Nome: _____ **Número:** _____

MESTRADO INTEGRADO EM ENGENHARIA INFORMÁTICA
UNIVERSIDADE DO MINHO

Sistemas Distribuídos

Teste[1]

11 de Janeiro de 2019 Duração: 2h00m

I

1 Distinga *escala geográfica* de *escala numérica* em sistemas distribuídos e identifique técnicas usadas para as atingir.

Escala geográfica corresponde à distância máxima entre os recursos (entre os nós). Escala numérica corresponde ao número de utilizadores e/ou processos que compõem o sistema. Técnicas usadas para atingir esta escalabilidade são: mascarar a latência entre os recursos, evitando esperas ativas; dividir o esforço da computação; replicação dos dados em vários nós / caching.

2 Defina *transparência de acesso* e explique em que medida é que a *invocação remota* (RPC) contribui para a obter.

A Remote Procedure Call (RPC), funciona baseada numa comunicação entre cliente e servidor, através de um mecanismo de procedure call. Neste mecanismo, um cliente inicia um processo, que é enviado para a stub, criando uma mensagem, que é enviada para o seu OS. O OS do cliente responsabiliza-se por transmitir a mensagem para o OS remoto que a envia para o seu stub e a descompacta, enviando para ser processada no Servidor. O Servidor executa o processo e efetua todos os passos anteriores até o cliente receber o resultado descompactado.

Os stubs são usados como referências para acesso aos dados, tanto no cliente como no servidor remoto, contribuindo assim para a transparência de acesso.

3 Explique uma forma de mitigar a incerteza quanto ao tempo de transmissão de mensagens para conseguir sincronizar relógios em sistemas distribuídos.

[1] Cotação — 10+10

II

Considere um serviço de transferência de passageiros. Assuma 5 terminais, um percurso circular (ou seja, 1–2–3–4–5–1 repetidamente), e um shuttle com capacidade para 30 passageiros. O shuttle pára em cada terminal para permitir saída e entrada de passageiros. Por questões de rentabilidade, o shuttle só viaja com pelo menos 10 passageiros, esperando por mais se necessário. A viagem entre terminais demora 5 minutos. Cada passageiro utiliza uma aplicação (cliente) que interage com o servidor que controla o sistema, devendo permitir: o passageiro requisitar ao servidor uma viagem entre o terminal onde está (origem) e o terminal de destino; o servidor informar o passageiro que pode entrar no shuttle; o servidor informar o passageiro que chegou ao seu destino.

1 Apresente uma classe (para ser usada no servidor) que implemente a interface abaixo, tendo em conta que os seus métodos serão invocados num ambiente multi-threaded.

```
interface Controlador {
  void requisita_viagem(int origem, int destino);
  void espera(int destino);
}
```

O método `requisita_viagem` deve bloquear até o passageiro poder ser informado que pode entrar no shuttle (o shuttle chegou à origem e há lugar disponível). O método `espera` deve bloquear até o shuttle ter chegado ao terminal `destino`. Caso haja mais passageiros num terminal do que lugares disponíveis, os passageiros devem entrar por ordem de requisição. Nota: esta interface deve considerar o uso dos seus métodos apenas por threads que representam passageiros; considere a possibilidade de criar threads auxiliares na sua implementação.

2 Implemente o programa servidor usando threads, sockets TCP, e a classe desenvolvida na pergunta anterior.