CURSO**: Tecnologia em Análise e Desenvolvimento de Sistemas (TADS)**
DISCIPLINA**: Sistemas Distribuídos**
Prof. Claudio Martins

TURMA: _____
ALUNO: _____

Exercício de Fixação - 1ª Avaliação
Fonte de pesquisa básica*: Livro do Coulouris, capítulo 1 (Caracterização de Sistemas Distribuídos)*
Disponível em https://drive.google.com/open?id=0B4HFHo9_RMpBWjVDOEFRVWxaZGM (pasta "materiais-ebooks-papers")

1) Defina, com base nos conceitos apresentados em sala de aula e na sua experiência, o que vem a ser um sistema distribuído.

2) Cite e explique resumidamente, três fatores que contribuíram para a disseminação dos sistemas distribuídos.

3) Cite cinco tipos de recurso de *hardware* e cinco tipos de recursos de dados ou de *software* que possam ser compartilhados com sucesso. Dê exemplos práticos de seu compartilhamento em sistemas distribuídos.

4) Explique a relação tecnológica e comparação entre Redes de Computadores e Sistemas Distribuídos.

5) Cite três exemplos de sistemas distribuídos.

6) Cite e explique três vantagens dos sistemas distribuídos sobre os sistemas centralizados.

7) Cite e explique duas desvantagens na adoção de sistemas distribuídos.

8) Defina e explique as características básicas de Sistemas Distribuídos:
A) Compartilhamento de recursos
B) Extensibilidade (*openness*)
C) Concorrência
D) Escalabilidade (crescimento gradativo suave)
E) Tolerância a falhas
F) Transparência

9) Com base na formulação da Internet como exemplo de um grande sistema distribuído, explique o conceito de "serviço" e cite três exemplos de serviços disponibilizados na Internet.

10) O que vem a ser conhecida como *computação ubíqua* (ou pervasiva)?

11) Explique o termo "*computação em nuvem*", no contexto de sistemas distribuídos.

12) Compare e contraste a computação em nuvem com a computação cliente-servidor mais tradicional. O que há de novo em relação à computação em nuvem como conceito?

13) Use a World Wide Web (WWW) como exemplo para ilustrar o conceito de compartilhamento de recursos, cliente e servidor. Quais são as vantagens e desvantagens das tecnologias básicas HTML, URLs e HTTP para navegação em informações? Alguma dessas tecnologias é conveniente como base para a computação cliente-servidor em geral?

14) Explique o que é um *middleware*.

15) Liste os três principais componentes de software que podem falhar quando um processo cliente chama um método em um objeto servidor, dando um exemplo de falha em cada caso. Sugira como os componentes podem ser feitos de modo a tolerar as falhas uns dos outros.

16) Os recursos na World Wide Web e outros serviços são nomeados por URLs. O que denotam as iniciais URL? Dê exemplos de três diferentes tipos de recursos da Web que podem ser nomeados por URLs.

17) Cite um exemplo de URL HTTP. Liste os principais componentes de um URL HTTP, dizendo como seus limites são denotados e ilustrando cada um, a partir de seu exemplo. Até que ponto um URL HTTP tem transparência de localização?

Questões sobre Arquitetura, Modelos e Comunicação entre Processos

1) (2.1) Dê três exemplos específicos e contrastantes dos níveis de heterogeneidade cada vez maiores experimentados nos sistemas distribuídos atuais, conforme definido na Seção 2.2. *página 39*

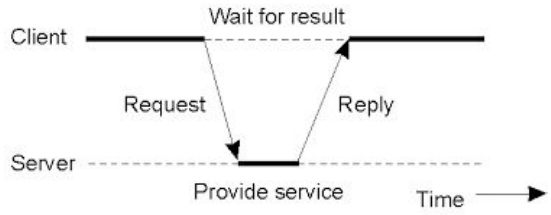
2) (2.2) Quais problemas você antevê no acoplamento direto entre entidades que se comunicam, que está implícito nas estratégias de invocação remota? Consequentemente, quais vantagens você prevê a partir de um nível de desacoplamento, conforme o oferecido pelo não acoplamento espacial e temporal? Nota: talvez você queira rever sua resposta depois de ler os Capítulos 5 e 6. *página 43*

3) (2.3) Descreva e ilustre a arquitetura cliente-servidor de um ou mais aplicativos de Internet importantes (por exemplo, Web, correio eletrônico ou *news*). Elabore também um diagrama para auxiliar a explicação. *página 46*
4) Defina como é organizado um sistema em três camadas físicas. Faça um diagrama relacionando as camadas existentes.
5) (2.9) Considere uma empresa de aluguel de carros hipotética e esboce uma solução de três camadas físicas para seu serviço distribuído de aluguel de carros. Use sua resposta para ilustrar vantagens e desvantagens de uma solução de três camadas físicas, considerando problemas como desempenho, mudança de escala, tratamento de falhas e manutenção do *software* com o passar do tempo. *página 53*
6) (2.6) Frequentemente, os computadores usados nos sistemas *peer-to-peer* são computadores *desktop* dos escritórios ou das casas dos usuários. Quais são as implicações disso na disponibilidade e na segurança dos objetos de dados compartilhados que eles contêm e até que ponto qualquer vulnerabilidade pode ser superada por meio da replicação? *páginas 47, 48*
7) (2.7) Liste os tipos de recurso local vulneráveis a um ataque de um programa não confiável, cujo *download* é feito de um *site* remoto e que é executado em um computador local. *página 49*

8) (2.14) Considere dois serviços de comunicação para uso em sistemas distribuídos assíncronos. No serviço A, as mensagens podem ser perdidas, duplicadas ou retardadas, e somas de verificação (*checksums*) se aplicam apenas aos cabeçalhos. No serviço B, as mensagens podem ser perdidas, retardadas ou entregues rápido demais para o destinatário manipulá-las, mas sempre chegam com o conteúdo correto. Descreva as classes de falha exibidas para cada serviço. Classifique suas falhas de acordo com seu efeito sobre as propriedades de validade e integridade. O serviço B pode ser descrito como um serviço de comunicação confiável? *páginas 67*, *71*

9) (2.15) Considere dois processos, X e Y, que utilizam o serviço de comunicação B do Exercício 2.14 para se comunicar entre si. Suponha que X seja um cliente e que Y seja um servidor e que uma *invocação* consiste em uma mensagem de requisição de X para Y, seguida de Y executando a requisição, seguida de uma mensagem de resposta de Y para X. Descreva as classes de falha que podem ser exibidas por uma invocação. *página 67*
10) Quais são as vantagens do uso de “threads” em vez de Processos em Sistemas Distribuídos?
11) Defina o que é comunicação persistente, transiente, assíncrona e síncrona.
12) No funcionamento de sistemas distribuídos, um parâmetro importante na avaliação do desempenho na

comunicação entre processos é conhecido por tempo de latência. Com base na figura a seguir, explique esse conceito.



13) (4.1) É concebivelmente útil que uma porta tenha vários receptores? *páginas 148*

14) (4.2) Um servidor cria uma porta que ele utiliza para receber pedidos dos clientes. Discuta os problemas de projeto relativos ao relacionamento entre o nome dessa porta e os nomes usados pelos clientes. *páginas 148*
15) O que seria uma chamada remota de procedimento (RPC) e qual a vantagem do seu uso em Sistemas Distribuídos?
16) Qual a utilidade de uma linguagem de definição de interface (IDL – Interface Definition Language)?
17) O que seria um procedimento remoto idempotente?
18) Por que serviços de comunicação em nível de transporte (por exemplo, usando "sockets") frequentemente são inadequados para construir aplicações distribuídas?

19) (4.12) Por que dados binários não podem ser representados diretamente em XML, por exemplo, como valores em Unicode? Os elementos XML podem transportar *strings* representados como *base64*. Discuta as vantagens ou desvantagens de usar esse método para representar dados binários. *página 164*
20) (4.13) Defina uma classe cujas instâncias representem referências de objeto remoto. Ela deve conter informações semelhantes àquelas mostradas na Figura 4.13 e deve fornecer métodos de acesso necessários para os protocolos de nível mais alto (consulte requisição-resposta, no Capítulo 5, para ver um exemplo). Explique como cada um dos métodos de acesso será usado por esse protocolo. Dê uma justificativa para o tipo escolhido para a variável de instância que contém informações sobre a interface do objeto remoto. *página 168*
21) O problema de heterogeneidade entre computadores quanto a representação de dados manifesta-se quando se deseja transmitir dados em sistemas distribuídos. Neste caso, a abordagem de *Marshalling* (ou "empacotamento") é o processo normalmente usado. Explique como esse processo (*Marshalling*) funciona e quais as duas técnicas básicas usadas para implementá-lo.

22) (5.1) Defina uma classe (em Java) cujas instâncias representem as mensagens de requisição-resposta, conforme ilustrado na Figura 5.4. A classe deve fornecer dois construtores, um para mensagens de requisição e o outro para mensagens de resposta, mostrando como o identificador de requisição é atribuído. Ela também deve fornecer um método para empacotar a si mesma em um vetor de bytes e desempacotar um vetor de bytes em uma instância. (*página 188)*

messageType	*int (0=Request, 1=Reply)*
requestId	*int*
remoteReference	*RemoteRef*
OperationId	*int ou operação*
arguments	*// vetor de bytes*

Figura 5.4 *Estrutura da mensagem de requisição-resposta.*

23) (5.2) Programe cada uma das três operações do protocolo de requisição-resposta da Figura 5.3 usando comunicação UDP, mas sem adicionar quaisquer medidas de tolerância a falhas. Você deve usar as classes que definiu no capítulo anterior para referências de objeto remoto (Exercício 4.13) e acima para mensagens de requisição-resposta (Exercício 5.1). (Ver página 187)

public byte[ ] doOperation (RemoteRef s, int operationId, byte[ ] arguments)
Envia uma mensagem de requisição para o servidor remoto e retorna a resposta.
Os argumentos especificam o servidor remoto, a operação a ser invocada e
os argumentos dessa operação.

public byte[ ] getRequest ( );
Lê uma requisição do cliente por meio da porta do servidor.

public void sendReply (byte[ ] reply, InetAddress clientHost, int clientPort);
Envia a mensagem de resposta *reply* para o cliente como seu endereço de Internet e porta.

Figura 5.3 *Operações do protocolo de requisição-resposta.*

24) (5.3) Forneça um esboço da implementação de servidor, mostrando como as operações getRequest e *sendReply* são usadas por um servidor que cria uma nova *thread* para executar cada requisição do cliente. Indique como o servidor copiará o *requestId* da mensagem de requisição na mensagem de resposta e como obterá o endereço IP e a porta do cliente. *página 187*

25) (5.11) Uma interface *Election* fornece dois métodos remotos:

***vote**:* este método possui dois parâmetros por meio dos quais o cliente fornece o nome de um candidato (um *string*) e o "número do votante" (um valor inteiro usado para garantir que cada usuário vote apenas uma vez). Os números dos votantes são alocados esparsamente a partir do intervalo de inteiros para torná-los difíceis de adivinhar.
result: este método possui dois parâmetros com os quais o servidor fornece para o cliente o nome de um candidato e o número de votos desse candidato.

Quais dos parâmetros desses dois métodos são de *entrada* e quais são parâmetros de *saída*? *página 195*

26) (5.22) Um cliente faz invocações a método remoto a um servidor. O cliente demora 5 milissegundos para computar os argumentos de cada requisição, e o servidor demora 10 milissegundos para processar cada requisição. O tempo de processamento do sistema operacional local para cada operação de envio ou recepção é de 0,5 milissegundos, e o tempo que a rede leva para transmitir cada mensagem de requisição ou resposta é de 3 milissegundos. O empacotamento ou desempacotamento demora 0,5 milissegundos por mensagem.

Calcule o tempo que leva para o cliente gerar e retornar duas requisições:
(i) se ele tiver só uma thread;
(ii) se ele tiver duas threads que podem fazer requisições concorrentes em um único processador.
Você pode ignorar os tempos de troca de contexto. Há necessidade de invocação assíncrona se os processos cliente e servidor forem programados com múltiplas threads? página 213

**** Fim da lista ****