CORBA

Common Object Request Broker Architecture

Unicamp

Centro de Computação

Rubens Queiroz de Almeida

queiroz@unicamp.br

Objetivos

- Apresentação Tecnologia CORBA
- Conceitos Básicos e Terminologia
- Considerações Gerais

O Problema

Necessidade de compartilhamento de informações entre empresas e integração de hardware e software de plataformas diversas de forma a resolver problemas presentes e futuros

Facilidades

A arquitetura CORBA permite:

- Acesso a recursos computacionais e informações distribuídas a partir de aplicações populares
- Tornar dados e aplicações legadas disponíveis como recursos de rede
- Atualizar sistemas baseados em rede de forma a refletir novas topologias ou recursos

Integração das Aplicações

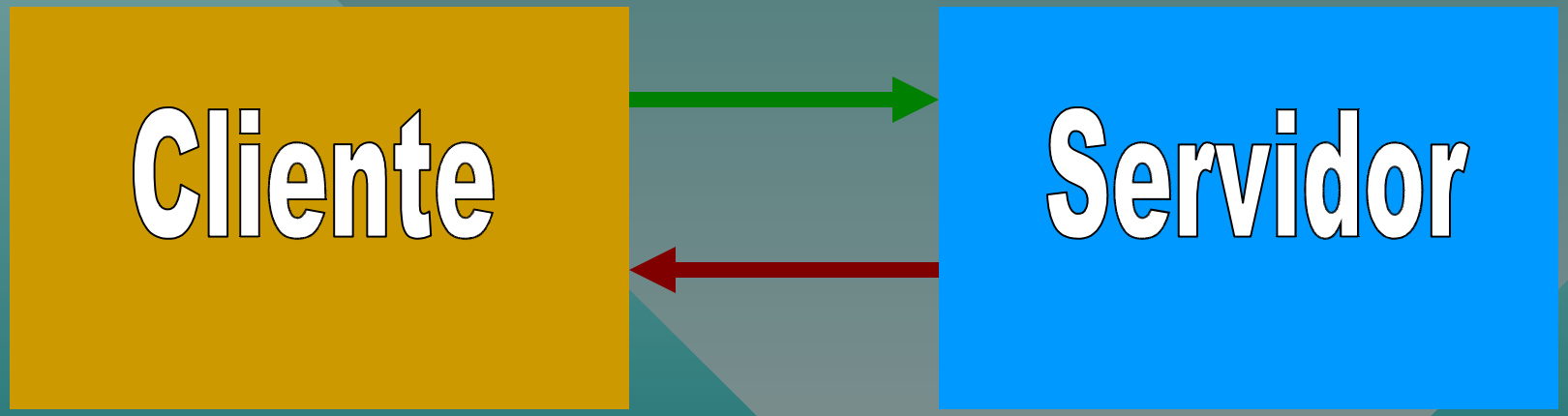
- Integração baseada em um modelo orientado a objetos
  - Modelo provê técnicas para análise, projeto e implementação de software que seja extensível, reusável e menos dispensioso para produzir e manter
- Próxima geração de software

Computação Distribuída com Objetos

Dois ingredientes essenciais

- Associação da computação distribuída com um modelo de objetos
- Uso de um corretor (broker)

Arquitetura Cliente/Servidor



CORBA Broker

Computação Distribuída

Dois ou mais elementos de software compartilhando informação entre si. Grande parte das implementações de computação distribuída existentes são baseados no modelo cliente/servidor:

- software cliente: pedidos
- software servidor: informação ou serviço

Benefícios da Computação Distribuída

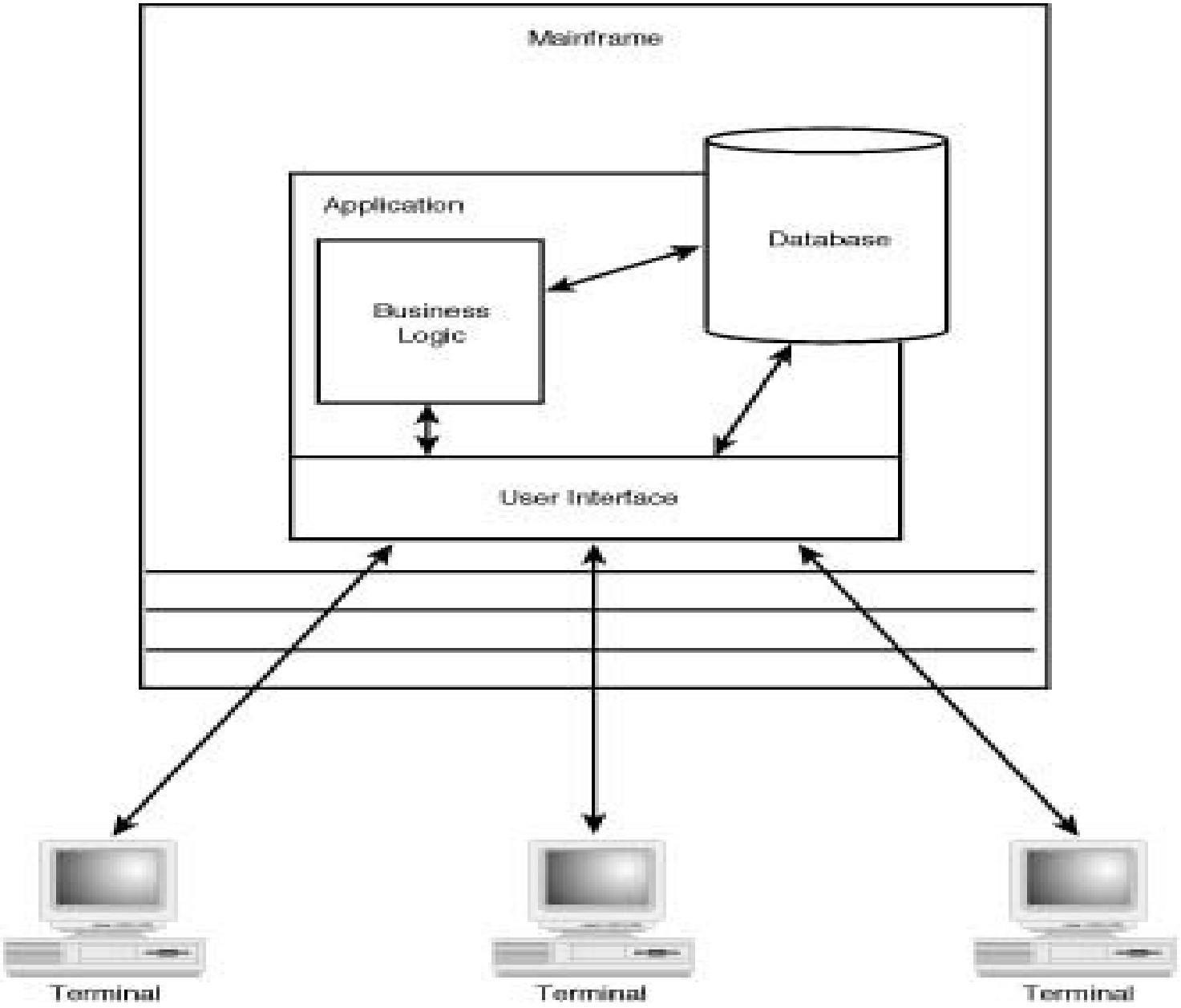
Uso mais eficiente de recursos computacionais

- Compartilhamento de recursos escassos e dispendiosos
- Distribuição da carga computacional
- Execução de aplicações nos ambientes mais adequados

Evolução de Sistemas Distribuídos

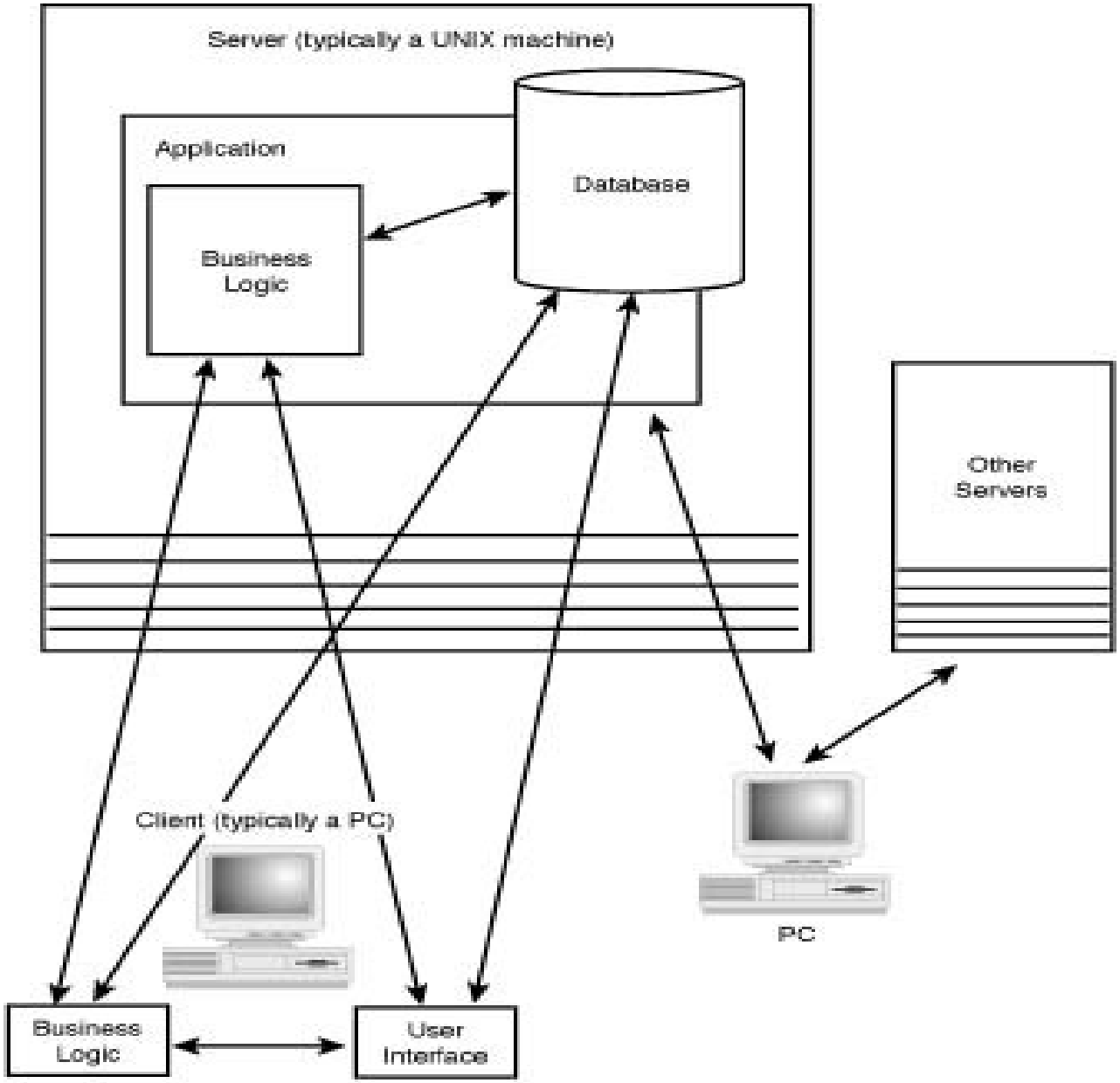
Sistemas monolíticos e mainframes

A interface com o usuário, a lógica do negócio e a funcionalidade de acesso aos dados estavam todas contidas em uma única aplicação


Mainframe
Application
Database
Business Logic
User Interface
Terminal
Terminal
Terminal

Arquitetura Cliente/Servidor

- Parte do processamento realizado por microcomputadores na mesa dos usuários
- Computação mais acessível
- Maior poder para os usuários

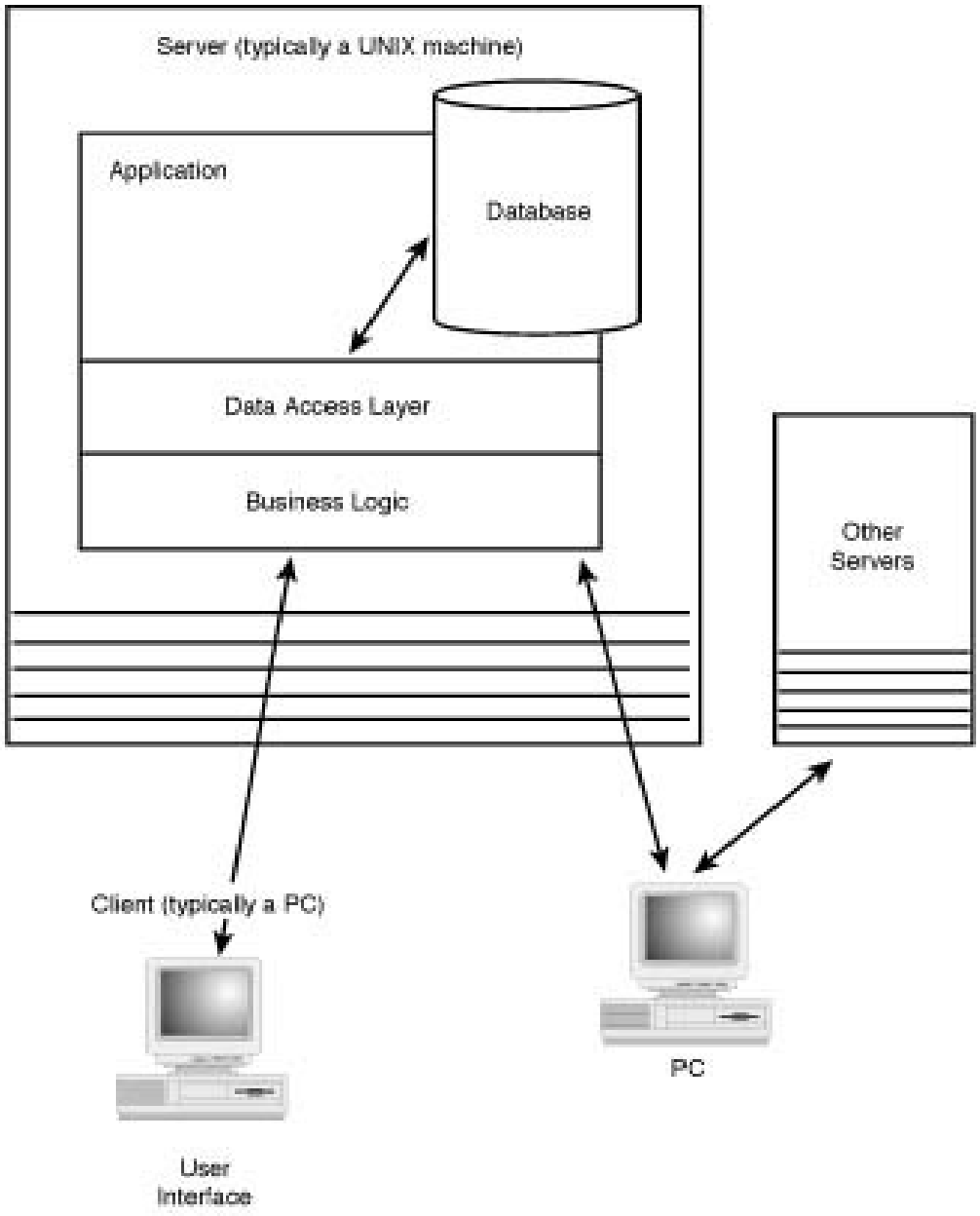

Server (typically a UNIX machine)
Application
Database
Business Logic
Other Servers
Client (typically a PC)
PC
Business Logic
User Interface

Desvantagens

- Funcionalidade de acesso ao banco de dados e lógica do negócio contidas no componente cliente
- Mudanças na lógica do negócio envolvem a substituição dos clientes
- Aplicações mais frágeis

Arquitetura Cliente/Servidor de Múltiplas Camadas

- Modelo mais comum são sistemas particionados em três camadas lógicas:
  - interface com o usuário
  - regras do negócio
  - acesso ao banco de dados


Server (typically a UNIX machine)
Application
Database
Data Access Layer
Business Logic
Other
Servers
Client (typically a PC)
PC
User
Interface

Vantagens

- Cliente isolado das mudanças no restante da aplicação
- Componentes executáveis menores resultando em maior facilidade na distribuição das aplicações
  - múltiplos bancos de dados, servidores, distribuição da carga de processamento
- Maior isolamento entre as camadas da aplicação
  - mudanças na aplicação não afetam o componente cliente

Sistemas Distribuídos

O modelo de sistemas distribuídos expõe toda a funcionalidade da aplicação como objetos, cada um dos quais pode usar qualquer dos serviços oferecidos por outros objetos do sistema, sendo extremamente flexível e configurável

Interação entre Objetos

- Flexibilidade obtida pela adoção da definição de interfaces específicas para cada componente
- A interface de cada componente especifica para os demais quais serviços são oferecidos e como devem ser usados
- Enquanto a interface de um componente se mantiver inalterada sua implementação pode ser radicalmente alterada sem afetar os demais objetos
  - P. ex., informação armazenada em bancos de dados relacionais podem ser mudadas para bancos de dados orientados a objetos

Interfaces

- Protocolo de comunicação a ser utilizado entre dois componentes de um sistema
- Descreve que serviços são oferecidos por qual componente e o protocolo para utilizar estes serviços
- Para objetos, a interface pode ser entendida como um conjunto de métodos definidos por aquele objeto, incluindo os parâmetros de entrada e saída

Serviços de Catálogo (Directory Services)

Conjunto de serviços que permitem com que objetos (servidores, empresas ou até mesmo pessoas) sejam localizados por outros objetos

Monitores de Transação

Mantém o sincronismo entre diversos componentes, garantindo um estado consistente entre todos os participantes em uma transação, cancelando ou efetuando operações

Porque CORBA?

CORBA oferece um mecanismo padrão para a definição de interfaces entre componentes e ferramentas para facilitar a implementação destas interfaces nas linguagens escolhidas pelos desenvolvedores

Independência de Linguagens de Programação

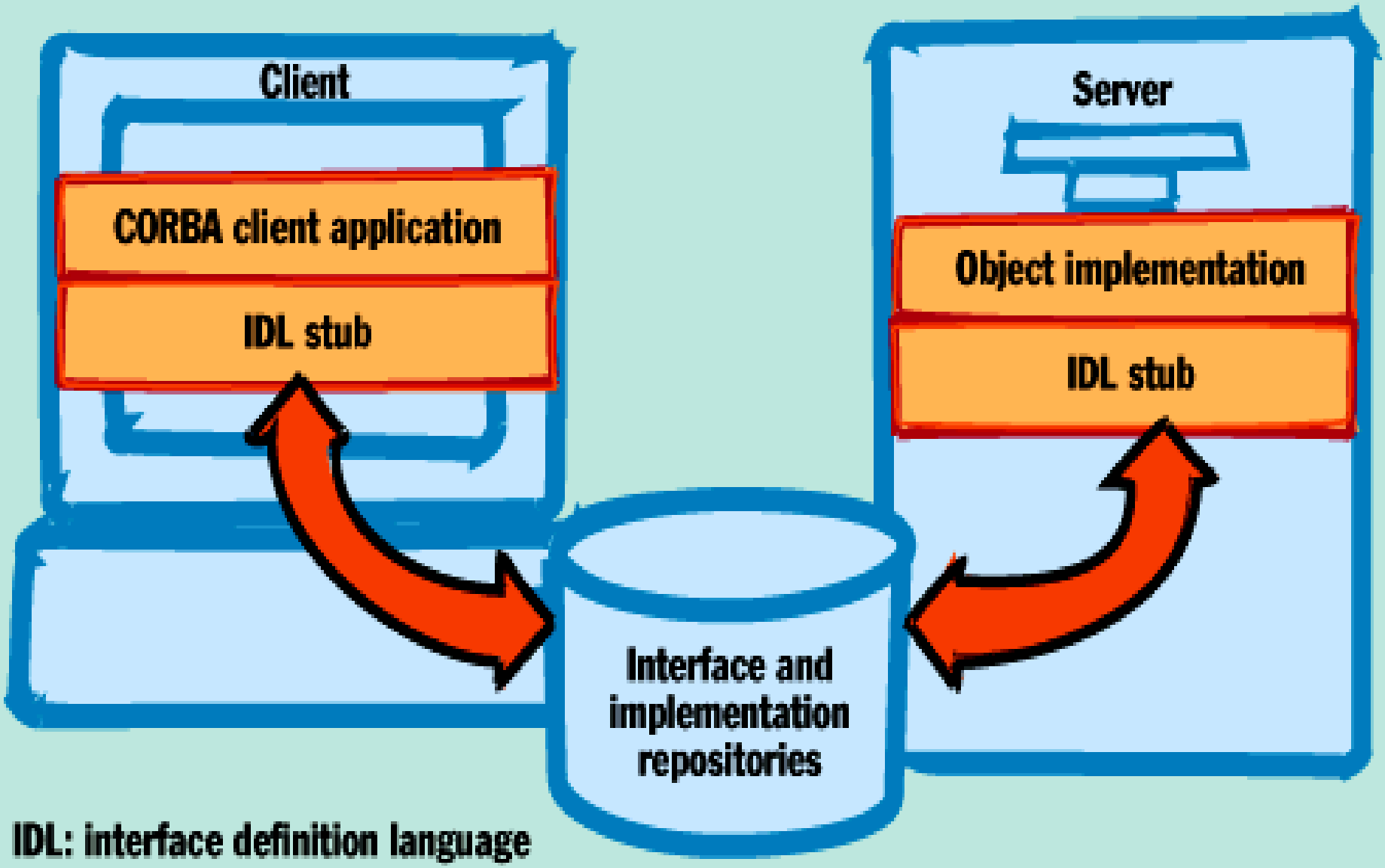
- objetos e clientes CORBA podem ser implementados em qualquer linguagem de programação
- objetos CORBA não precisam saber como foram implementados os objetos com os quais se comunicam

Independência de Plataforma Computacional

Objetos CORBA podem ser usados em qualquer plataforma para a qual exista a implementação de um CORBA ORB (Object Request Broker), ou seja, praticamente qualquer sistema computacional existente

CORBA
Visão Geral

CORBA Architecture



IDL: interface definition language
CORBA: Common Object Request Broker Architecture

Object Request Broker (ORB)

Componente de software cuja função é facilitar a comunicação entre objetos

- localização de objetos remotos
- passagem e recepção de parâmetros (marshaling/unmarshaling)

Marshaling/Unmarshaling

- Marshaling - formatação dos parâmetros para transmissão via rede
- on-the-wire format - formato para transmissão via rede
- Unmarshaling - transformação do formato on-the-wire para o formato local
- Processo ocorre sem intervenção do programador
- Diferenças entre plataformas são resolvidas pelo ORB

Interface Definition Language (IDL)

- Interfaces CORBA são definidas em uma linguagem de sintaxe neutra conhecida como Interface Definition Language ou **IDL**
- Fundamental para a independência de linguagem
- Interfaces descritas em IDL podem mapear para qualquer linguagem de programação
- Seu único propósito é definir interfaces e estruturas de dados
- Não é utilizada para escrever algoritimos

IDL: Vantagens/Desvantagens

Vantagens

- IDL é uma linguagem neutra que permite que clientes e servidores sejam implementados em linguagens diferentes. Um cliente Java pode interagir com aplicações COBOL empacotadas com CORBA
- IDL permite que a especificação dos serviços seja separada da implementação.

Desvantagens

- Difícil de usar
- Um passo extra de compilação é necessário

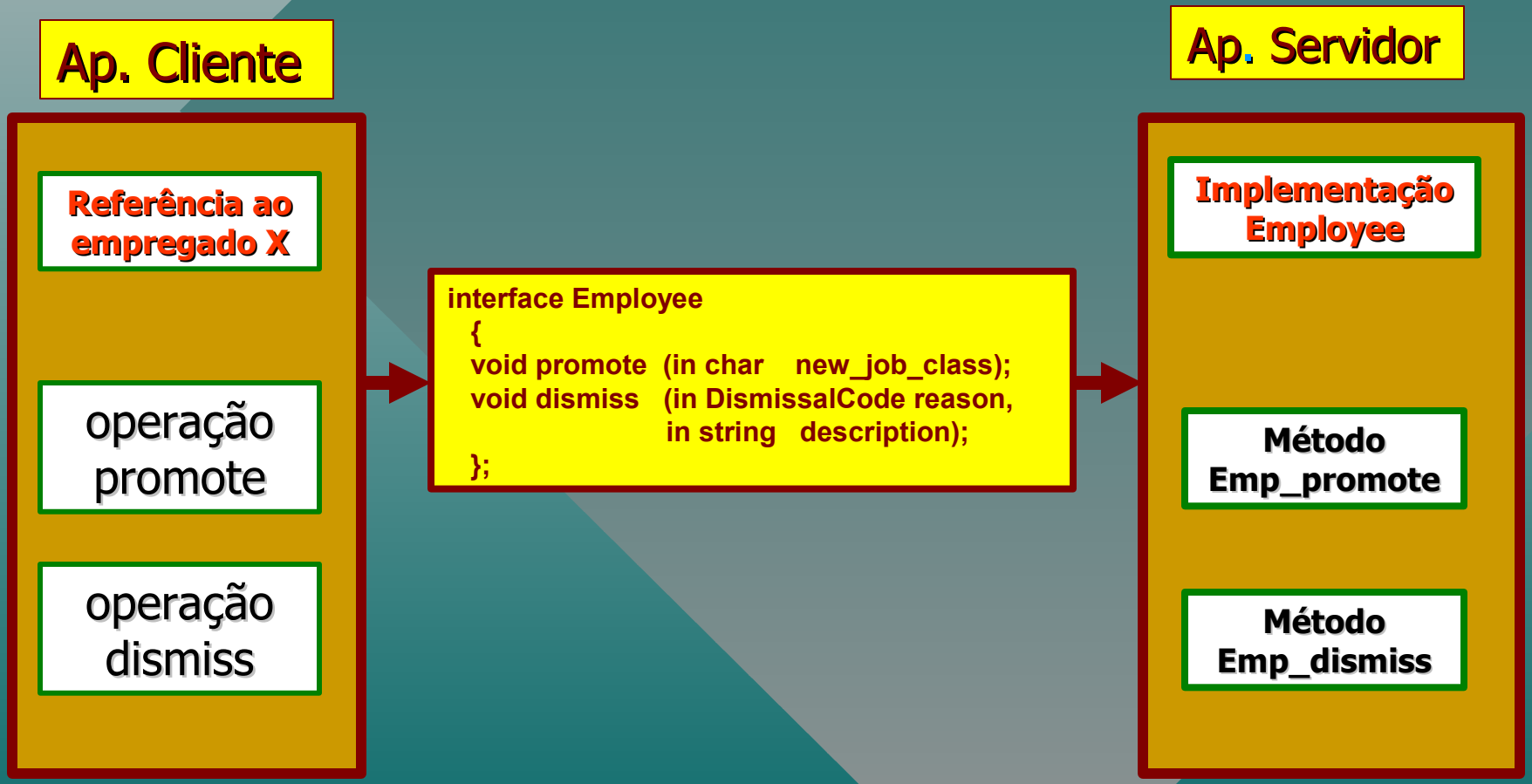
Arquivo OMG IDL

Descreve o formato dos dados, operações e objetos que o cliente pode usar para fazer um pedido e que o servidor precisa prover para a implementação de um objeto

Exemplo

```
interface Employee
    {
    void promote (in char     new_job_class);
    void dismiss    (in DismissalCode reason,
                     in string    description);
    };
```

Cliente e implementação no servidor

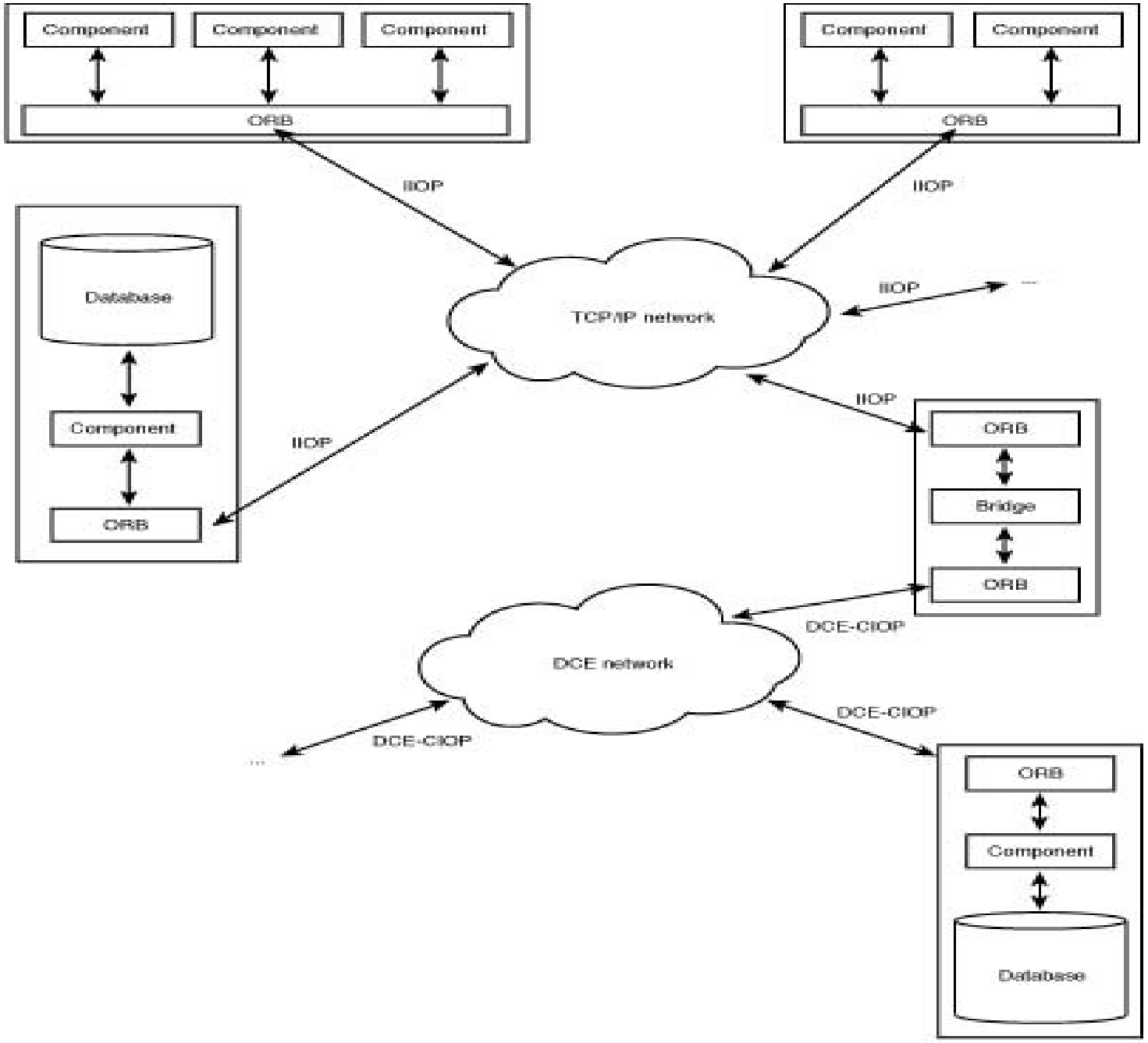


Mapeamento de Linguagens

Mapeamento de linguagem é uma especificação que mapeia as construções da linguagem IDL às construções de uma determinada linguagem de programação

Protocolos Inter-ORB

- Especificação CORBA é neutra em relação a protocolos de rede
- Padrão CORBA especifica um padrão geral (GIOP - General Inter-ORB Protocol)
- IIOP - Internet Inter-ORB Protocol
  - CORBA ORBs se comunicam utilizando o protocolo IIOP
  - mais popular (transportado sobre TCP/IP)


Component
Component
Component
ORB
Component
Component
ORB
IIOP
IIOP
Database
Component
ORB
TCP/IP network
IIOP
...
IIOP
IIOP
ORB
Bridge
ORB
DCE-CIOP
DCE network
DCE-CIOP
DCE-CIOP
...
ORB
Component
Database

Modelo de Comunicações

- Cliente: objetos que invocam um ou mais métodos em outros objetos
- Servidor: aplicação que cria objetos CORBA e torna os serviços oferecidos por estes objetos disponíveis para outras aplicações

Modelo de Comunicações ...

- IOR - Interoperable Object References
  - quando um componente deseja acessar um objeto CORBA, é necessário obter um IOR para este objeto

CORBA: Modelo de Objetos

- Toda comunicação entre objetos se dá por meio de referências a objetos (IOR)
- BOA - Basic Object Adapter
  - Provê a objetos CORBA um conjunto comum de métodos para acessar funções ORB. Estas funções variam desde a autenticação de usuários a ativação de objetos. Segundo a especificação CORBA, o BOA deve estar disponível em toda implementação ORB

OMG
Object Management Group

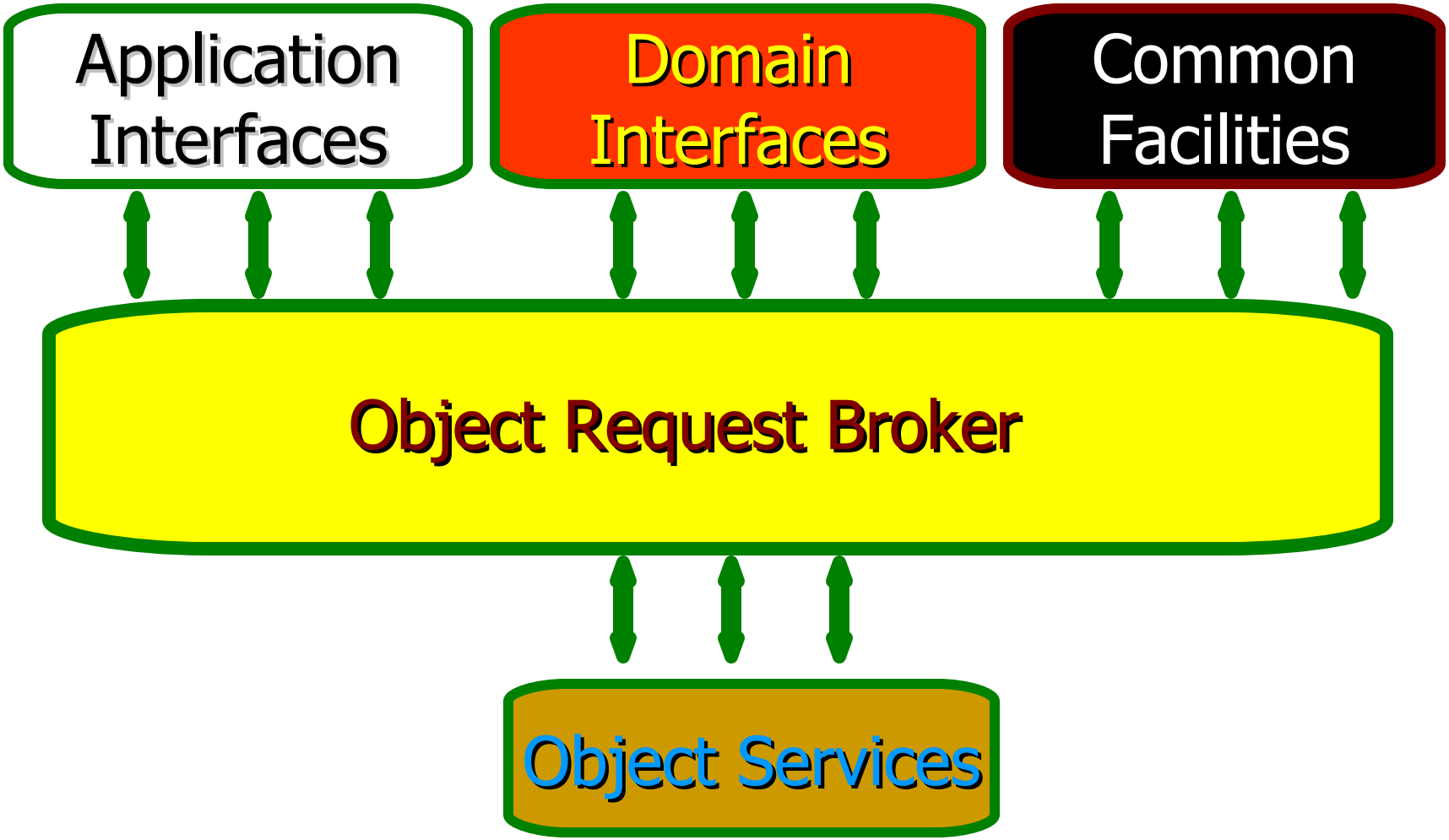
- Organização sem fins lucrativos fundada em 1989, com 8 membros
- Conta hoje com mais de 800 membros
- Seu objetivo é promover a teoria e o uso de tecnologia OO para sistemas distribuídos

OMA:
Object Management Architecture

- Conjunto de padrões que definem a arquitetura sob a qual aplicações distribuídas são construídas
- CORBA: parte da arquitetura OMA

OMA: Componentes

- ORB (Object Request Broker)
- serviços de objetos (CORBAservices)
- recursos comuns (CORBAfacilities)
- Interfaces de domínio (domain interfaces)
- objetos


Application Interfaces
Domain Interfaces
Common Facilities
Object Request Broker
Object Services

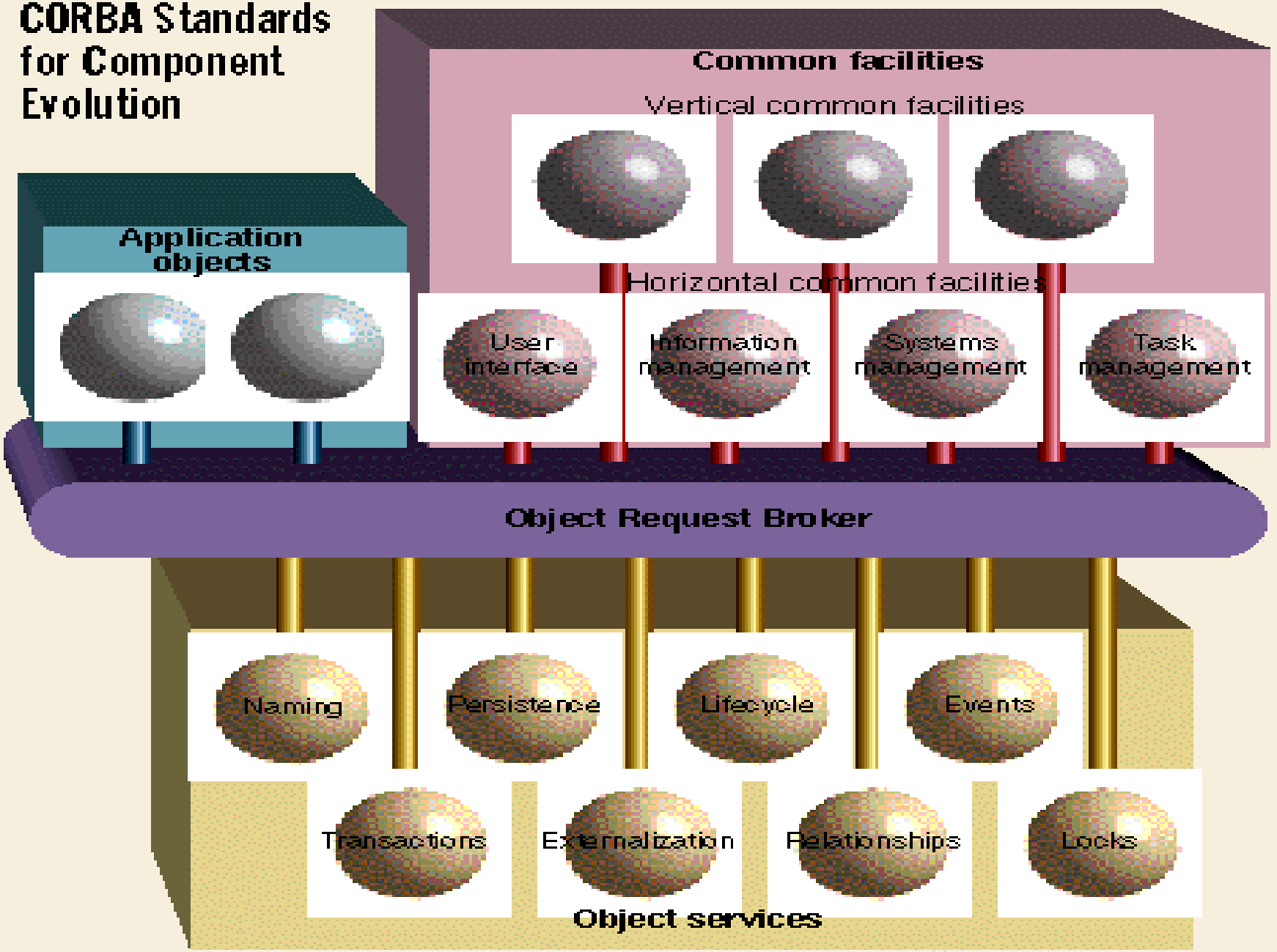
Stubs / Skeletons

- Client Stub
  - código que permite um componente cliente acessar um componente servidor. Compilado juntamente com a parte cliente da aplicação
- Server Skeleton
  - código integrado quando da implementação do servidor
- Código gerado quando a definição das interfaces IDL são compiladas

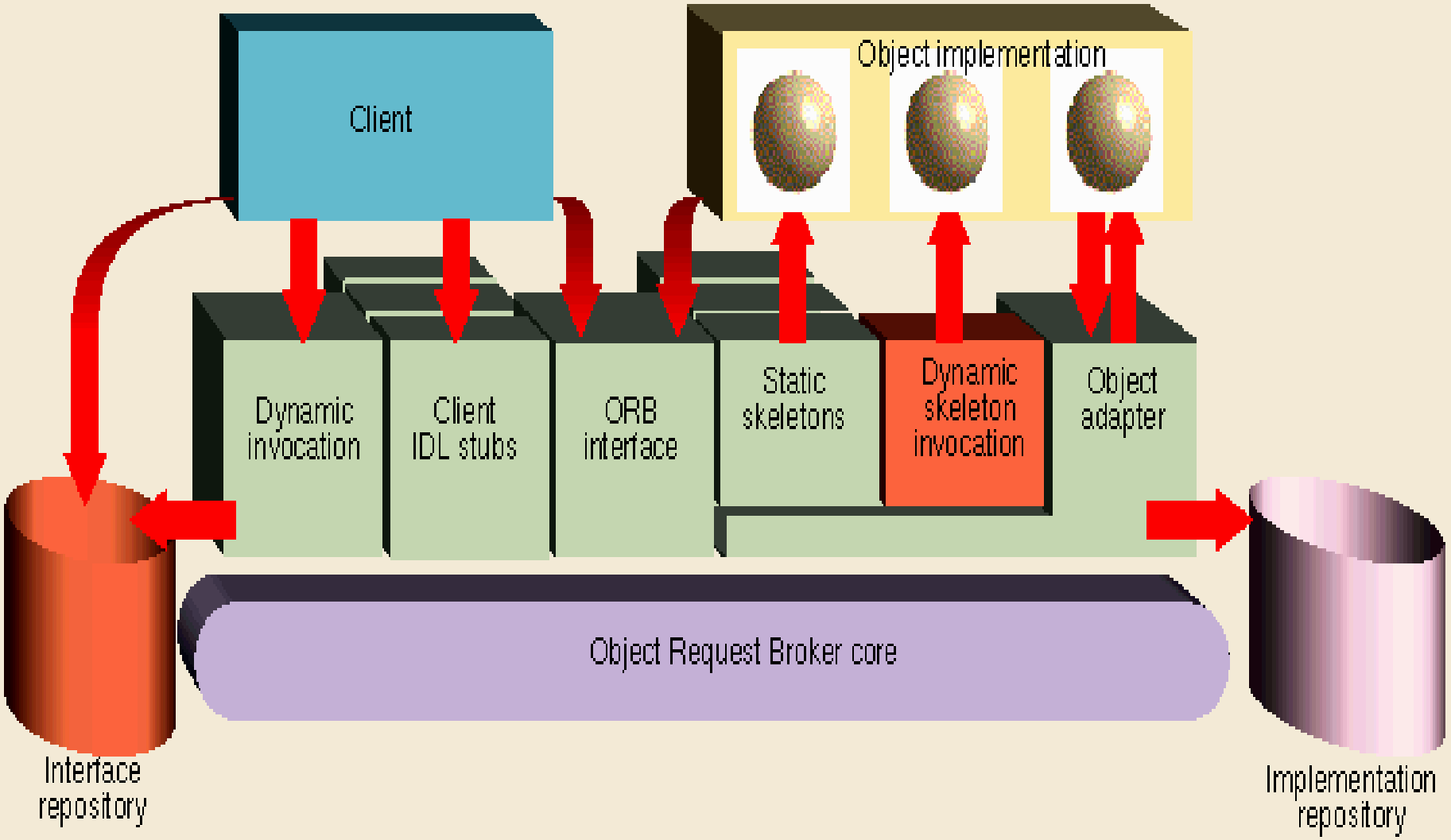
CORBAservices/CORBAfacilities

- Não fazem parte da especificação CORBA, mas são um componente complementar da OMA
- Serviços e facilidades horizontais (comuns a todas as empresas) e verticais (comuns a um setor específico)

CORBA Standards for Component Evolution



The Structure of a CORBA 2.0 Object Request Broker



Aplicações Legadas

- Especificar a funcionalidade de aplicações legadas em termos de interfaces
- Criar envelopes (wrappers) CORBA para conversar com aplicações legadas
- Fingir que a aplicação legada é um objeto e desenvolver normalmente

CORBA Wrappers

- Uma aplicação legada escrita em Cobol pode ser usada em ambiente CORBA através da codificação de um "wrapper" usando as ligações para COBOL definidas pela OMG
- O "wrapper" conecta a aplicação ao servidor. Devido ao uso de IDL e estabelecimento das conexões via ORB, os clientes não se dão conta de estarem se comunicando com uma aplicação legada. A aplicação COBOL por sua vez pensa estar conversando com outras aplicações COBOL

Linux e CORBA

- GNOME
  - Utiliza CORBA para troca de dados entre aplicações
  - Red Hat ORBit http://www.labs.redhat.com/orbit
- IONA Orbix
  - versão para Linux em breve

Linux e CORBA ...

- IBM Websphere
  - capacidade de executar nativamente, em breve, transações CORBA em Linux
- AT&T omniORB
  - utilizado por mais de 700 desenvolvedores (ALCATEL entre eles)
  - implementação CORBA enxuta
  - Projeto Active Badge
  - Livre (GPL)
  - http://www.uk.research.att.com/omniORB

CORBA: Alternativas

- sockets
- RPC (Remote Procedure Call)
- DCE (Distributed Computing Environment)
- Microsoft DCOM (Distributed Computing)
- JAVA RMI (Remote Method Invocation)

Bibliografia

- Understanding CORBA
  Randy Otte, Paul Patrick, Mark Roy
  Prentice Hall
- Byte Magazine
  http://www.byte.com
- Object Management Group
  http://www.omg.org
- Links
  http://www.Dicas-L.unicamp.br/hotlinks/