	Departamento de Informática **Faculdade de Ciências e Tecnologia** **UNIVERSIDADE NOVA DE LISBOA**

Licenciatura em Engenharia Informática
2ª CHAMADA – Sistemas Distribuídos I – 1 de Julho de 2003
2º Semestre, 2002/2003

Leia com atenção cada questão antes de responder. A interpretação do enunciado de cada pergunta é um factor de avaliação do teste. O exame tem 6 questões e 5 páginas.
Não pode utilizar elementos pessoais de consulta. **A duração do exame é 3h00**.
Os alunos devem colocar o nome e o número em todas as páginas.

1) O package java.net disponibiliza um conjunto de classes que permitem comunicar através de troca de mensagens e através de fluxos de dados ("data streams"). A funcionalidade disponibilizada é no essencial equivalente à interface de "sockets" para os protocolos TCP/IP. Das seguintes questões, indique quais as verdadeiras e quais as falsas justificando a sua resposta no caso das falsas.

a) A comunicação por mensagens disponibilizada pelo package java.net é do tipo da de um sistema de comunicação por mensagens persistentes.

Verdadeiro. Falso porque:

b) A comunicação por mensagens disponibilizada pelo package java.net é do tipo da de um sistema de comunicação síncrono por mensagens.

Verdadeiro. Falso porque:

c) A comunicação por fluxos de dados disponibilizada pelo package java.net é do tipo da de um sistema de comunicação por mensagens voláteis.

Verdadeiro. Falso porque:

d) A comunicação por fluxos de dados disponibilizada pelo package java.net é do tipo da de um sistema de comunicação por mensagens assíncronas.

Verdadeiro. Falso porque:

e) A comunicação por mensagens disponibilizada pelo package java.net mascara as falhas arbitrárias do meio de comunicação.

Verdadeiro. Falso porque:

f) A comunicação por mensagens disponibilizada pelo package java.net mascara as falhas temporais do meio de comunicação.

Verdadeiro. Falso porque:

g) A comunicação por fluxos disponibilizada pelo package java.net mascara as falhas de omissão do meio de comunicação.

Verdadeiro. Falso porque:

2) Comente a seguinte afirmação, isto é, diga se está ou não de acordo com a mesma e justifique a sua opinião: "a utilização da replicação de componentes é uma técnica muito simples de implementar para mascarar as falhas das componentes de um sistema distribuído sobretudo quando estas mudam constantemente de estado".

3) Um sistema de replicação de objectos não imutáveis está estruturado internamente segundo a aproximação "Peer-to-peer". Todos os servidores participantes no sistema podem utilizar, para comunicarem entre si, um canal de multicasting que se admite fiável por hipótese. Sempre que um objecto é modificado, o seu novo estado é transmitido aos outros participantes. Em que condições este sistema pode usar uma ordem "FIFO" para transmitir as novas versões dos objectos e em que condições tem de utilizar uma ordem "TOTAL" para transmitir as modificações dos objectos.

4) No sistema de Java/RMI, é frequente os objectos remotos registarem um seu "proxy" no servidor "Registry", que é um serviço de "naming" de objectos remotos que permite estabelecer ligações num sistema distribuído. Das questões a seguir indique quais as verdadeiras e quais as falsas justificando a sua resposta no caso das falsas. **Nota:** uma resposta errada decrementa a cotação global desta pergunta; em qualquer hipótese a cotação será sempre maior ou igual a zero.

a) Este mecanismo é imprescindível para que um objecto remoto possa ser invocado pois tal é a única forma de o seu "proxy" ser acessível a outra máquina virtual Java.

Verdadeiro. Falso porque:

b) Os dados registados no "proxy" de um objecto remoto permitem localizá-lo e invocá-lo e resumem-se a um endereço IP, uma porta e um identificador único não reutilizado.

Verdadeiro. Falso porque:

c) Quando um servidor de um objecto remoto tem um "crash" e vai abaixo, se conseguir reinicializar-se de novo muito rapidamente, e não houver nenhuma invocação em curso, o "crash" do servidor do objecto é completamente transparente pois o proxy continua disponível no servidor "Registry".

Verdadeiro. Falso porque:

d) Quando um servidor de um objecto remoto A tem um "crash" e vai abaixo, o servidor "Registry" local detecta logo tal facto e anula imediatamente o registo do "proxy" do objecto remoto A.

Verdadeiro. Falso porque:

e) A partir do momento em que o servidor "Registry" local ao site de um objecto remoto tem um "crash", o objecto deixa de poder ser invocado.

Verdadeiro. Falso porque:

5) Considere um servidor de ficheiros remotos "stateless" com uma interface semelhante à do sistema NFS baseada em SUN/RPC. As invocações remotas são feitas sobre UDP utilizando um protocolo de invocação com a semântica "1 ou mais vezes". O servidor conhece a lista dos utilizadores registados e uma palavra chave ("password") dos mesmos. O servidor de ficheiros tem uma chave criptográfica assimétrica pública (a chave é uma chave RSA) que é bem conhecida dos clientes. Descreva detalhadamente uma solução de autenticação e segurança que permita a invocação dos serviços do servidor de ficheiros e que garanta segurança contra "masquerading", olhares indiscretos, "replaying", etc. A invocação deverá poder continuar a ser sobre UDP e o servidor deverá poder continuar a ser, no essencial, "stateless" do ponto de vista do protocolo de invocação das operações sobre os ficheiros. Argumente sobre a correcção da sua proposta.

6) Embora o protocolo NFS não especifique na norma (RFC) um mecanismo particular de gestão de *caching*, muitas implementações implementam mecanismos de *caching* geridos do lado do cliente, no pressuposto da ausência de estado sobre o cliente por parte do servidor. Na sua forma usual, a gestão de *caching* das implementações baseia-se num mecanismo de etiquetas temporais (*timestamps*) e interrogações do servidor pelos clientes. Admita, por hipótese, que essa gestão é como se resume a seguir.

Sempre que um cliente obtém dados sobre um ficheiro a partir do servidor, associa-lhes uma etiqueta temporal, fornecida pelo servidor, com o momento da última escrita no ficheiro. Quando se lêem dados de um ficheiro *cached* no cliente, o cliente apenas invoca um procedimento no servidor para verificar se a sua *cache* ainda se mantém válida, isto é, que verifica o momento da última escrita no ficheiro. Caso a *cache* ainda seja válida, isto é, o ficheiro não foi no entretanto modificado, a leitura do ficheiro pelo cliente é resolvida na *cache*. Se isso não se verificar, a *cache* é invalidada e faz-se a leitura invocando o servidor remotamente. Caso o cliente escreva no ficheiro, a escrita é repercutida na cache e é imediatamente enviada ao servidor.

a) Diga porque é que um esquema tão simples como o anterior não é nada interessante do ponto de vista de desempenho (*performance*).

b) Diga quais as vantagens, se existir alguma, que este esquema apresenta.

c) Indique uma ligeira modificação que poderia introduzir no protocolo NFS para melhorar a gestão da *cache* do cliente inspirada na directiva "if-modified-since" do protocolo http.

d) Que optimização(ões) são incluídas nas implementações de NFS na gestão da cache para melhorar o desempenho e em que pressupostos se baseia(m) essa optimização(ões) ?

e) Que tipo de dados sobre os ficheiros são guardados na *cache* do cliente, conjuntamente com o conteúdo do ficheiro, tendo em vista o controlo de validade dos dados na *cache* ?