LEIC/LETI 2016/17, 1º Teste de Sistemas Distribuídos, 28 de março de 2017

Responda no enunciado, usando apenas o espaço fornecido. Identifique todas as folhas.
Uma resposta errada numa escolha múltipla desconta 1/N do valor da pergunta (sendo N o número de opções).
Duração da prova: 1h30m

Grupo I [7]

Considere o seguinte método em Java para adicionar um produto a um carrinho de compras:

```
public void addToCart(String itemName, double price, int quantity) {
    Item temp = new Item(itemName, price, quantity);
    totalPrice += (price * quantity);
    itemCount += quantity;
    cart[itemCount] = temp;
    if(itemCount==capacity) {
        increaseSize();
    }
}
```

1) A função é auto-explicativa. Pretende usar esta função como base para a definição de uma função idêntica a ser executada num servidor em SUN RPC, para a qual se pretende definir a respetiva especificação em IDL do SUN RPC.
   a) [0,3] Comece por definir a designação da interface remota . Seja completo na especificação para que contenha todos os elementos importantes que permitem a um cliente encontrar o servidor.

   b) [0,5] Defina quais vão ser em Sun-RPC os parâmetros de entrada e de saída da função. Escreva-os na IDL do SUN RPC. Inclua todos os elementos necessários à compreensão da sua resposta.

   c) [0,7] No espaço abaixo programe a totalidade da especificação.

2) Considere a função `addToCart`

a) [0,6] A função é idempotente? Justifique apresentando um exemplo de uma execução, passo por passo.

b) [0,7] Considere que quando o cliente se liga ao servidor (*binding*) escolhe o protocolo TCP. Explique, com base na semântica de execução, que faltas são toleradas e quais as não toleradas.

c) [0,7] Suponha que a ligação se efetuava com UDP e que, numa dada situação, se verificou que foram adicionados ao carrinho de compras 3 produtos quando o cliente só tinha efetuado a compra de um. Consegue explicar esta situação com base no modelo de faltas do Sun-RPC em UDP? Justifique.

d) [0,5] Na situação da alínea anterior (usando UDP) a invocação remota falha no cliente. O que pode saber com certeza?

i) O servidor foi abaixo e não respondeu ao pedido.
ii) Excedeu o numero máximo de tentativas de repetição por parte do RPC do cliente.
iii) A mensagem de invocação nunca conseguiu chegar ao servidor.
iv) Ou se perdeu a mensagem de resposta ou a de envio.

3) Num RPC hipotético (não necessariamente o SUN RPC), considere que os dados da tabela abaixo são utilizados para invocar `addToCart(item, price, qty)`

Valor dos parâmetros	Tag Arquitectura do cliente	Tag	Representação em UTF-8	Representações binárias/ internas ao cliente (idênticas ao formato canónico)
Nikon d750 (v1)	12 (MS_Little_endian)	<string> (t1)	\x4E\x69\x6B\x6F\x6E\x20\x44\x37\x35\x30 (u1)	4e 69 6b 6f 6e 64 37 35 30 (b1)
1925,5 (v2)		<real> (t2)	\x31\x39\x32\x35\x2C\x35 (u2)	0x409e160000000000 (b2)
2 (v3)		<integer> (t3)	\x32 (u3)	0x0002 (b3)

a) Represente a parte da mensagem de invocação que envia os parâmetros da função, com as políticas de tratamento da heterogeneidade identificadas em cada alínea.
(pode usar para simplificar os elementos v1, v2, v3, t1, t2, t3, u1, u2, u3, b1, b2, b3 para não ter de copiar todos os elementos da tabela).

i) [0,5] Implícito, canónico, binário.

ii) [0,5] Explícito, texto em formato canónico.

iii) [0,5] O recetor converte, implícito, binário.

b) Considerando as três alíneas anteriores.

i) [0,5] Qual se aproxima mais do modo de tratamento da heterogeneidade do SUN-RPC? Justifique.

ii) [0,5] E do formato usado pelo XML? Justifique.

c) [0,5] Em que componente dos sistemas de RPC é executada a transformação dos parâmetros que permite colocá-los no formato de transmissão usado no RPC:

i) *Stub* do cliente em conversão canónica.
ii) *Stub* do cliente e do servidor em recetor-converte.
iii) *Run-time* do RPC.
iv) No *binding.*

Grupo II [6]

Considere as duas interfaces remotas seguintes, em Java RMI:

```
public interface InterfaceX extends Remote {
      int metodoA() throws RemoteException;
}

public interface InterfaceY extends Remote {
      void metodoB(InterfaceX a) throws RemoteException;
}
```

Assuma que, num dado sistema distribuído, existem 3 máquinas virtuais – M1, M2, M3 –, ligadas por uma rede local. Na mesma rede existe um *RMI Registry* ativo.

Num dado momento:

- Existe um objeto de tipo InterfaceX instanciado na máquina M1; esse objeto está registado no *RMI Registry* com o nome "//M1/objX".

- Existe um objeto de tipo InterfaceY instanciado na máquina M2; esse objeto está registado no *RMI Registry* com o nome "//M2/objY".

1. [1,8] Programe um método main que se executará na máquina M3.
   O programa deve invocar o método *metodoB* do objeto "//M2/objY", passando como argumento uma referência para o objeto "//M1/objX".
   Na sua resposta: omita a definição do gestor de segurança; não se esqueça de tratar as exceções (basta retornar do programa em caso de exceção).

```
public static void main(String args[]){









}
```

2. No programa pedido na alínea 1, descreva sucintamente:
   a. [1,0] Descreva uma situação que leva a que o programa acima apanhe uma exceção RemoteException.

   b. [1,0] Uma situação que leva a que o programa acima cause o envio de uma mensagem addRef. Indique a quem é enviada e justifique.

3. Entre as classes/interfaces seguintes, indique aquelas que precisam de estar disponíveis na máquina M3 para que o programa da alínea 1 se execute corretamente.
   Assinale apenas "Necessário" ou "Desnecessário"; resposta errada desconta 50% da alínea respetiva.

   a. [0,2] Interface InterfaceX Necessário ☐; Desnecessário ☐

   b. [0,2] Interface InterfaceY Necessário ☐; Desnecessário ☐

   c. [0,2] Classe que implementa o objeto "//M1/objX" Necessário ☐; Desnecessário ☐

   d. [0,2] Classe que implementa o objeto "//M2/objY" Necessário ☐; Desnecessário ☐

   e. [0,2] Classe proxy referente à interface InterfaceX Necessário ☐; Desnecessário ☐

   f. [0,2] Classe proxy referente à interface InterfaceY Necessário ☐; Desnecessário ☐

4. [1,0] Considere agora que a classe que implementa o objeto "//M2/objY" é a seguinte:

```
public class ServantY extends UnicastRemoteObject implements InterfaceY {
      private InterfaceX r;
      void metodoB(InterfaceX a) throws RemoteException {
            r = a;
      }
}
```

Assumindo que o programa descrito na alínea 1 se executou completamente, e tendo em conta a implementação acima, indique quantos *proxies* existem na máquina M2. Justifique.

Grupo III [7]

1) Considere a seguinte implementação do jogo Tic Tac Toe usando as tecnologias de Web Services em Java:

```
1.    @WebService
2.    public class TTTImpl {
3.        public String currentBoard() {
4.              // ...
5.        }
6.        public boolean play(int row, int column, int player) {
7.              // ...
8.        }
9.        public int checkWinner() {
10.             // ...
11.       }
12.   }
```

a) [0,5] Qual a abordagem seguida para a implementação?

A. Test-first
B. Contract-first
C. Implementation-first
D. Network-first

b) [0,8] Descreva uma vantagem e uma desvantagem desta abordagem ao desenvolvimento.

Vantagem:
Desvantagem:

c) [0,8] Com esta abordagem, é depois possível construir um cliente do serviço numa outra linguagem de programação (Python, por exemplo)? Justifique a sua resposta indicando o aspeto concreto que permite/impede a construção do dito cliente.

d) [1,0] Construa uma mensagem SOAP de um pedido referente à chamada `play(1, 2, 0)`. (pode simplificar a sintaxe, desde que a estrutura seja apresentada de forma clara)

```
<s:envelope>




</s:envelope>
```

e) [0,7] Pretende-se adicionar à mensagem anterior um identificador do cliente que faz o pedido. De que forma se pode passar esta informação sem modificar a assinatura do método `play()`?

f) [0,2] Por omissão, qual é o protocolo de transporte utilizado?

g) [0,7] Caso se pretenda utilizar um protocolo de transporte alternativo, onde deve ser feita essa opção?

h) [0,8] Considere agora a implementação da operação que devolve o tabuleiro de jogo.

```
 1.    public String currentBoard() {
 2.        StringBuilder sb = new StringBuilder();
 3.        synchronized (this) {
 4.            sb.append(board[0][0]).append(" | ");
 5.            sb.append(board[0][1]).append(" | ");
 6.            sb.append(board[0][2]).append(" ");
 7.            sb.append("\n---+---+---\n ");
 8.            // ...
 9.        }
10.        return sb.toString();
11.    }
```

Para que servem as linhas 3 e 9 ?
Descreva uma situação errada que poderia ocorrer caso fossem omitidas.

i) [0,5] Que tipo de informação sobre um serviço pode ser guardada no UDDI?

A. Dados sobre a empresa que presta o serviço.
B. Endereço do serviço.
C. Classificação do serviço em área de negócio.
D. Todas as anteriores.
E. A e B

j) [1,0] Identifique duas potenciais vantagens concretas da utilização de UDDI no serviço TTT.

1.

2.