Perguntas - SD

1. Explique o que entende por middleware.

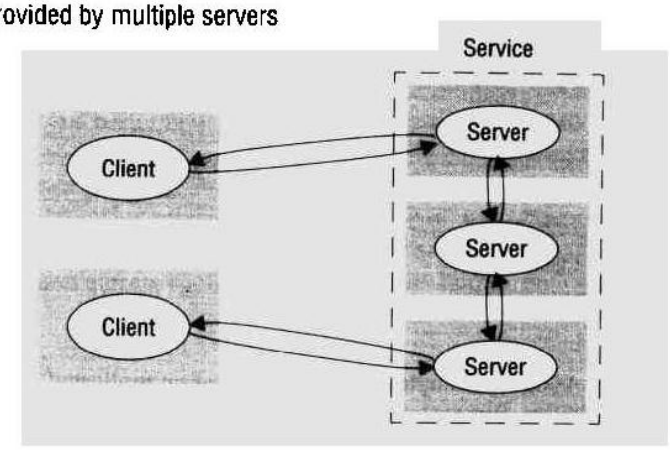
Middleware no campo de computação distribuída, é um programa de computador que faz a mediação entre outros softwares. É utilizado para mover informações entre programas ocultando do programador diferenças de protocolos de comunicação, plataformas e dependências do sistema operacional. Seu objectivo é mascarar a heterogeneidade e fornecer um modelo de programação mais produtivo para os programadores de aplicativos. É composto por um conjunto de processos ou objectos em um grupo de computadores, que interagem entre si de forma a implementar comunicação e oferecer suporte para compartilhamento de recursos a aplicativos distribuídos.

2. Descreva esquematicamente, introduzindo os comentários apropriados, um modelo cliente-servidor, com replicação de dados (tipo 3), em q eu o serviço prestado é alterar uma base de dado interna organizada em registos base. Assuma que o servidor é implementado em 3 sistemas computacionais diferentes.

O modelo cliente-servidor é um dos modelos de sistemas distribuídos, constituído por clientes e servidores.

O modelo 3, é o modelo com replicação de dados, neste existem vários servidores, garantindo que os dados em todos eles são iguais, ou seja, tem de se fazer lock dos dados.
Neste caso, o nosso modelo inclui 3 sistemas computacionais diferentes:

A service provided by multiple servers



Neste sistema temos clientes que estão ligados a um sistema computacional, existe uma região partilhada onde estão armazenados os dados. Os sistemas computacionais comunicam através de uma rede, sempre que ocorre alguma modificação na região partilhada tem de se fazer lock dos dados para manter a consistência da informação.

3. Explique os princípios gerais do modelo de passagem de mensagens baseado no protocolo UDP em java. Quais são as suas limitações principais?

O modelo de passagem de mensagens, baseado no protocolo UDP, funciona do seguinte modo:

➔ O cliente requer o serviço, enviando ao servidor uma mensagem identificando o serviço que pretende e os dados de input.
➔ O servidor está sempre à espera de mensagens processando-as, e enviando ao cliente uma mensagem com os resultados

O protocolo UDP é do tipo connectionless, não há garantia de entrega nem de ordenamento das mensagens. A transmissão é unidireccional, gestão das mensagens é feita a nível do utilizador, não há avisos em caso de falhas.
Em Java, o UDP é suportado através das seguintes classes:

- Datagram Packet class -> um objecto deste tipo representa os dados para a transmissão UDP, contém a informação de endereço IP e portos.
- Datagram Socket class -> proporciona acesso para o socket UDP, o qual permite que os pacotes sejam enviados e recebidos.

As principais limitações deste modelo são a falta de garantia de entrega dos pacotes, a inexistência de sequencia de pacotes (pacotes não numerados), inexistência de controlo de fluxo.

4. Que papel é desempenhado pelo servidor http no modelo RMI de java.

O RMI do java usa HTTP para pedir a invocação de um objecto remote quando este está protegido por uma firewall.
O RMI normalmente tenta abrir sockets directamente para os hosts da internet, em muitas intranets, como estas tem firewalls que não permitem que esta ligação seja efectuada directamente, para isso são utilizados dois métodos alternativos baseados em mecanismos http que permitem que um cliente atrás de uma firewall consiga invocar um método num objecto remoto fora da firewall.

5. Explique o que é um sistema distribuído e indique dois aspectos que têm que ser ponderados na sua concepção.

Um sistema distribuído é um sistema de operação cujos componentes, que estão localizados nos diferentes elementos de um sistema computacional paralelo, coordenam e comunicam normalmente as suas acções por passagem de mensagens.

É um conjunto independente de computadores que pela sua coerência aparenta ser um único sistema para os seus utilizadores.

A motivação principal que leva à construção e à utilização destes sistemas, é a partilha de recursos.

A concepção destes sistemas distribuídos, e das aplicações executadas sobre eles, envolve algumas questões:

- Grau de abertura que o sistema apresenta
- Segurança dos fluxos de informação trocados e da informação armazenada
- Potencial apresentado de modo a aumentar a escala
- Tratamento de falhas
- Sincronização e a exclusão mútua no acesso a regiões críticas
- Um grau de transparência
- Heterogeneidade dos componentes que formam o sistema computacional paralelo.

6. Caracterize o modelo de programação providenciado pelo protocolo TPC, quer do lado do servidor, quer do lado do cliente.

O modelo de programação consiste em:

- ➔ Existe o pedido de abertura do canal (cliente)
- ➔ Existe a confirmação do pedido (servidor)
- ➔ Existe o biding de ambos os lados
- ➔ Envio de mensagem (cliente)
- ➔ Confirmação de recepção de mensagem (servidor)
- ➔ Fecho do canal

7. Descreva o modelo cliente-servidor e indique qual é o papel aí desempenhado por um servidor proxies.

O modelo cliente-servidor é constituído por um conjunto de PCs, e os dados e aplicações que se encontram armazenadas num servidor.

A comunicação neste modelo é feita num modelo request-reply, ou seja o cliente envia uma mensagem ao servidor indicando qual o serviço que pretende e os dados de input necessários.

Por sua vez, o servidor executa a tarefa e responde ao cliente, enviando os resultados requeridos.

O proxy é um intermediário na interacção com o servidor. Um servidor de proxies, por sua vez tem a função de analisar o pedido do cliente e, dependendo das permissões aí existentes, o servidor irá executar o pedido se este tiver permissão.

Isto é, o cliente pede um serviço, o proxy faz o marshalling do pedido do cliente e o unmarshalling das respostas do servidor. Funciona basicamente como um tradutor.

8. Em que consiste o método de invocação remota? Será que dentro deste modelo é possível a migração de código? Como?

O método de invocação remota (RMI) permite que haja interacção entre programas que estão a ser executados em diferentes nós de um sistema distribuído, através da invocação de procedimentos remotos (RPC).
Neste modelo é possível a migração de código através de operações de marshalling e unmarshalling. O objecto remoto é instanciado e registado. Utilizando o marshalling o código é copiado para o servidor e é feito um unmarshalling de modo a que o código seja executado.

9. Um atributo importante de um sistema distribuído é a sua transparência. Explique em que consiste e, neste contexto, distinga transparência de acesso de transparência de localização.

A transparência é um dos objectivos principais dos sistemas distribuídos, consiste em esconder o facto de que os processos e recursos estão fisicamente distribuídos por múltiplos computadores. Um sistema distribuído que é capaz de se apresentar como um único computado é chamado transparente.

Formas de transparência:

- Acesso: mesmo tipo de operações no acesso local e remoto.

- Posição: mascara a localização precisa de um dado recurso (remoto)

- Rede: acesso + posição

- Replicação: réplicas dos recursos sem que seja notório.

- Concorrência: quando o acesso aos recursos partilhados é feito sem interface

- Falhas: quando as malhas podem ser mascaradas e as tarefas terminadas

- Movimento: quando os utilizadores e recursos podem ser movidos dentro do sistema

- Desempenho: Quando pode ocorrer uma reconfiguração de sistema.

10. Qual é o papel desempenhado por uma arquitectura distribuída que disponibiliza um conjunto de serviços na internet? Faça um diagrama que ilustre a sua localização.

Uma arquitectura distribuída tem como função, fazer a distribuição dos serviços que disponibiliza por várias unidades de processamento interligando entre si cada uma destas unidades + qualquer coisa

11. Quais são as diferenças principais entre os protocolos TCP e UDP?

TCP (connection oriented):
- Controlo de mensagens (detecção de mensagens repetidas e o ordenamento de pacotes)
- Gestão do fluxo (bloqueio de remetente em caso de congestionamento)
- Tamanho de mensagens variável
- Comunicação Bidireccional
- Comunicação ponto-a-ponto (um servidor é um cliente)

UDP (connectionless):
- Não há garantia de entrega nem de ordenamento das mensagens
- Unidireccional
- Gestão das mensagens feita ao nível do utilizador
- Não existe notificação em caso de falha

No TCP só depois de estabelecer (aberto) o canal de comunicação começa a transferir.
No UDP o canal é estabelecido no envio da 1ª mensagem.
TCP é – eficiente e + seguro
UDP é + eficiente e - seguro

12. Uma maneira de se efectuar a localização de objectos dentro do modelo de invocação remota de métodos (RMI) é através do rmiregistry. Explique em que consiste e descreva as operações que têm que ser realizadas pelo servidor e pelo cliente para que tal seja possível.

O rmiregistry permite guardar e obter informação sobre objectos remotos existentes numa máquina. Para se poderem distinguir os objectos, é usado um URL com o seguinte formato: rmi://host:port/name. Cada objecto remoto

regista-se no RMI Registry da máquina onde se encontra. Um objecto que deseje invocar os métodos de um objecto remoto pode entar em contacto com o RMI Registry da máquina onde se encontra o objecto remoto indicando o nome do objecto pretendido e obter a respectiva referência.

13. A disponibilização generalizada de serviços e a aparência demonstrada pelos sistemas computacionais paralelos para a execução cooperativa de aplicações levantam a questão de se saber quão seguros são os fluxos de informação trocados e quão segura é a informação armazenada nos recursos associados. Indique os diferentes níveis em que se pode colocar a segurança de informação e descreva medidas que podem ser usadas para reduzir os riscos.

A segurança da informação pode ser vista a três níveis.

➔ Confidencialidade – protecção contra a sua divulgação a entidades não autorizadas, refere-se a propriedade do computador, a informação está "fechada" só para quem está autorizado.

➔ Integridade – protecção contra a alteração ou corrupção. Os dados só podem ser alterados por utilizadores autorizados, alterações impróprias devem ser detectadas pelo sistema distribuído e corrigidas.

➔ Disponibilidade – protecção contra interferências que perturbem os mecanismos de acesso.

Mecanismos de segurança:

1- Encriptação: transforma os dados numa "coisa" que o atacante não consegue perceber.

2- Autenticação: é usado para verificar a identidade do utilizador, servidor, tipicamente implementado usando pass.

3- Autorização: depois do cliente ser autenticado tem de se verificar se pode efectuar certas operações.

4- Auditoria: são logados os dados sobre quem, quando e que frequência foram armazenados dados. É muito importante para analisar em caso de problemas.

Algumas medidas para reduzir os riscos:

➔ Introdução de firewalls – que cria uma barreira, controlando o fluxo que entra e sai da rede.

➔ Encriptação das mensagens – mascara o conteúdo dos fluxos.

➔ Assinaturas electrónicas – para certificar o autor das mensagens contudo há situações que não se podem controlar (ataques DoS, segurança do código móvel)

14. Uma forma de implementação do modelo cliente-servidor é efectuar a replicação do servidor sempre que há uma solicitação de um novo cliente. Mostre que vantagens e inconvenientes é que esta arquitectura apresenta.

O modelo cliente-servidor é um dos modelos de sistemas distribuídos, constituído por clientes e servidores.

Modelo 2, Replicação do servidor existe um servidor que atende os clientes à medida que eles vão chegando, criando um Agente Prestador de Serviços para cada um deles, ou seja, não ficar a aguardar que o APS termine para atender o próximo pedido.

Vantagens da replicação do servidor: em vez de um sistema multiprocessador temos um sistema paralelo.
Desvantagens: Garantir a exclusão mutua no acesso a região partilhada, efectuar sincronização.
Neste modelo o APS "nasce" quando invocado pelo servidor para uma determinada tarefa, e uma vez terminada, devolve o resultado ao cliente e "morre".
Existe portanto, uma paralelização das aplicações disponibilizadas pelo servidor, permitindo uma maior eficiência e rapidez na obtenção dos resultados desejados pelo cliente.
1-> O Cliente faz request de um pedido
2-> O servidor cria APS para atender o cliente
3-> O APS devolve o resultado e morre

15. A serialização de objectos em Java constitui uma abordagem possível ao problema da comunicação de dados em ambientes distribuídos. Explique neste contexto em que consiste o marshalling e o unmarshalling de informação.

Na serialização de objectos em JAVA um objecto é uma instância de uma classe que possui um estado.

O marshalling e o unmarshalling são usados de modo a "serializar" um objecto que é transformado numa sequência de bytes e posteriormente reconstruilo.

Para fazer estas operações são usados os mecanismos de marshalling e unmarshalling.

Estas operações têm de ser realizadas para que seja possível transferir a informação através da rede. Para isso o objecto tem de ser convertido num conjunto de bytes (marshalling) e depois recontruido no destino (unmarshalling).

16. O modelo de invocação remota de métodos (RMI) constitui um modelo de comunicação com uma abstracção mais elevada do que o modelo puro de passagem de mensagens implementado com sockets. Mostre porquê.

O RMI permite uma maior abstracção, devido ao facto de permitir a invocação de métodos que não estão presentes no mesmo sistema computacional, e permite isto de uma forma igual à que se usa para invocar um método local. Para o mesmo ser feito usando sockets, era necessário estabelecer uma ligação com o servidor, enviar uma mensagem com uma indicação de qual o método a executar e os seus argumentos de entrada, e depois era necessário esperar que o servidor devolve-se uma mensagem com os resultados da operação. Ora este método torna-se muito mais complexo e muito pouco abstracto. O RMI esconde todos estes pormenores dando a sensação ao programador que os métodos são locais simplificando o código e permitindo uma melhor legibilidade.

17. Explique como é que o RMI (Remote Method of Invocation) em Java lida com a implementação a transparência de acesso de transparência de localização. Que limitações apresenta?

O RMI para criar uma camada de abstracção, estende o modelo de programação baseado em objectos por forma a suportar a invocação de métodos de objectos que pertencem a outros processos. Isto é feito através da criação de uma interface que estende a interface Remote e que mantém uma assinatura dos métodos remotos a executar. O programa que irá usar a interface, contacta o rmiregistry a pedir uma referencia do objecto remoto ao qual irá associar à interface, permitindo assim uma execução dos métodos como se fossem locais. Apesar de o RMI criar uma abstracção de alto nível na invocação dos métodos de objectos remotos, não é possível criar uma transparência total devido ao facto de ser necessário obter uma referência para o objecto remoto. Para obter uma referência é obrigatório que o rmiregistry esteja a decorrer no sistema computacional aonde as classes com os métodos respectivos estão registadas.

18. No contexto do RMI em Java, explique o que entende por objectos [remotos] estáticos e dinâmicos. Descreva funcionalmente como é que eles podem ser implementados e dê um exemplo de cada tipo da sua aplicação.

Estático (Cave)-> tempo de vida igual ao da thread que o instancia, implementada uma região de dados partilhada

Dinâmico (party) -> o tempo de vida inferior ao da thread, o que lhe permite instanciar múltiplos objectos, implementa um canal de comunicação para permitir a comunicação entre entidades existentes.

19. No âmbito da comunicação entre pares, descreva detalhadamente um algoritmo que permite de uma forma distribuída e dinâmica a u dado processo assumir-se como líder. Indique claramente quais são os pressupostos que estão subentendidos.

Admitindo que a rede está em perfeitas condições e não existem falhas, logo de inicio cada um dos nós gera para si um número de identificação, assume-se a

ele mesmo como sendo o líder, marca a mensagem como sendo sua e passa-a ao nó seguinte, no sentido do relógio.
O nó seguinte ao receber a mensagem, verifica o identificador do líder na mensagem e verifica se o seu identificador é maior que o identificador de líder e caso seja, substitui o identificador de líder na mensagem pelo seu, caso contrário, deixa ficar como está, e terminando a avaliação passa-a ao nó seguinte.
As mensagens continuam passando por todos os nós, até ao ponto em que cada uma das mensagens de eleição chega ao nó inicial. Neste momento, cada um dos nós ao verificar que a mensagem de eleição foi iniciada por si, apercebe-se que esta já percorreu por todos os nós e compara o valor de identificação de líder com o seu número de identificação e caso seja igual, identifica-se como sendo o líder, caso contrário, identifica-se como sendo um coordenador.

20. Considere uma aplicação distribuída implementada segundo a arquitectura do modelo cliente-servidor. Indique as características principais que este modelo apresenta. Descreva detalhadamente a sua variante em há aquilo q se designa de replicação de servidor.

Este modelo caracteriza-se por ter um cliente e um servidor, fornecedor de um recurso partilhado, em que ambos estão ligados por uma via de comunicação. O funcionamento deste modelo é muito simples, o cliente estabelece uma ligação com o servidor e envia uma mensagem que internamente, indica o seu pedido. O servidor que até aqui tem estado à espera de receber uma mensagem, ao recebe-la, avalia o seu conteúdo e processa-o, devolvendo uma mensagem que contem o recurso pedido pelo cliente. O cliente recebe a mensagem e caso não deseje fazer mais nenhum pedido, termina a via de comunicação.
A variante de replicação deste serviço, não é muito diferente da sua versão normal que foi acabada de descrever, estando a diferença no número de servidores disponíveis no sistema distribuído que fornece o recurso partilhado e na forma como este mesmo recurso é distribuído pelos servidores. Neste método de replicação, a informação é replicada por cada um dos servidores, aumentando a performance e disponibilidade, e aumentando a redundância do sistema e tolerância a falhas.

21. Que diferenças fundamentais existem entre os protocolos TCP e UDP na troca de mensagens entre programas cooperantes instalados em sistemas computacionais distintos?

@11
Podemos acrescentar que usando o protocolo UDP no envio de mensagens, o programa não bloqueia na transmissão de um objecto, ao passo que ao usar o protocolo TCP, o programa é bloqueado até ser garantido que o objecto a ser transmitido é efectivamente escrito no buffer do sistema de destino, quer seja o cliente quer seja o servidor.

22. Em java é relativamente simples estabelecer a migração de código entre sistemas computacionais distintos a partir do RMI. Descreva um modelo que permite implementa-lo.

23. Descreva os diferentes níveis em que a segurança de informação se coloca num sistema distribuído. Indique que medidas podem ser tomada para reduzir os riscos.

@13

24. Como se resolver em Java o problema da transparência de posição no acesso a objectos remotos?

Transparência de posição - quando se realiza o acesso aos recursos sem necessidade de conhecimento da sua localização precisa

Um dos aspectos importantes para a transparência de posição é o nome do arquivo, isto é, os arquivos tem um nome lógico, que não está directamente ligado à localização. Um exemplo disto, é uma URL, que não indica de forma alguma a localização do servidor onde a página se encontra, nem se esta sempre esteve na mesma posição ou se foi movida recentemente. Sistemas distribuídos em que os recursos podem ser movidos sem afectar como podem ser acedidos, fornecem transparência de migração. Quando os recursos podem ser acedidos

ao mesmo tempo que estão a ser movidos, existe transparência de reposição. Um exemplo disso é quando utilizadores podem usar o wireless do seu portátil em movimento, sem ser desconectado temporariamente ao mudar de localização.

Utilizando uma middleware, mais propriamente o RMI (Remote Method Invocation). O RMI Permite criar uma layer de software entre o nível da aplicação e o nível das comunicações e módulos de referência remotos. Internamente o RMI, funciona em parte como uma proxy em que invés de executar o pedido de uma invocação, reenvia o pedido encapsulado numa mensagem para o objecto remoto, escondendo assim os detalhes das referências dos objectos e ocultando todas as acções intermédias

25. Considere uma dada aplicação distribuída em que estivesse contemplada a implementação de um modelo cliente servidor. Que tipo de modelo escolheria, se um factor importante a ter em conta fosse a tolerância a falhas do lado do servidor? Justifique a sua resposta.

…

Escolheria o tipo 3, replicação de dados, porque é neste modelo que o controlo de falhas é feito do lado de servidor.

26. Explique o conceito de middleware. Mostre em que termos o ambiente providenciado por Java para implementar o paradigma de passagem de mensagens cumpre especificações básicas associadas a este conceito.

@1
O java usa primitivas que enviam as mensagens, a mensagem é enviada e é função da primitiva trata de todo o processo e comunicação, tradução e envio de mensagens. O mesmo para a recepção, ficando assim marcada todo o processo sobre as primitivas elementares.
- Primitiva send especifica o destinatário e inclui a mensagem a enviar.
- Primitiva received especifica o remetente e fornece um buffer para armazenar a mensagem recebida.
…

27. Se na construção de uma dada aplicação distribuída em que estivesse contemplada a implementação de um modelo cliente-servidor, tivesse de escolher entre um modelo do tipo 2 ou de tipo 3, explique que razões teria em conta na sua decisão.

Tipo 3 (replicação de servidor) – Cada servidor de processamento fornece os seus recursos de forma independente e não precisam de comunicar entre si para sincronização de dados, sistemas não precisam de ser tão potentes, não precisam de tanta memória mas não fornecem redundância nos dados nem tolerância a falhas o que significa que no momento em que um servidor falha, os recursos desse mesmo, deixam de estar disponíveis aos clientes e pode inclusive haver corrupção dos dados, não sendo a sua recuperação possível.
Tipo 2 (replicação de dados) – Os servidores com replicação de dados, mantêm todos, uma cópia igual entre si dos dados, fornecendo todos os mesmos recursos. É necessário que todos os servidores tenham as mesmas características em termos de processamento e de memória e é necessário estarem ligados entre si para sincronização de dados, o que provoca que uma alteração dos dados num servidor, terá que ser comunicada a todos os outros, o que provoca uma necessidade de transmitir mais dados para a rede. Este tipo de modelo suporta redundância nos dados e tolerância a falhas, ou seja, é necessário falharem todos os servidores, para que o sistema distribuído deixe de poder fornecer os recursos aos seus clientes e para deixar de ser possível

28. Considere uma aplicação distribuída em que se usa um modelo de comunicação entre pares que implementa o paradigma de passagem de mensagens. Como é que se estabelecem neste contexto as regiões de dados partilhados?

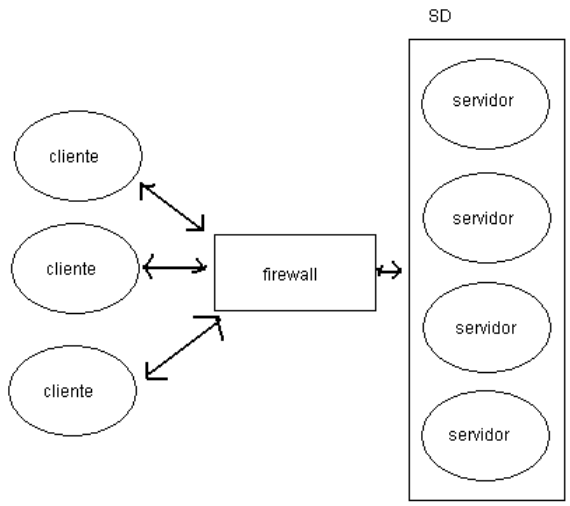
29. Que papel desempenha o firewall no acesso a uma rede local

Uma firewall quando aplicada no acesso a uma rede local, pode ser configurada para filtrar pacotes de informação provenientes de clientes, que não estão autorizados a ter contacto com elementos no interior da rede ou que queiram

ter acesso a serviços ou recursos para os quais não têm autorização. Sendo assim podemos garantir que a firewall desempenha um papel de garantir que os recursos da rede sejam protegidos contra ataques de elementos exteriores à rede e garantir que todos os recursos da rede se mantêm disponíveis aos seus clientes.

30. Qual é o papel desempenhado por uma firewall numa arquitectura distribuída que disponibiliza uma conjunto de serviços na Internet? Faça um diagrama que ilustre a sua localização.

@ 29



31. Distinga modelo cliente-servidor de modelo de comunicação entre pares (peer-to-peer).

No modelo de comunicação entre pares, todos os elementos da rede têm capacidades e responsabilidades equivalentes interagindo de forma cooperativa e sem distinção entre clientes e servidores. Os padrões de interacção podem variar em função das circunstâncias ou da tarefa.
No modelo cliente servidor, existe um cliente que contacta um servidor por forma a ter acesso a um recurso. Uma ligação é criada, é efectuado um pedido ao servidor, no qual este mesmo, fornece uma resposta adequada. Neste modelo, todo o processamento é feito do lado do servidor, ficando o cliente à espera de uma resposta proveniente desse mesmo.
No modelo peer-to-peer, todos os envolventes interagem de forma cooperativa como peers, não existindo distinção entre clientes e servidores. Neste modelo, o código nos peers, mantem a consistência dos recursos ao nível aplicacional e sincroniza as acções do mesmo nível quando for necessário. Podemos ver este modelo como uma área de trabalho distribuída em que todos os envolventes participam no processamento de dados que estão distribuídos entre si.

32. Porque é que em Java e relativamente simples estabelecer a migração de código entre sistemas computacionais distintos? Descreva um modelo que permite implementa-lo.

33. Que diferenças existem em Java na construção de objectos remotos e de objectos locais?

Objectos locais são instanciados localmente e ficam localizados na memória da máquina virtual do java, os objectos remotos são instanciados em máquinas diferentes, e a que não se tem acesso à sua referência directa… dizer mais…

34. A autenticação dos intervenientes numa transacção é um componente importante na implementação de uma política de segurança de informação. Mostre como é que o problema pode ser abordado numa aplicação distribuída que implementa um modelo cliente-servidor.

A autenticação é um dos mecanismos de segurança que permite saber que é um dado cliente. Para isto existe uma interface bem definida do lado do cliente.
No caso dos clientes, a permissão tem de ser pedida antes de que seja efectuada alguma tarefa.
Depois de autenticado tem de ser verificado o nível de autorização que o cliente possui para saber as tarefas que pode realizar.

35. Descreva esquemática e funcionalmente as operações principais que têm que ser estabelecidas, que no lado do servidor, quer no lado do cliente, para construção de uma aplicação distribuída que implemente um modelo cliente-servidor de tipo 2 (replicação de servidor), usando o paradigma de passagem de mensagens.

36. No contexto do RMI em Java, explique como se processa a migração de código entre sistemas computacionais distintos. Refira claramente qual é o papel desempenhado pelos diferentes componentes envolvidos.

37. No âmbito da comunicação entre pares, descreva detalhadamente um algoritmo que permita de uma forma distribuída e dinâmica a um dado processo assumir-se como líder. Assuma que: i) não há perda de mensagens trocadas entre os diferentes processos e o tempo de transmissão é limitado superiormente; ii) o tempo de execução do algoritmo é muito pequeno face ao tempo de vida de cada processo, de modo que se pode presumir que durante a sua execução não ocorre a entrada ao serviço de um novo processo, nem a sua desactivação.

@19

38. Explique o que entende por transparência e justifique a sua importância no âmbito dos sistemas distribuídos.

O **grau de transparência de** um sistema computacional é uma característica que exprime o maior ou menor sucesso que foi conseguido a mascarar a complexidade subjacente. A sua funcionalidade é descrita de uma forma integrada, conceptualmente simples, em vez de resultar da interacção de um conjunto de componentes independentes.

O objectivo da transparência é, por isso, esconder os recursos que não são directamente relevantes para a tarefa em curso, tornando-os anónimos ao utilizador e/ou ao programador de aplicações.

39. Descreva esquemática e funcionalmente as operações principais que têm que ser estabelecidas, que no lado do servidor, quer no lado do cliente, para construção de uma aplicação distribuída que implemente um

modelo cliente-servidor de tipo 2 (replicação de servidor), usando o paradigma das variáveis partilhadas sobre objectos remotos.

@14
Paradigma das variáveis partilhadas sobre objectos remotos ?

40. Distinga claramente o protocolo TCP do UDP enquanto modelo de comunicação. Qual é o mais adequado para a comunicação entre pares? Porque?

TCP (connection oriented):
- Controlo de mensagens (detecção de mensagens repetidas e o ordenamento de pacotes)
- Gestão do fluxo (bloqueio de remetente em caso de congestionamento)
- Tamanho de mensagens variável
- Comunicação Bidireccional
- Comunicação ponto-a-ponto (um servidor é um cliente)

UDP (connectionless):
- Não há garantia de entrega nem de ordenamento das mensagens
- Unidireccional
- Gestão das mensagens feita ao nível do utilizador
- Não existe notificação em caso de falha

No TCP só depois de estabelecer (aberto) o canal de comunicação começa a transferir.
No UDP o canal é estabelecido no envio da 1ª mensagem.
TCP é – eficiente e + seguro
UDP é + eficiente e - seguro

Para comunicação entre pares, o protocolo TCP é o mais adequado, pois apesar de menos eficiente, é mais seguro e garante uma comunicação mais fiável.

41. No âmbito da comunicação entre pares, descreva detalhadamente um algoritmo que permita de uma forma distribuída e dinâmica fazer-se a sincronização dos relógios locais. Que tipo de ordenação de acontecimentos é que resulta da sua aplicação. Mostre como é possível

garantir-se a consistência de dados entre diferentes cópias localizadas em locais geograficamente separados quando ele é usado.