INE5645 – Programação Paralela e Distribuída – Turma B - Prof. Bosco - 12/06/2012
Nome _____BOSCO - GABARITO DA PROVA 2 _____

Indique Verdade (V) ou Falso. Quando for falso corrija a definição ou explicação dada, colocando a frase que exprime a correção conceitual. Preencha as quastões com frases incompletas.

Cada questão rigorosamente correta vale 2.00 pontos.

1. (V) RPC – Remote Procedure Call - serve para desenvolver aplicações distribuídas cliente-servidor com processos locais e remotos.

(F) RMI - Remote Method Invocation – serve para desenvolver aplicações distribuídas cliente-servidor com threads executadas assincronamente.

RMI - Remote Method Invocation – serve para desenvolver aplicações distribuídas cliente-servidor desenvolvidas com objetos distribuídos remotamente.

2. (V) RPC utiliza sockets para a comunicação entre um cliente-servidor.
(F) RMI utiliza sockets entre os módulos de comunicação cliente-servidor com objetos.

RMI utiliza sockets entre os módulos de comunicação cliente-servidor com objetos.
RMI utiliza stubs e skeletons associados a cada dos objetos-servants da lógica da aplicação, os quais implementam transparentemente toda a comunicação entre cliente e servidor.

3. (F) Sockects TCP stream utiliza o protocolo UDP com datagramas.
Sockects TCP stream utiliza o protocolo TCP.

Sockects TCP stream utiliza o protocolo UDP com datagramas.
Sockects TCP stream utiliza o protocolo TCP.

(F) Multicast Socket utiliza múltiplos endereços IP únicos para um grupo de membros participantes no multicast.

Multicast Socket utiliza um único endereço IP para um grupo de membros participantes no multicast.

4. (V) Importância de transparência em RMI - Todas as etapas necessárias para empacotamento e troca de mensagens, mais a tarefa de localizar e contactar um objeto remoto, são ocultadas do programador que faz uma chamada remota.

(F) Em um sistema distribuído com transparência de acesso, o cliente não precisa saber da localização do servidor.

Pelo contrário, o que é certo é que o cliente se utiliza de uma instrução tal como Mensageiro m = (Mensageiro) Naming.lookup("rmi://localhost/MensageiroService"), caso em que os programas cliente e servidor sejam executados na mesma máquina. Mas, se o cliente está numa máquina e o servidor em outra, a instrução acima, no lugar de localhost, terá um IP da máquina do servidor e a porta (exemplo: 150.162.60.159:1099, onde roda o objeto servidor. Logo, o cliente tem que saber onde está o objeto-servidor. O que o cliente não precisa saber é a localização dos objetos-servants que implementam a lógica da aplicação do lado do servidor, ou os objetos que são criados por esses servants.

5. Middleware é uma camada de software que fornece um modelo de programação (apropriado para aplicativos distribuídos, ou seja, para implementação de aplicações com objetos distribuídos na rede, executados em várias máquinas), e que é situado entre camadas de software (no nosso caso, o Java RMI, entre as camadas de Aplicação e de Transporte que usa um protocolo baseado em mensagens entre objetos (locais e remotos), no sentido de fornecer abstrações de mais alto nível (o que é de baixo nível fica oculto pela transparência de localização), como as invocações a métodos remotos, proporcionando transparência de localização nessas invocações.