

# Integração de Sistemas Distribuídos

Fernando Trinta & Windson Viana

# Sistema Distribuído

“A distributed system as one in which hardware or software components located at networked computers communicate and coordinate their actions only by passing messages”

George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair. 2011. *Distributed Systems: Concepts and Design* (5th ed.). Addison-Wesley Publishing Company, USA.

# Sistema Distribuído

- Heterogeneidade
- Abertura
- Segurança
- Tolerância a falhas
- Escalabilidade
- Concorrência
- Transparência

# Gerações de Sistemas Distribuídos

<i>Sistemas distribuídos</i>	<i>Primitivos</i>	<i>Adaptados para Internet</i>	<i>Contemporâneos</i>
<i>Escala</i>	Pequenos	Grandes	Ultragrandes
<i>Heterogeneidade</i>	Limitada (normalmente, configurações relativamente homogêneas)	Significativa em termos de plataformas, linguagens e <i>middleware</i>	Maiores dimensões introduzidas, incluindo estilos de arquitetura radicalmente diferentes
<i>Sistemas abertos</i>	Não é prioridade	Prioridade significativa, com introdução de diversos padrões	Grande desafio para a pesquisa, com os padrões existentes ainda incapazes de abranger sistemas complexos
<i>Qualidade de serviço</i>	Em seu início	Prioridade significativa, com introdução de vários serviços	Grande desafio para a pesquisa, com os serviços existentes ainda incapazes de abranger sistemas complexos

# Sistema Distribuído

- Maior Complexidade na sua implementação
  - Falhas de Comunicação
  - Falhas individuais e independentes
  - Falta de um relógio global
- Características das aplicações atuais
  - Alta Complexidade
  - Grande Heterogeneidade
  - Distribuídos
  - Heterogêneos

# Elementos Arquiteturais

- Quais entidades se comunicam em sistema distribuído?
- Como elas se comunicam (Qual paradigma de comunicação)?
- Quais os papéis e responsabilidades relacionados a cada componente na arquitetura?
- Qual a distribuição física destes componentes?

# Elementos Arquiteturais

- Quais entidades se comunicam em sistema distribuído?
  - Processos
  - Threads
  - Objetos
  - Componentes
  - Serviços

## Elementos Arquiteturais

- Como elas se comunicam (Qual paradigma de comunicação)?
  - Comunicação direta
  - Invocação Remota
    - RPC
    - RMI
  - Comunicação Indireta
    - Comunicação em Grupo
    - Publish/Subscribe
    - Espaço de Tuplas
    - Memória Distribuída Compartilhada

## Elementos Arquiteturais

- Quais os papéis e responsabilidades relacionados a cada componente na arquitetura?
  - Cliente/Servidor
  - Peer-to-peer

# Elementos Arquiteturais

- Qual a distribuição física destes componentes?
  - Clusters
  - Caching
  - Código Móvel
  - Agentes Móveis

# Padrões (ou Estilos) Arquiteturais

- Todo sistema/aplicação possui uma arquitetura
- Aplicações em um mesmo domínio (Web, Colaborativos, Móvel) compartilham características em suas arquiteturas
- Estilo/Padrão

“Um estilo arquitetural define uma família de sistemas em termos de um padrão de organização estrutural”

**BUSCHMANN, Frank, MEUNIER, Regine, ROHNERT, Hans, SOMMERLAD, Peter, STAL, Michael. Pattern-Oriented Software Architecture: A System of Patterns. Vol. 1. John Wiley & sons Ltd.**

# Estilos Arquiteturais

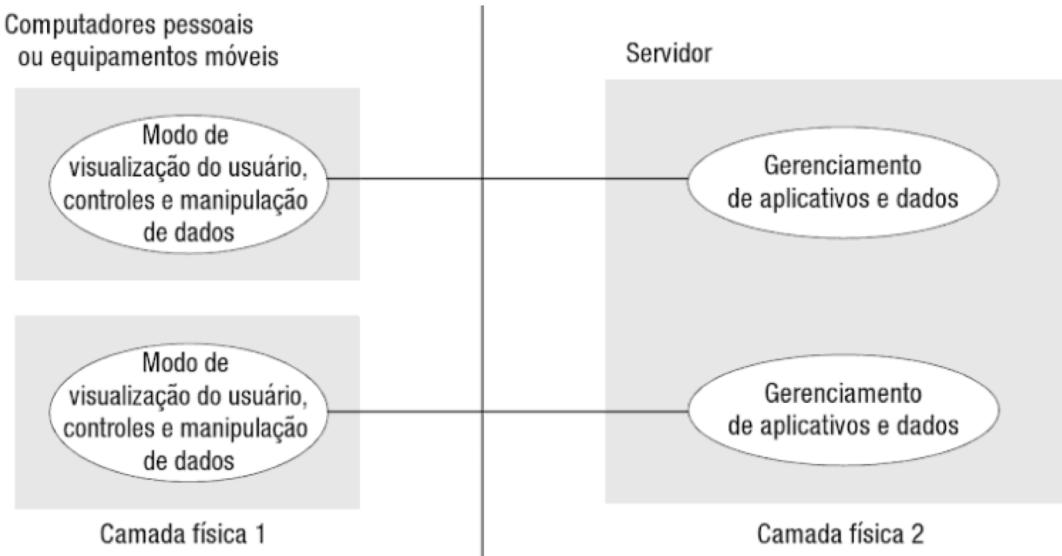
- Model-View-Controller
- Blackboard
- Pipers and Filters
- Layers
- Broker

# Estilos Arquiteturais

- Model-View-Controller
- Blackboard
- Pipers and Filters
- Layers
- Broker

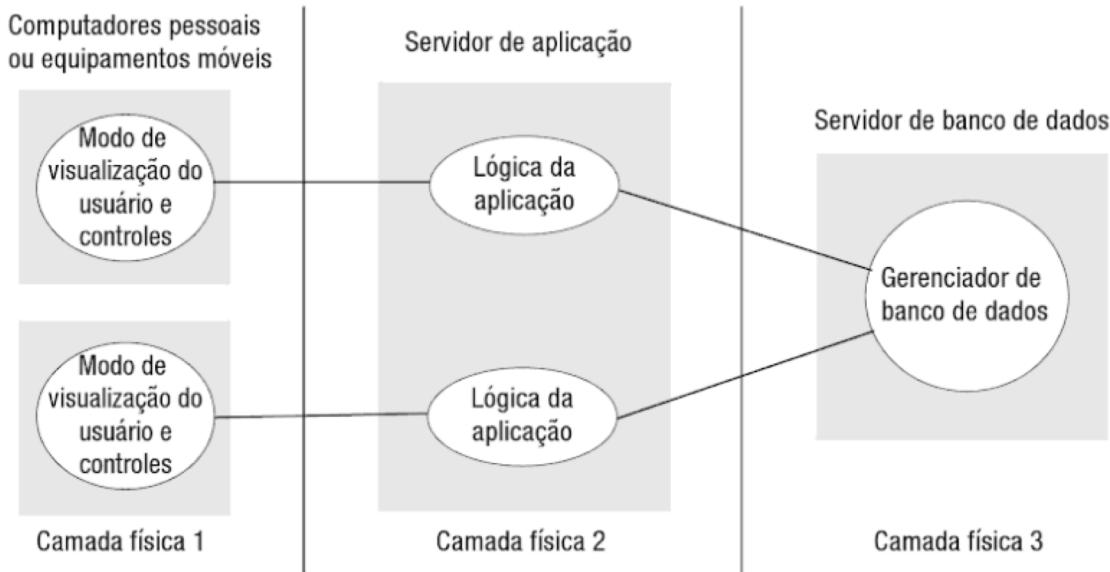
# Layers vs Tiers

- Camadas Físicas e Camadas Lógicas



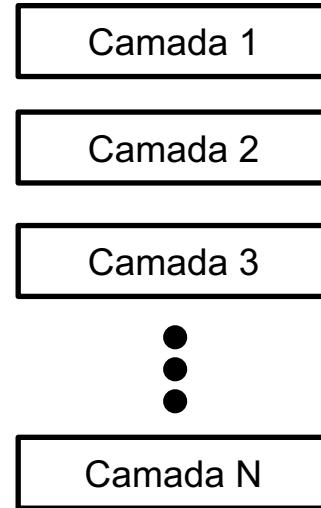
# Layers vs Tiers

- Camadas Físicas e Camadas Lógicas



# Layers

- Diferentes camadas sobrepostas, com propósitos definidos e ofertando serviços à camada superior
- Modelo OSI
- Modelo base para o conceito de Middleware



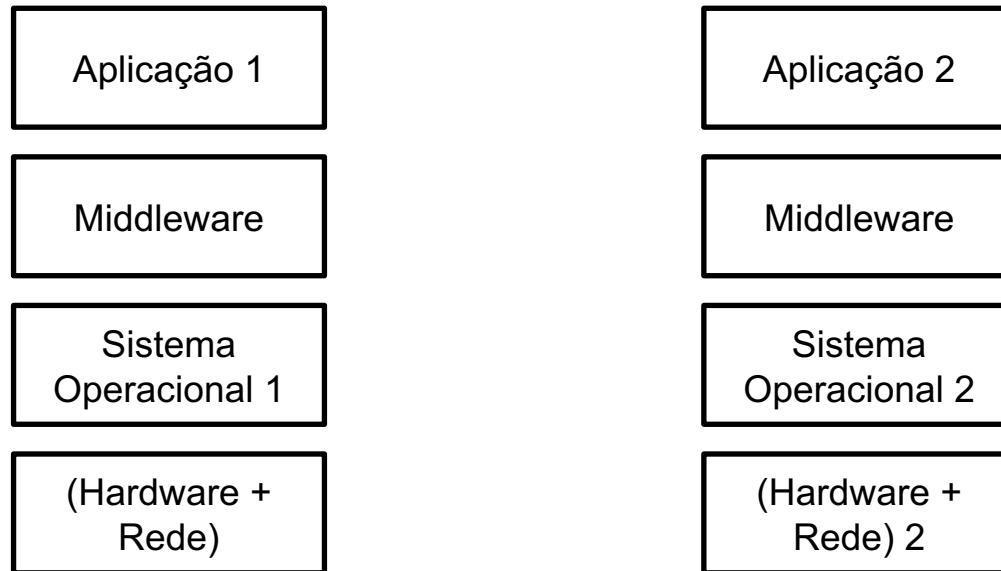
# Layers



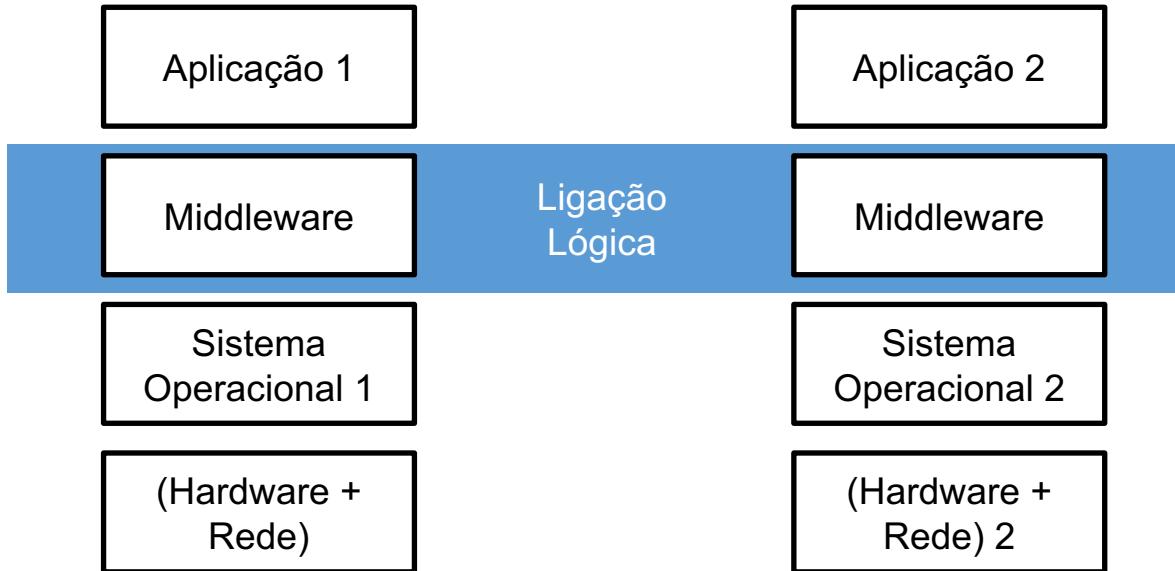
# Middleware

“Camada de Software que provê uma abstração de programação com objetivo de mascar a heterogeneidade de protocolos de rede, arquiteturas de hardware, sistemas operacionais e linguagens de programação”

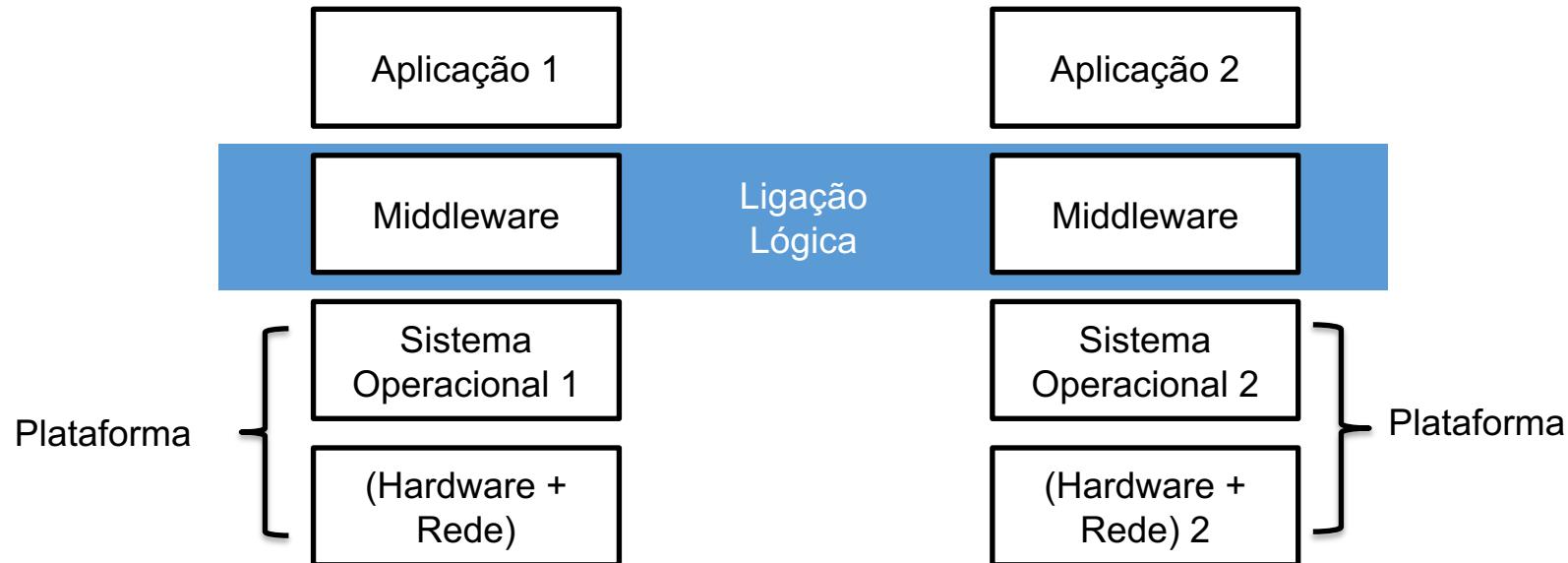
# Visão de um Middleware



# Visão de um Middleware



# Visão de um Middleware



# Middleware

**Middleware = API + Conjunto de Serviços**

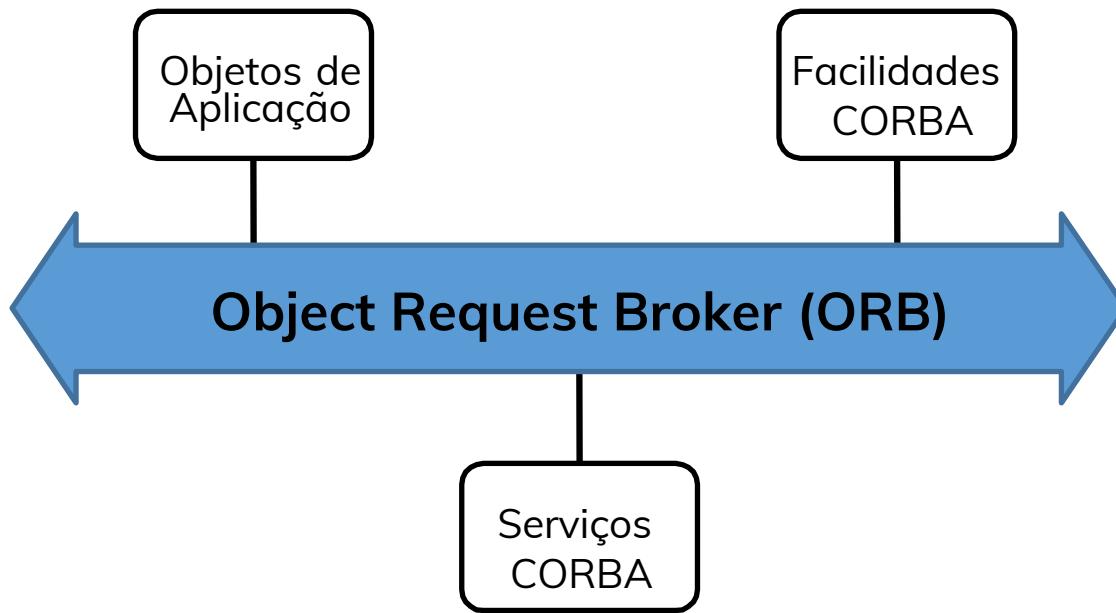
# Corba

- Common Object Request Broker Architecture
  - Objetivo: Padrão para desenvolvimento de aplicações distribuídas para sistemas heterogêneos usando orientação a objetos
- OMG: Object Management Group (<http://www.omg.org>)
  - Consórcio de empresas responsável pela proposição e manutenção do padrão CORBA, criado em 1989

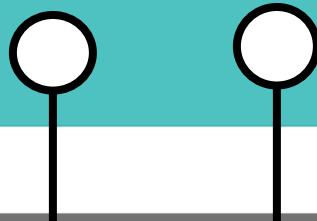
# Object Management Architecture

- Visão do OMG de como deve ser a arquitetura básica de um sistema distribuído
- Dois modelos:
  - Modelo de Objetos
    - Define conceitos tradicionais de orientação a objetos (objetos, herança, interface etc)
  - Modelo de Referência
    - Relaciona serviços de um sistema distribuído passíveis de serem padronizados e, posteriormente, implementados por diferentes empresas

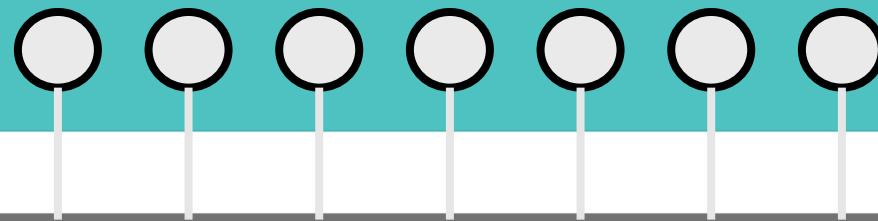
# Arquitetura OMA



## Objetos da Aplicação



## Common Facilities



## ORB – *Object Request Broker*

Nomes persistência concorrência segurança trader transações eventos tempo



## Common Object Services

# Taxonomia de Middleware

<i>Principais categorias</i>	<i>Subcategoria</i>	<i>Exemplos de sistemas</i>
<i>Objetos distribuídos (Capítulos 5, 8)</i>	Padrão	RM-ODP
	Plataforma	CORBA
	Plataforma	Java RMI
<i>Componentes distribuídos (Capítulo 8)</i>	Componentes leves	Fractal
	Componentes leves	OpenCOM
	Servidores de aplicação	SUN EJB
	Servidores de aplicação	CORBA Component Model
	Servidores de aplicação	JBoss

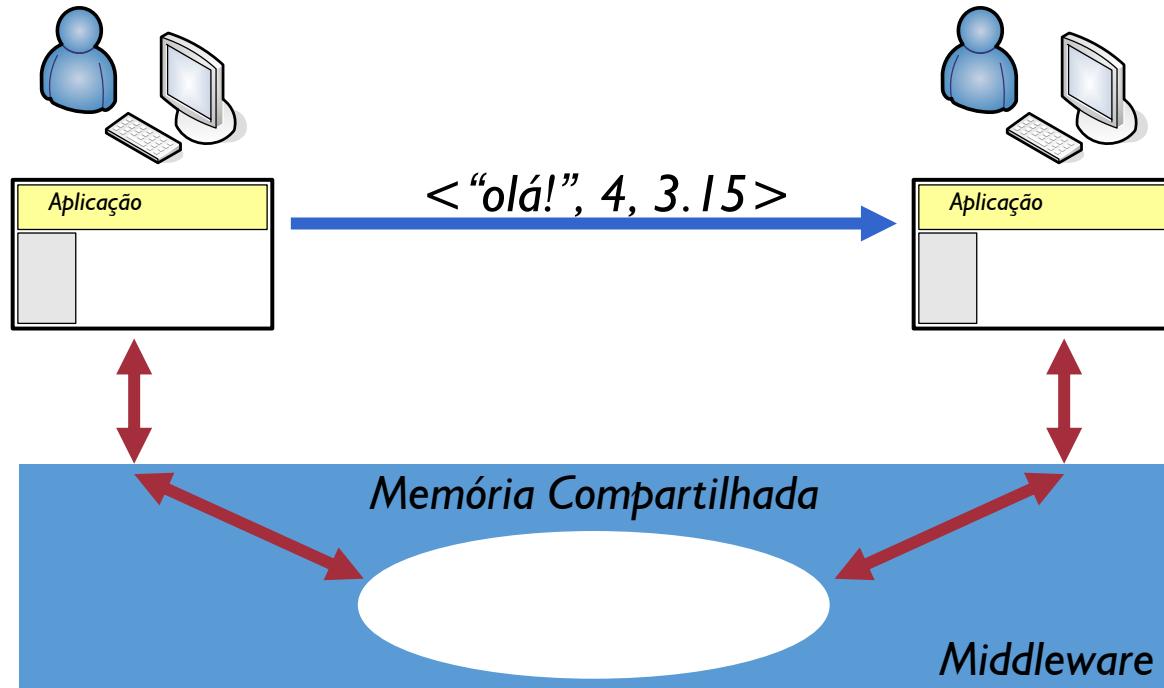
# Taxonomia de Middleware

<i>Principais categorias</i>	<i>Subcategoria</i>	<i>Exemplos de sistemas</i>
<i>Sistemas publicar-assinar (Capítulo 6)</i>	—	CORBA Event Service
	—	Scribe
	—	JMS
<i>Filas de mensagem (Capítulo 6)</i>	—	Websphere MQ
	—	JMS
<i>Serviços web (Capítulo 9)</i>	Serviços web	Apache Axis
	Serviços de grade	The Globus Toolkit
<i>Peer-to-peer (Capítulo 10)</i>	Sobreposições de roteamento	Pastry

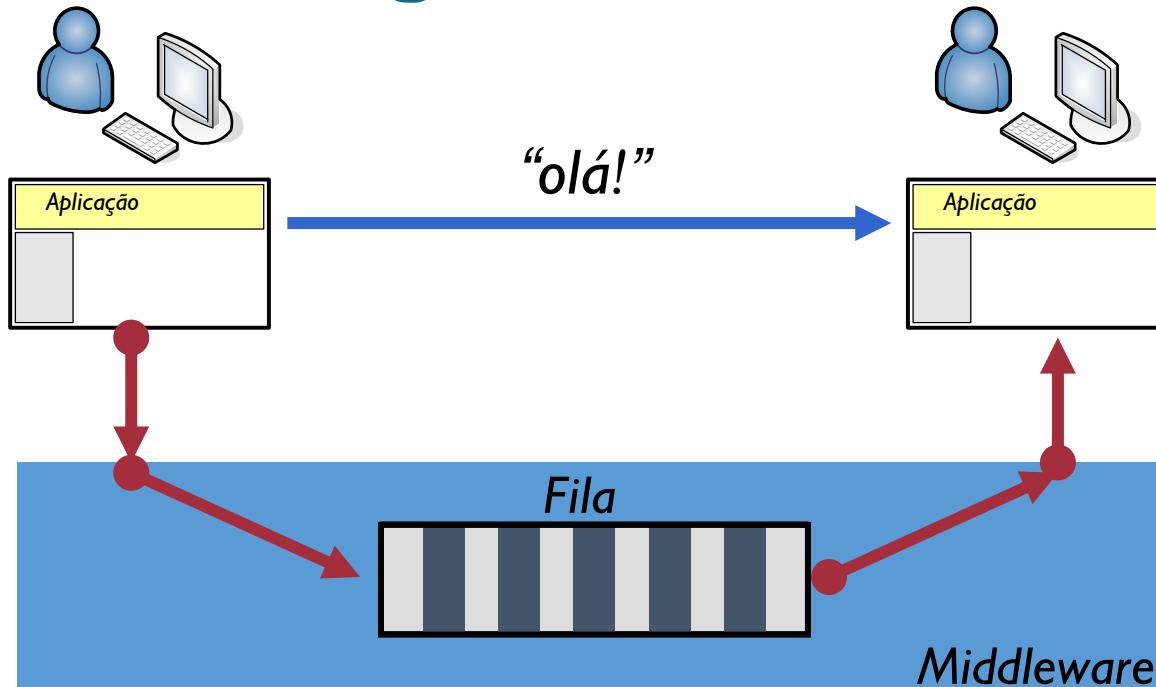
# Modelos de Middleware

- Middleware orientados a transação
- Middleware com memória compartilhada
- Middleware orientados a mensagem
- Middleware orientados a RPC
- Middleware orientados a Objetos
- Middleware Adaptativo
- Middleware Multimídia

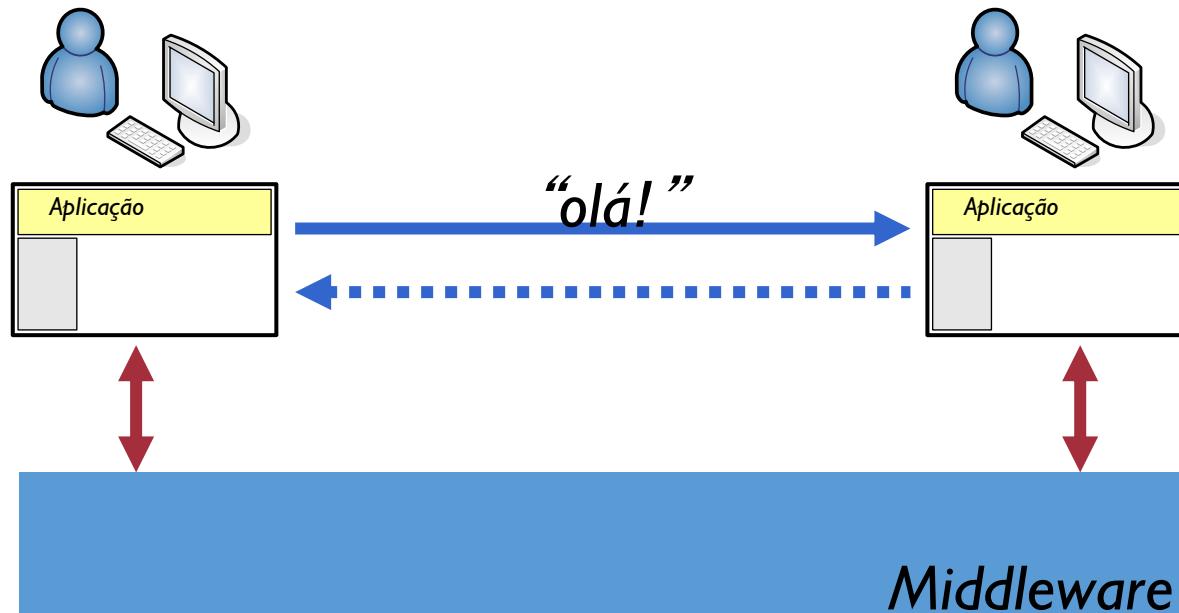
# Memória Compartilhada



# Orientado a Mensagem



# RPC/Objetos



# Middleware Tradicional

- Middleware mascara heterogeneidade de sistemas distribuídos
  - Java VM: SO
  - CORBA: Linguagem
- Trata de concorrência, portabilidade, interoperabilidade
- API única, protocolo comum, conjunto de serviços
- Middlewares tradicionais facilitam a vida do desenvolvedor pela abstração de transparência

# Abordagem Middleware Tradicionais

- Metáfora da Caixa preta
  - Inerente Complexidade de distribuição, localização e heterogeneidade é escondida do desenvolvedor
  - Palavra-chave: Transparência ...
- Para alcançar este requisitos algumas limitações são impostas

# Limitações de Middleware Tradicionais

- Modelo de concorrência fixo
  - Estática ou dinâmica, aninhada ou plana, etc..
- Protocolo de transporte fixo
  - IIOP, TCP/IP
- Escalonamento Fixo
- Formatação de dados fixa
- Arquitetura monolítica
  - Tudo ou nada (Sistemas enormes)

# Limitações de Middleware Tradicionais

- Projetados (e usados) com sucesso em sistemas distribuídos estacionários, construídos com redes cabeadas fixas
- Porém, não adequadas a aplicações emergentes, como
  - Sistemas Altamente Dinâmicos
    - Computação móvel, ubíqua
  - Sistemas Especializados
    - Sistemas embarcados

# Dilema

- Sistemas Distribuídos Tradicionais
  - Boa parte das aplicações necessita que o middleware esconda detalhes de suas camadas inferiores
- Sistemas Distribuídos não tradicionais
  - Certas aplicações podem ter ganhos ao observar e manipular detalhes das camadas inferiores



*Transparência*



*Conhecimento (Awareness)*

FABIO KON, FABIO COSTA,  
GORDON BLAIR, AND ROY H. CAMPBELL

# The Case for REFLECTIVE Middleware

IT'S FLEXIBLE AND RECONFIGURABLE YET SIMPLE  
FOR PROGRAMMERS TO USE, NOTABLY FOR BUILDING  
DYNAMIC DISTRIBUTED APPLICATIONS OPERATING  
ON THE NET.

**R**ecent advances in distributed, mobile, and ubiquitous systems demand new comput- grammer the complicated details of network communication, remote method invocation, naming,

Fabio Kon, Fabio Costa,  
Gordon Blair, and Roy H.  
Campbell. 2002. The case for  
reflective middleware.  
Commun. ACM 45, 6 (June  
2002), 33-38. DOI:  
<https://doi.org/10.1145/508448.508470>

# Outras categorias

- Middleware para Computação Móvel
- Middleware para Computação Ubíqua
- Middleware para Jogos
- Middleware para Internet das Coisas
- Middleware para redes de sensores sem fio

# Outras categorias

- Middleware para Computação Móvel
- Middleware para Computação Ubíqua
- Middleware para Jogos
- Middleware para Internet das Coisas
- Middleware para redes de sensores sem fio
  
- TAREFA: Busque na Web por exemplos de middleware para diferentes domínios e relate quais os serviços propostos para cada um deles