CORBA
Um Padrão de Arquitetura para Objetos Distribuídos

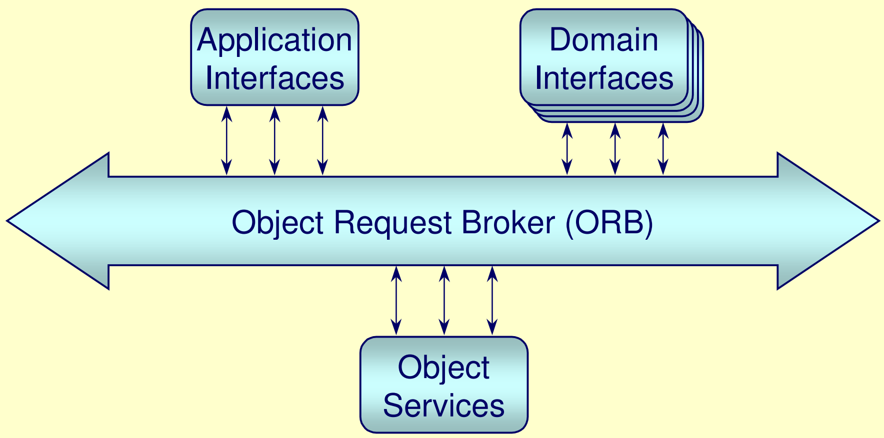
Francisco C. R. Reverbel

O Object Request Broker (ORB)

- Via de comunicação entre objetos (*object bus*), na arquitetura do OMG
- Definido pela especificação CORBA (*Common Object Request Broker Architecture*)
  - 1991: CORBA 1.0
  - 1995: CORBA 2.0
  - revisão atual: CORBA 2.3
  - próxima grande revisão: CORBA 3.0

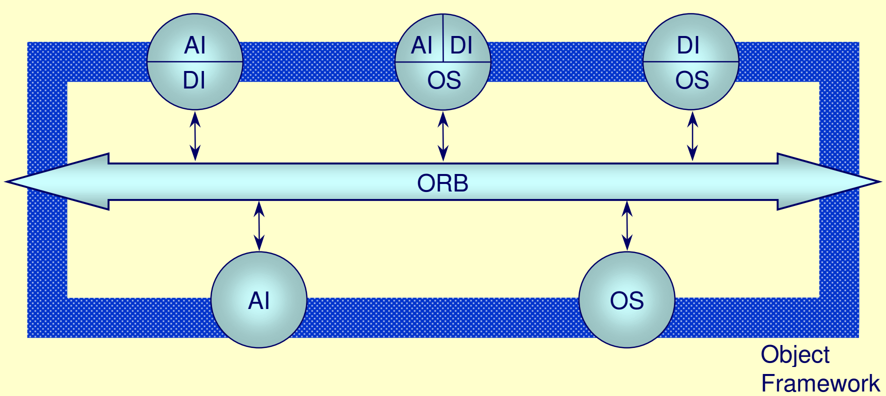
Arquitetura de Gerenciamento de Objetos

OMA (Object Management Arquitecture):



Utilização da OMA

Arcabouço (framework) para aplicações distribuídas:



AI = Application Interfaces

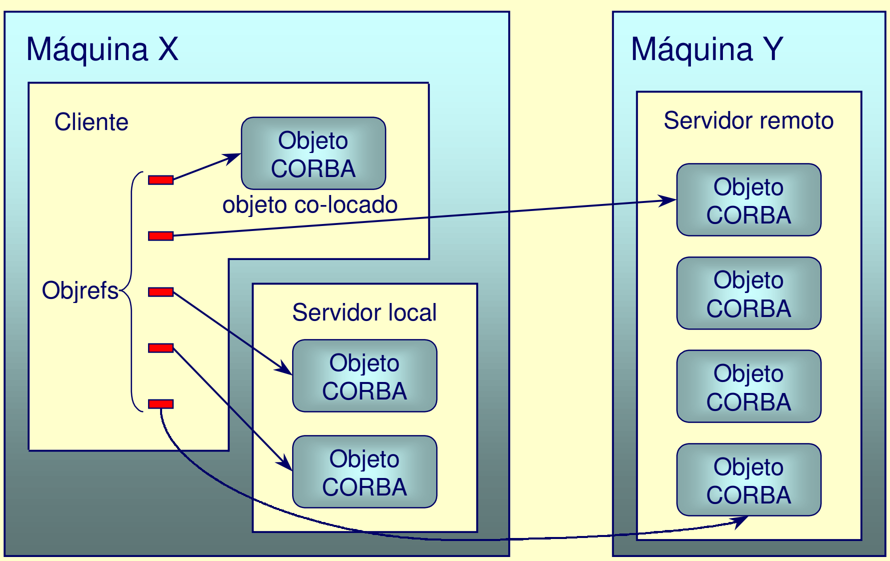
DI = Domain Interfaces

OS = Object Services

CORBA

- Arquitetura cliente/servidor orientada a objetos
- Serviço fundamental do ORB: invocação remota de métodos
- Clientes podem chamar métodos de objetos remotos do mesmo modo que chamam métodos de objetos locais (transparência de localização)
- Para chamar métodos de um objeto o cliente precisa ter uma referência para o objeto

Transparência de Localização

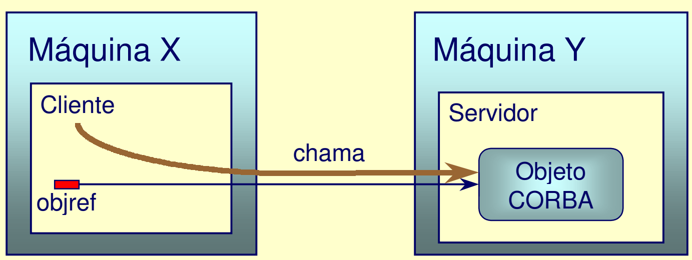


Invocação Remota de Métodos

- O cliente simplesmente chama um método sobre uma object reference
  - O ORB se encarrega de mandar uma mensagem de requisição, esperar a mensagem de resposta e retornar os resultados para o cliente
  - Para o cliente parece uma chamada de método normal
- Independência de linguagens de programação, sistemas operacionais e plataformas de hardware

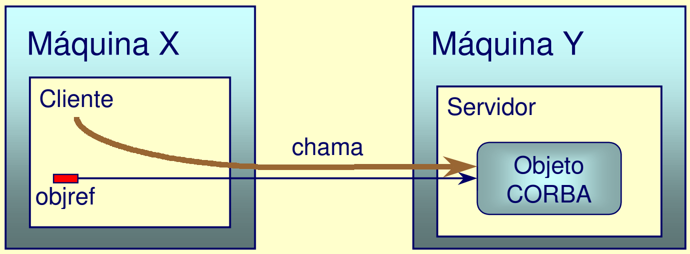
Invocação Remota de Métodos (cont.)

- O que parece acontecer

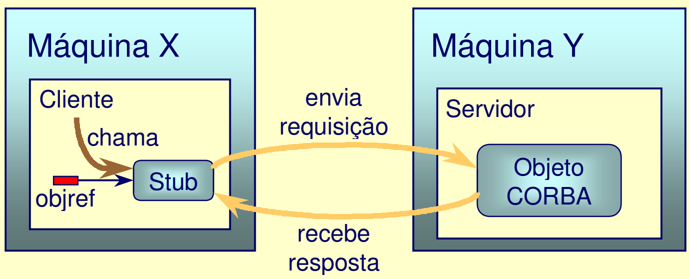


Invocação Remota de Métodos (cont.)

- O que parece acontecer



- O que acontece na verdade

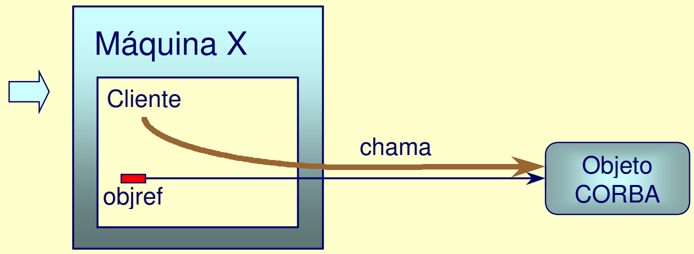


Serventes

- Objetos CORBA são encarnados por entidades da linguagem de programação na qual o servidor foi escrito
- Essas entidades são denominadas <u>serventes</u>
  - Ao longo de sua vida um objeto CORBA pode ser representado por diferentes serventes
- Em C++ ou Java um servente é uma instância de uma classe

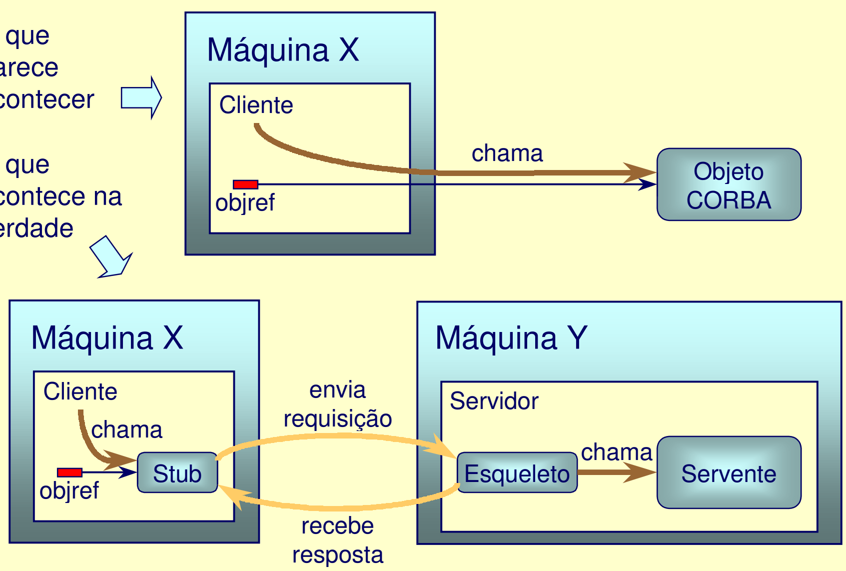
Refinando uma Figura Já Vista

- O que parece acontecer



Refinando uma Figura Já Vista

- O que parece acontecer
- O que acontece na verdade



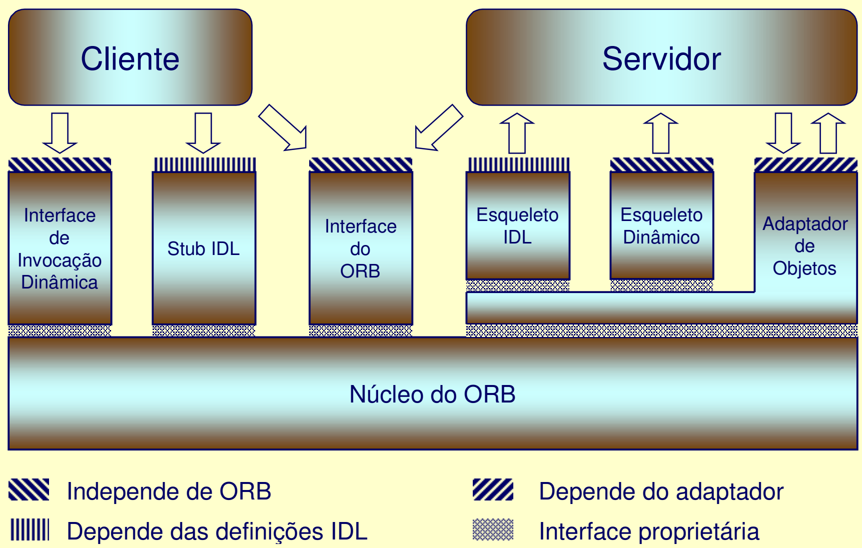
Stubs e Esqueletos

- O usuário não precisa escrever:
  - nem os <u>stubs</u> usado nos clientes
  - nem o <u>esqueletos</u> usados no servidores.
- O que faz o esqueleto:
  - Desempacota os parâmetros recebidos numa mensagem de requisição
  - Chama o método de um servente que implementa a operação requisitada
  - Empacota numa mensagem de resposta os resultados da operação

O Compilador IDL

- Os stubs e esqueletos são automaticamente gerados por um <u>compilador IDL</u>
- O compilador IDL recebe como entrada uma especificação das interfaces dos objetos
- Essa especificação é escrita numa linguagem neutra, a Interface Definition Language (IDL)
  - Independência de linguagem de programação

Componentes de CORBA



Partes da Especificação CORBA

- O núcleo (*core*) do ORB
- A linguagem de definição de interfaces (IDL)
- O repositório de interfaces
- Mapeamentos de IDL para linguagens de programação
- *Stubs* e esqueletos estáticos
- Interfaces de invocação dinâmica (DII) e de esqueleto dinâmico (DSI)
- Adaptadores de objetos (*Object Adapters*)
- O repositório de implementações
- Protocolos (GIOP e IIOP)

O Desenvolvimento de um Sistema

- Etapas Gerais:
  - Defina as interfaces dos objetos e expresse-as em OMG IDL (Interface Definition Language)
  - Gere stubs e esqueletos
  - Implemente objetos serventes
  - Implemente o programa servidor
  - Implemente aplicações clientes

OMG IDL

- Linguagem declarativa usada para especificar interfaces de objetos
  - Sintaxe baseada em C++
  - Suporta herança de interfaces
  - Todas as interfaces de objetos são derivadas de **`CORBA::Object`**
  - CORBA é especificada em IDL
    - de modo a ser independente de linguagem de programação
  - Oferece tipos básicos (pré-definidos) e permite definição de tipos pelos usuários

Mapeamentos de IDL

- Definem como IDL é traduzida para as diferentes linguagens de programação
- O OMG padronizou mapeamentos de IDL para C, C++, Smalltalk, COBOL, Ada e Java
  - Há mapeamentos para outras linguagens, mas ainda são proprietários (não padronizados)

Exemplo de Interface IDL

```
// IDL
module Stock {
    exception UnknownStock {
        string name;
    };
    interface Quoter {
        string name();
        long value(in string stock)
            raises(UnknownStock);
    };
};
```