata de um par concorrente: a || j

Não existe conexão direta entre o par (b,i) , pelo que se trata de um par concorrente: b || i

Não existe conexão direta entre o par (d,h) , pelo que se trata de um par concorrente: d || h

Longest Sequence of Sequential Pairs:

a -> e

e -> f

f -> g

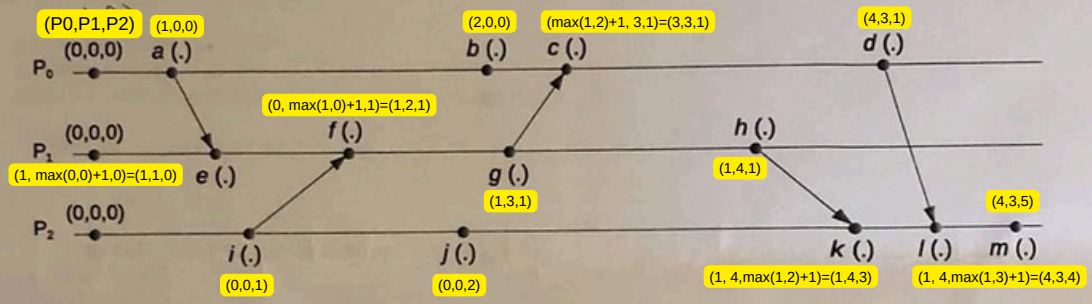
g -> c

c -> d

d -> l

l- > m

3. The schematics below depicts the *temporal* evolution of three processes whose local clocks are vector logical clocks synchronized according to the vector algorithm.



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

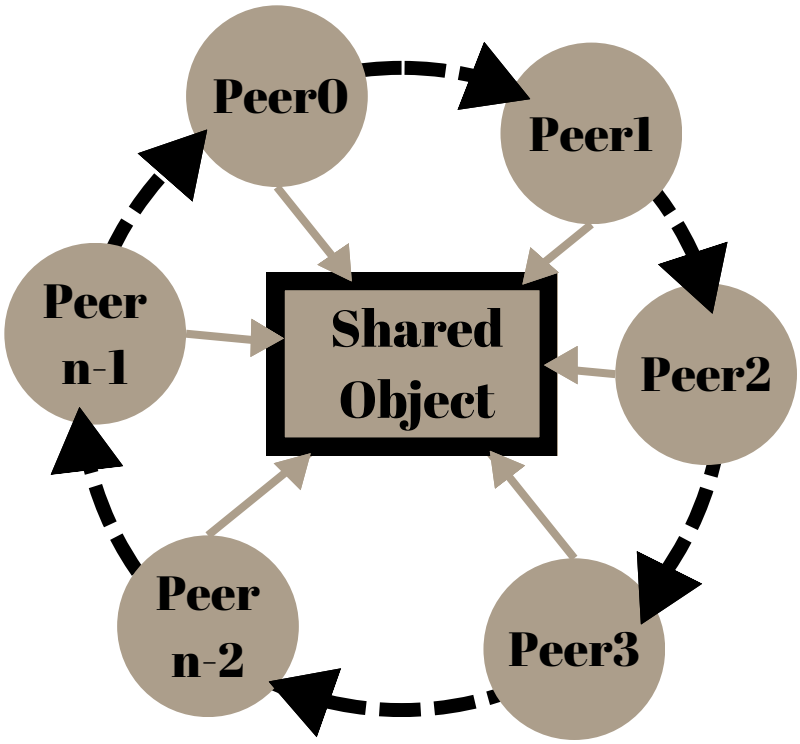
ii. Take from the schematics three examples of pairs of concurrent events (||) and list the longest sequence of sequential ones (->).

iii. Identify all the events that belong to the *past* of event *g*? Consider next all the events that belong to its *future*. Justify your reasoning clearly.

Os eventos anteriores ao evento g serão aqueles que possuirem vetores menores. Considerando o vetor de g = (1,3,1):

- Process 0 Events
  - a = (1,0,0) < g => evento anterior a g
  - b = (2,0,0) < g => evento anterior a g
  - c = (3,3,1) > g => evento posterior a g
  - d = (4,3,1) > g => evento posterior a g
- Process 1 Events
  - e = (1,1,0) < g => evento anterior a g
  - f = (1,2,1) < g => evento anterior a g
  - h = (1,4,1) > g => evento posterior a g
- Process 2 Events
  - i = (0,0,1) < g => evento anterior a g
  - j = (0,0,2) < g => evento anterior a g
  - k = (1,4,3) > g => evento posterior a g
  - l = (4,3,4) > g => evento posterior a g
  - m = (4,3,5) > g => evento posterior a g

4. In *peer-to-peer communication*, a virtual ring based on the individual *id* was established for communication through message passing among the active processing nodes of a distributed system. Describe in detail the algorithms which allow the insertion and the retrieval of a processing node in the ring. Draw diagrams for each case to help clarify your description. Assume that there are neither message loss, nor node crashing.



O método peer-to-peer parte do princípio que os Peers, neste caso, seriam Processos, designados Pi (i = 0, 1, ...), formando um anel lógico virtual em termos de comunicação.

Cada processo envia mensagens apenas para o seu sucessor. Existe apenas uma mensagem em circulação no anel, a mensagem token. Caso o processo Pi pretenda aceder ao objeto partilhado, deverá aguardar pela mensagem token e, ao recebê-la, assinalar que vai proceder ao seu acesso. Após terminar o acesso, passa a mensagem token ao processo seguinte.

Se um processo não pretender aceder ao objeto, ao receber a mensagem token, passa-a diretamente ao processo seguinte.