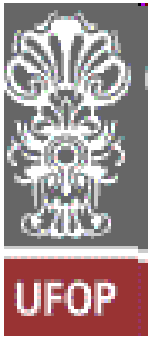


BCC 362 – Sistemas Distribuídos

Joubert de Castro Lima – joubertlima@gmail.com
Professor Adjunto – DECOM

UFOP

Figuras e textos retirados do livro:

Sistemas Distribuídos do Tanenbaum

Security

Middlewares: RMI e MPI

Como medir tempo ??

** Definitions

Security in computer systems is strongly related to the notion of *dependability* (trust).

A *dependable* computer is one that we justifiably trust to deliver its services.

Complementary a security system consider the *confidentiality*, information discussed among the authorized parties, and the *integrity*, alterations are performed only in authorized way.

** Security Mechanisms

Encryption transforms data into something an attacker cannot understand.

Authentication is used to verify the identity of a user, client, server and so on.

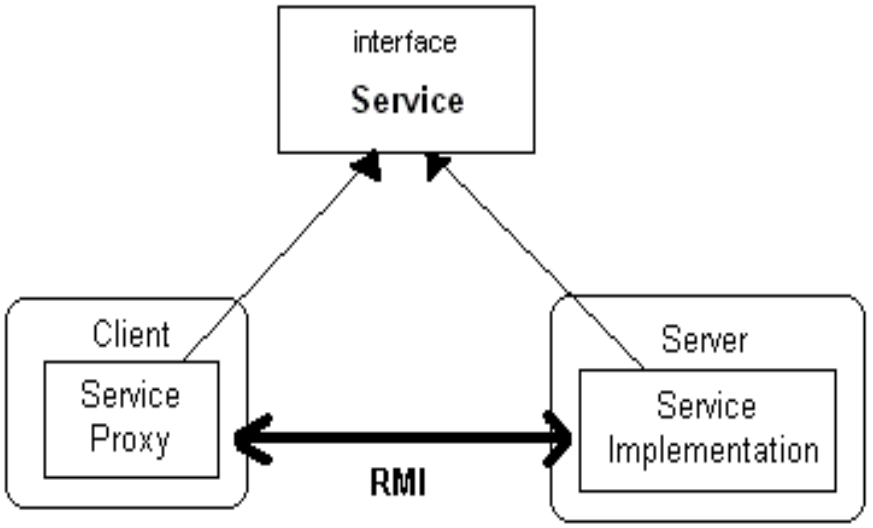
Authorization, after the authentication, it is necessary to check if that client is authorized to perform the action requested.

Auditing are used to trace which clients access what, which way.

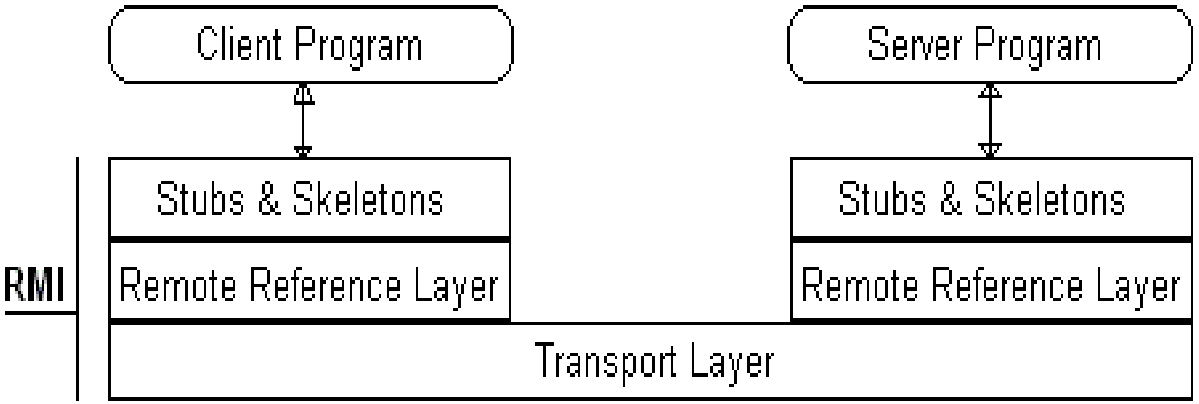
MIDDLEWARES

Objetos Distribuídos e Invocação Remota

Estudo de caso Java RMI



O programa cliente faz chamadas de métodos pelo objeto Proxy, o RMI envia a requisição para a JVM remota e redireciona para a implementação. Qualquer valor retornado pela implementação é devolvido ao Proxy e então ao programa cliente.



A camada Stub e Skeleton está abaixo dos olhos do desenvolvedor. Esta camada intercepta as chamadas de métodos feitas pelo cliente para que a variável de referência da interface redirecione essas chamadas para o serviço RMI remoto.

A próxima camada é a Remote Reference Layer. Esta camada sabe como interpretar e gerenciar referências feitas dos clientes para os objetos do serviço remoto.

A conexão do cliente ao servidor é Unicast (uma-para-um). A camada de transporte é baseada nas conexões TCP/IP entre as maquinas em uma rede.

Como um cliente acha o serviço remoto RMI?

Os clientes acham os serviços remotos usando o serviço de nomeação ou diretório (naming or directory). Isso parece um pouco redundante, mas o serviço de nomeação ou diretório roda como um endereço bem formado (host:port).

Como implementar um programa usando RMI ?

http://www.guj.com.br/article.show.logic?id=37

Example 2.33 Starvation-free producer-consumer code

```
...
typedef struct {
```



```
  char data[MAXSIZE];
  int datasize;
} Buffer;
Buffer *buffer;
MPI_Request *req;
MPI_Status status;
...

MPI_Comm_rank(comm, &rank);
MPI_Comm_size(comm, &size);
if(rank != size-1)  {  /* producer code  */
  buffer = (Buffer *)malloc(sizeof(Buffer));
  while(1) {  /* main loop */
    produce(buffer->data, &buffer->datasize);
    MPI_Send(buffer->data, buffer->datasize, MPI_CHAR,
             size-1, tag, comm);
  }
}
```

MPI

```
else { /* rank == size-1; consumer code */
  buffer = (Buffer *)malloc(sizeof(Buffer)*(size-1));
  req = (MPI_Request *)malloc(sizeof(MPI_Request)*(size-1));
  for (i=0; i<size-1; i++)
    req[i] = MPI_REQUEST_NULL;
  while (1) { /* main loop */
    MPI_Waitany(size-1, req, &i, &status);
    if (i == MPI_UNDEFINED) { /* no pending receive left */
      for(j=0;  j< size-1; j++)
        MPI_Irecv(buffer[j].data, MAXSIZE, MPI_CHAR, j, tag,
                  comm, &req[j]);
    }
    else {
      MPI_Get_count(&status, MPI_CHAR, &buffer[i].datasize);
      consume(buffer[i].data, buffer[i].datasize);
    }
  }
}
```

Mais detalhes em programação Paralela BCC 447.

PERFORMANCE ANALYSIS

Como medir tempo em sistemas distribuídos ???

Ler cap IV da disssertação do Joubert

---

[1] *Think time:* intervalo de tempo decorrido entre o momento em que o usuário recebe uma página e o momento em que o usuário requisita outra página. Esta variável modela o tempo que um usuário leva para ler ou assimilar a informação contida numa página. Quanto menor o *think time* maior será a taxa de chegada de requisições no servidor WEB.

[2] Tempo de resposta: intervalo de tempo decorrido entre o momento que serviço é requisitado e o momento em que este serviço acaba de ser atendido.

[3] Tempo de espera: intervalo de tempo durante o qual uma requisição espera na fila do servidor para chegar sua vez de ser atendida.

[4] Tempo de serviço: tempo que o servidor consome para processar a requisição.

Um exemplo no projeto ArchCollect de 2002

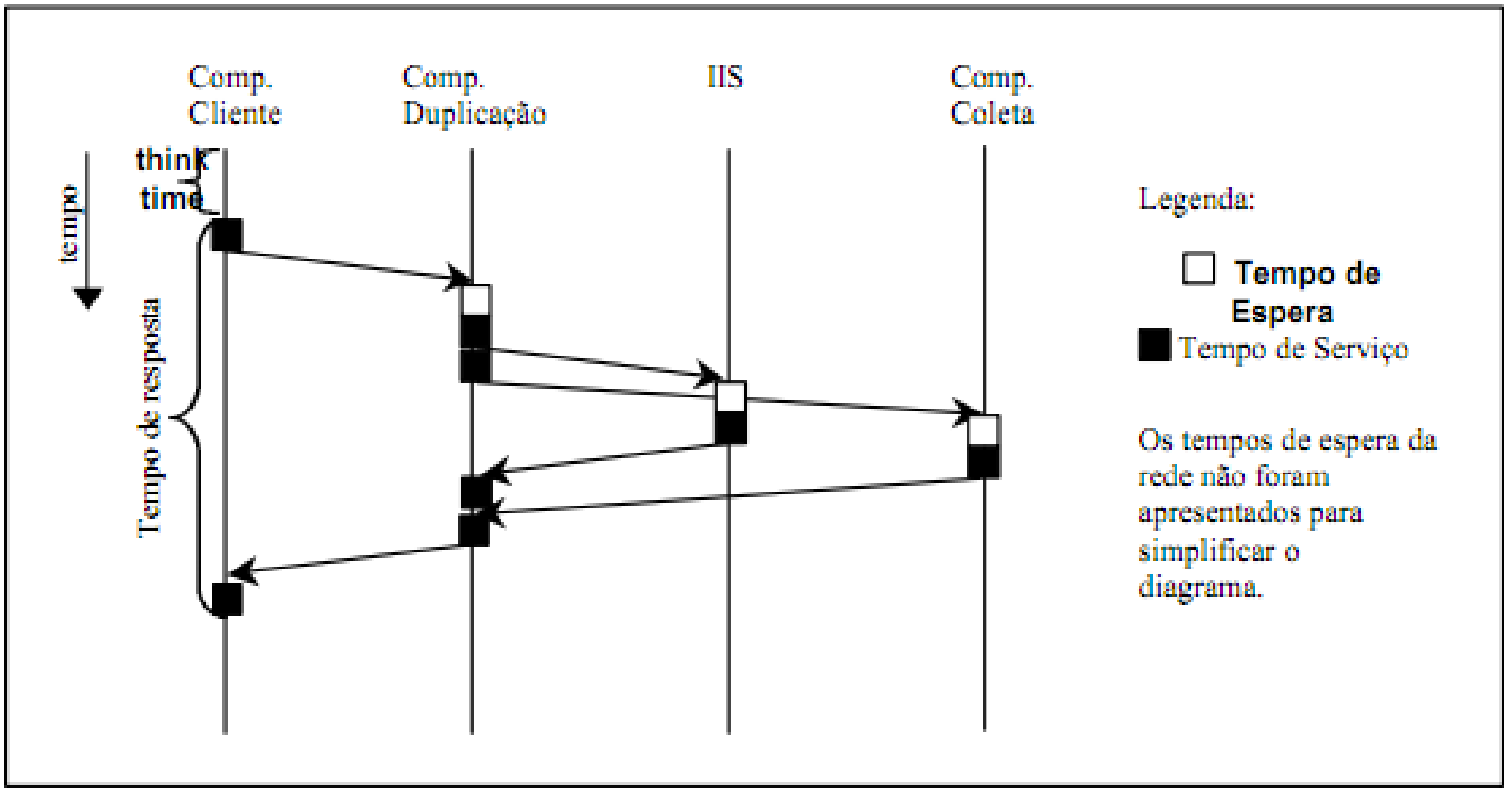


Figura 19. Diagrama de comunicação-processamento para o serviço de coleta de interações.

Vamos ao exemplo da máquina sendo usada em nosso curso

LISTA 2

1) Diferencie os serviços de nomeação e localização.
2) Quais as vantagens e desvantagens das implementações de resolução de nomes Ilustradas abaixo

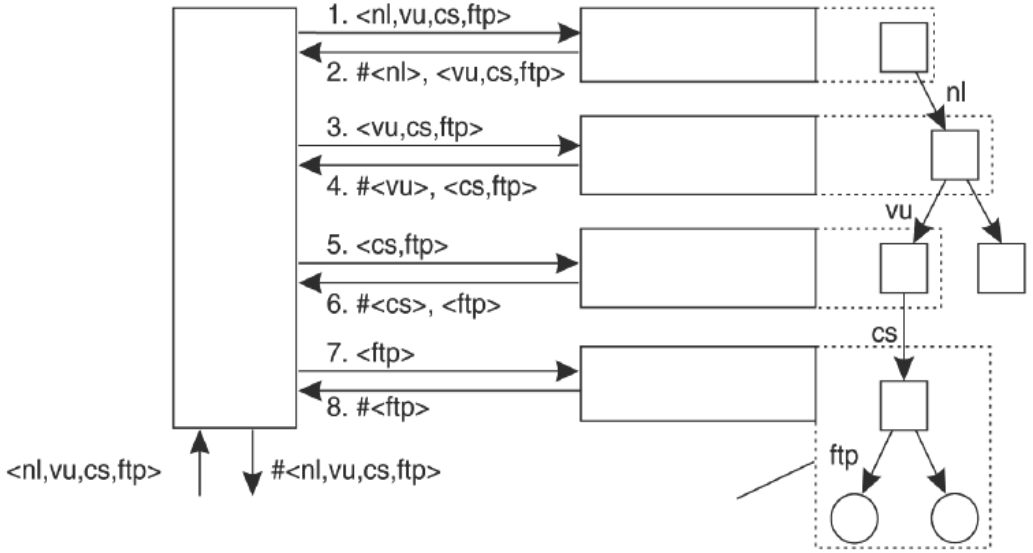


Figura 5.14 Princípio da resolução iterativa de nomes.

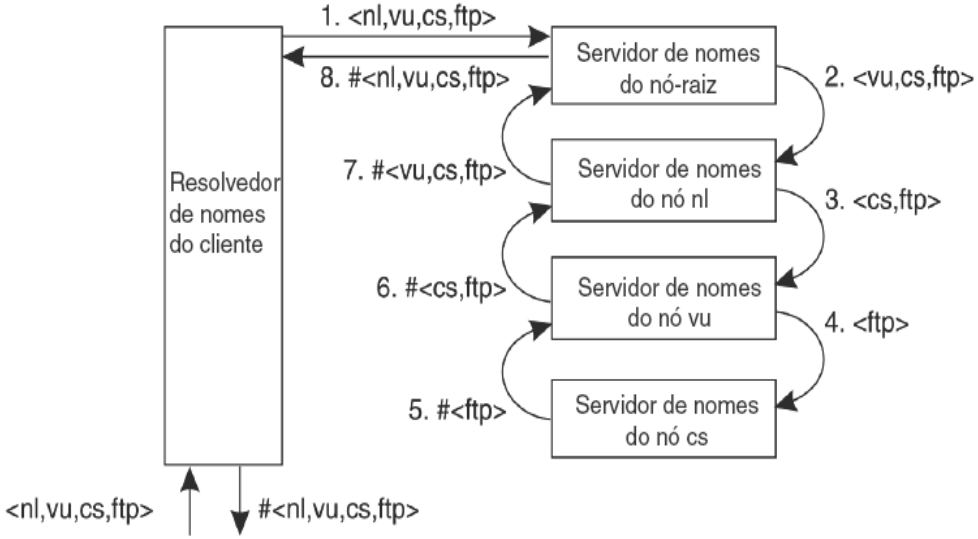


Figura 5.15 Princípio da resolução recursiva de nomes.

3) Descreva (usando figuras inclusive) como foi implementado os serviços de nomeação e localização no seu middleware
4) descreva duas estratégias de sincronização usando relógios físicos
5) descreva duas estratégias de sincronização usando relógios lógicos

5) Como conseguimos resolver o problema de exclusão mútua em sistemas Distribuídos ? Relate pelo menos uma abordagem centralizada e uma distribuída para solução do problema.

6) Como funciona o algoritmo de eleição do valentão, usando para eleger o coordenador de um grupo de processos

7) quais as razões para adotarmos replicação?

8) Explique como garantir consistência sequencial, causal, FIFO e de entrada. Use Exemplos e figuras em sua explicação.

9) Dentre os protocolos de consistência estudados explique e diferencie os protocolos de escrita local, escrita remota e escrita replicada.

10) Explique o que vem a ser o modelo de consistência centrado no cliente

11) Descreva os tipos de protocolos centrados no cliente (monotonic reads, monotonic Writes, read your writes, writes follow reads)

12) Classifique os tipos de falhas num sistema distribuído

13) Explique o acordo bizantino, usado para garantirmos respostas corretas

15) Identifique soluções (incluindo pontos fortes e vulnerabilidades) para as falhas abaixo:
a) O cliente não consegue localizar o servidor
b) A mensagem de requisição do cliente para o servidor se perde
c) O servidor cai após receber uma requisição
d) A mensagem do servidor para o cliente se perde
e) O cliente cai após enviar uma requisição

16) Explique o funcionamento de uma solução escalável para comunicação confiável em grupo (utilize desenhos!!!)

17) Como funciona o commit de uma fase, duas fases e três fases

18) Qual estratégia você escolheria para recuperação de falhas ? Explique em detalhes a estratégia escolhida

19) Quais os mecanismos de segurança mais utilizados em sistemas distribuídos? Descreva em detalhes cada um deles.

20) O que você entende da solução Java RMI? Qual a relação com o que ensinamos na disciplina Sistemas Distribuídos?

21) Descreva como foi implementada a solução de conectores em seu middleware. Considere as classes, diagramas, figuras, etc. em sua explicação.