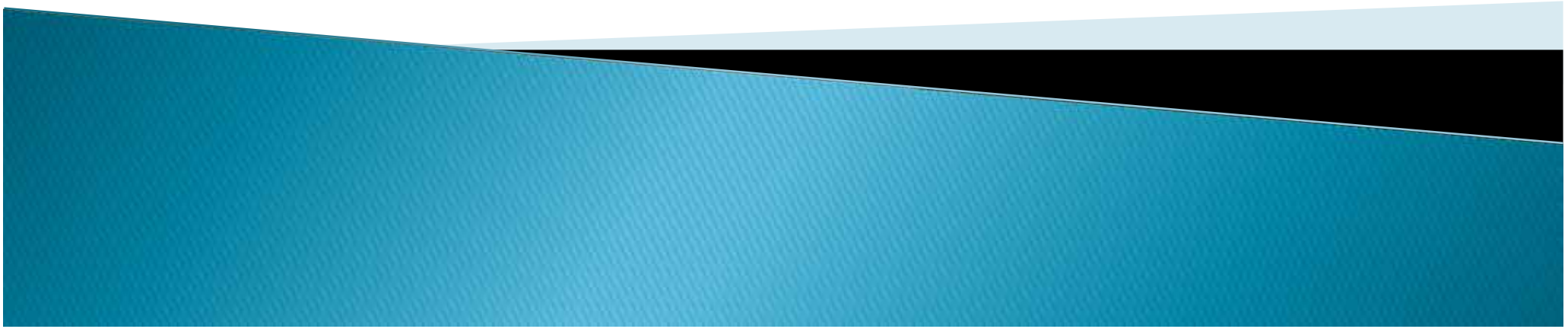
Sistemas Distribuídos

Modelos Arquiteturais

Prof. Tales Viegas

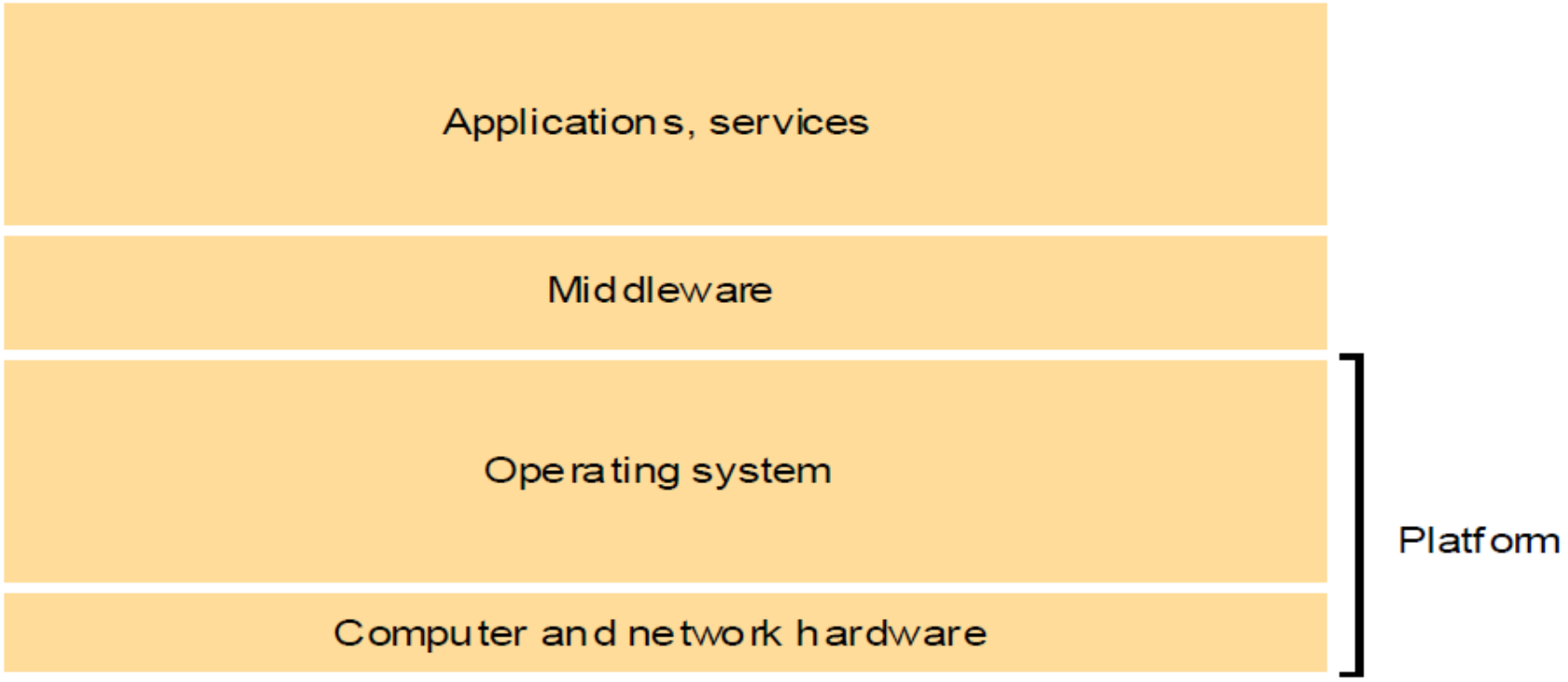


Modelos Arquiteturais

- Um modelo arquitetural de um sistemas distribuído é a estrutura de um sistema em termos de:
  - L**ocalização** das suas diferentes partes
  - **Papel** que cada parte desempenha
  - Como elas se **relacionam**

- A arquitetura tem implicações no desempenho, confiabilidade e segurança do sistema

Camadas de um Sistema Distribuído



Exemplos de Plataformas

- Intel PII/Windows
- Intel x86/Linux
- PowerPC/Solaris
- ...

Middleware

- Camada de software que tem o objetivo de abstrair a heterogeneidade de um sistema distribuído, fornecendo um modelo de programação uniforme

- Ex:
  - Sun RPC
  - Java RMI
  - Corba
  - Microsoft .Net

Arquitetura Cliente-Servidor

- Modelo independente do Middleware utilizado

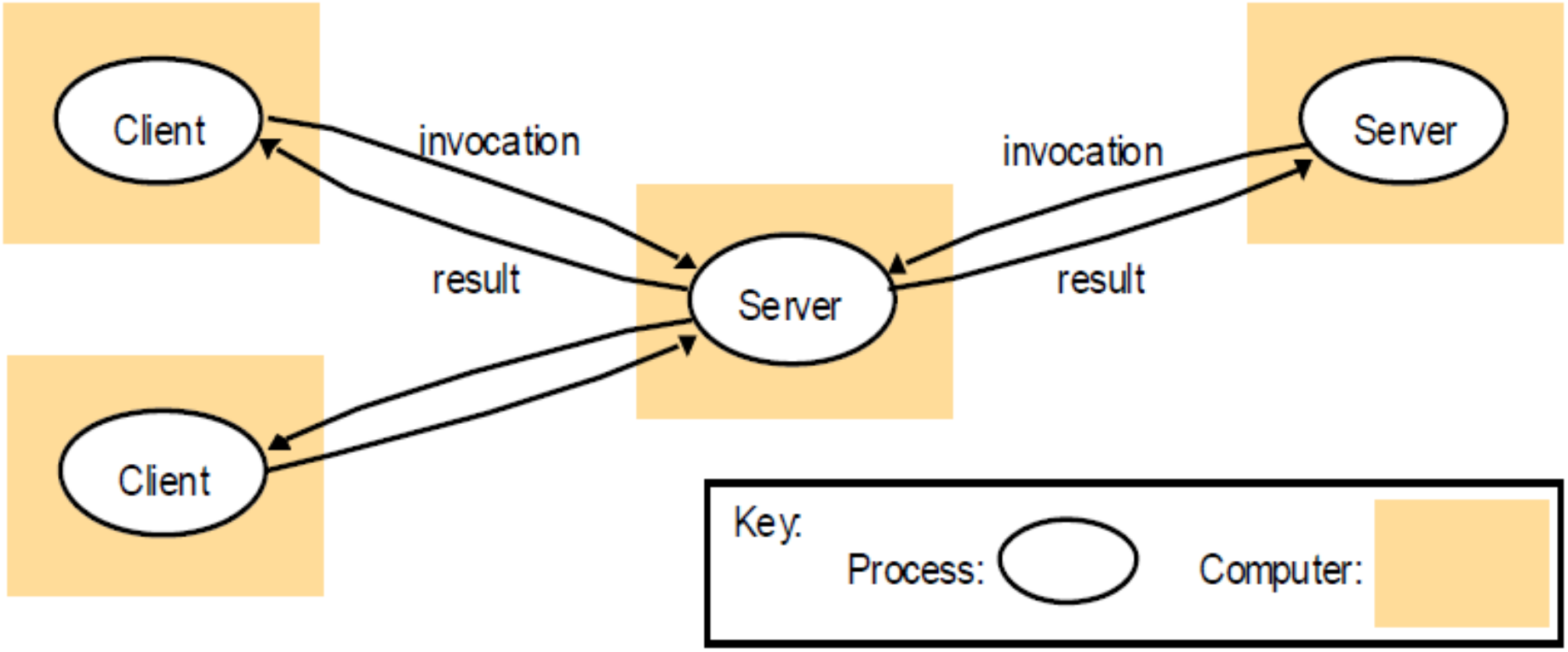
- **Servidor** (*back-end*)
  - Processo passivo que, quando contatado por um cliente, envia a resposta
- **Cliente** (*front-end*)
  - Contata o servidor com o objetivo de utilizar um serviço. Envia um pedido (request/invocation) e fica a espera da resposta (*reply/response*)

Arquitetura Cliente-Servidor

- Cliente e servidor são papeis que são desempenhados em um determinado momento

- Uma entidade pode simultaneamente ser cliente e servidor, pode ter que recorrer a um outro serviço, sendo cliente deste.

Arquitetura Cliente-Servidor



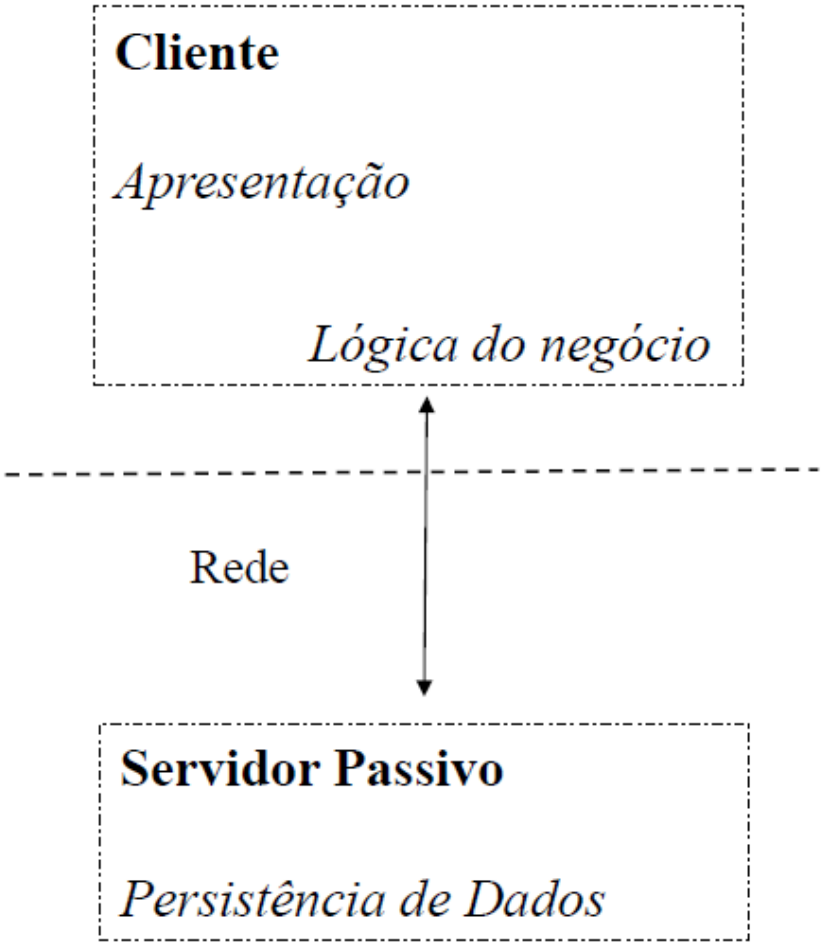
Arquitetura Cliente-Servidor

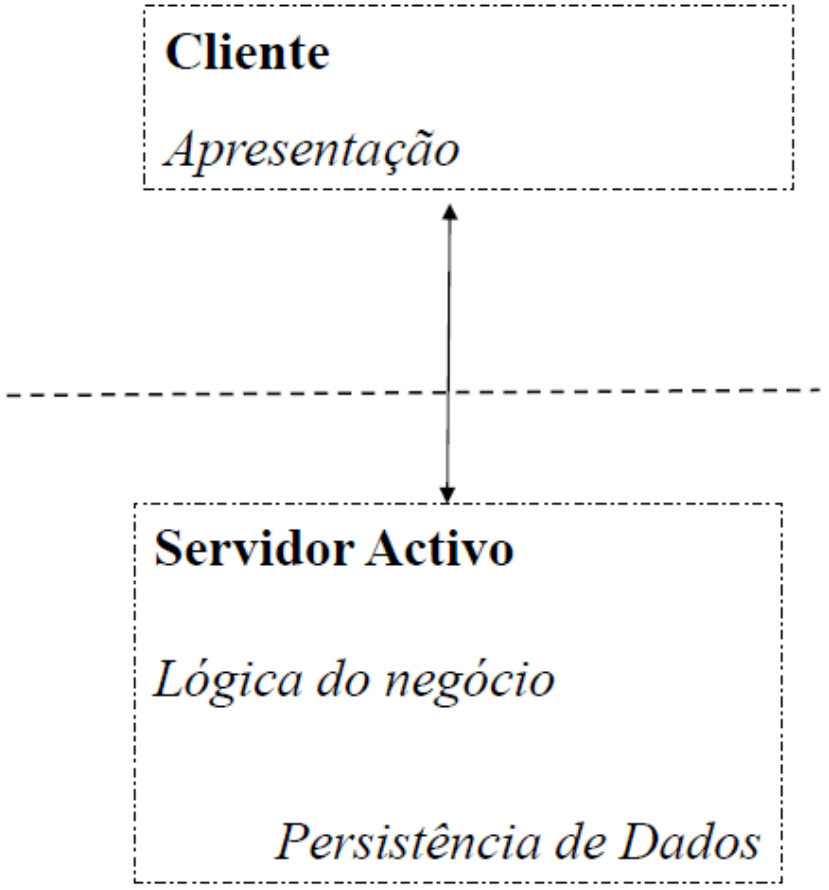
- A máquina que hospeda o processo servidor deve ter recursos suficientes a fim de suportar a quantidade máxima de requisições simultâneas esperadas no processo

Arquitetura Cliente-Servidor

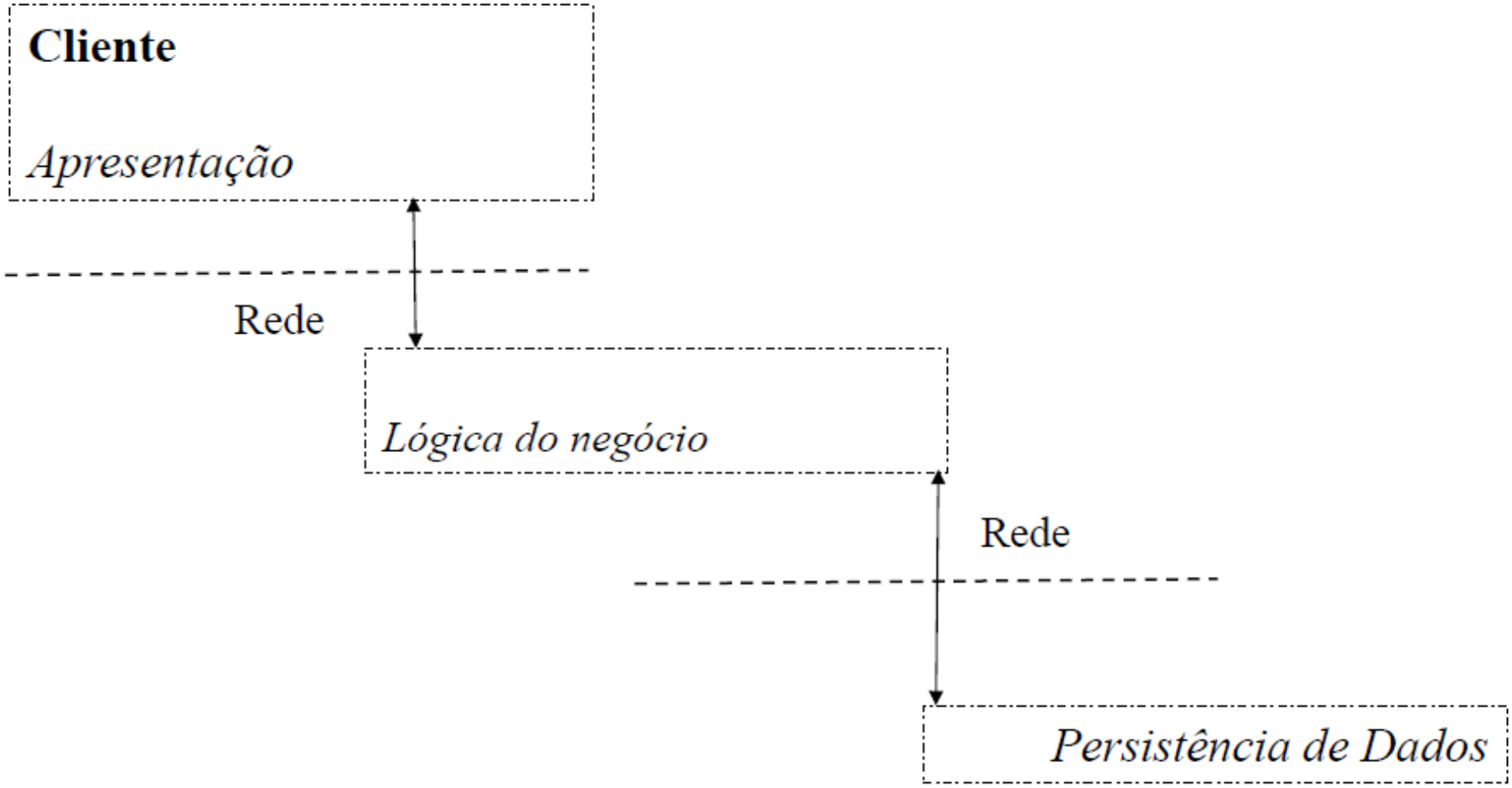
- Em um sistema de informação típico, existem três classes de funcionalidades:
  - Camada de Apresentação
    - Responsável pela interface com o usuário
  - Camada de Lógica de Negócio
    - Regras de negócio que controlam o comportamento da aplicação
  - Camada de Persistência de Dados
    - Parte que assegura o armazenamento e integridade dos dados

Modelo de 2 camadas (2-tiers)

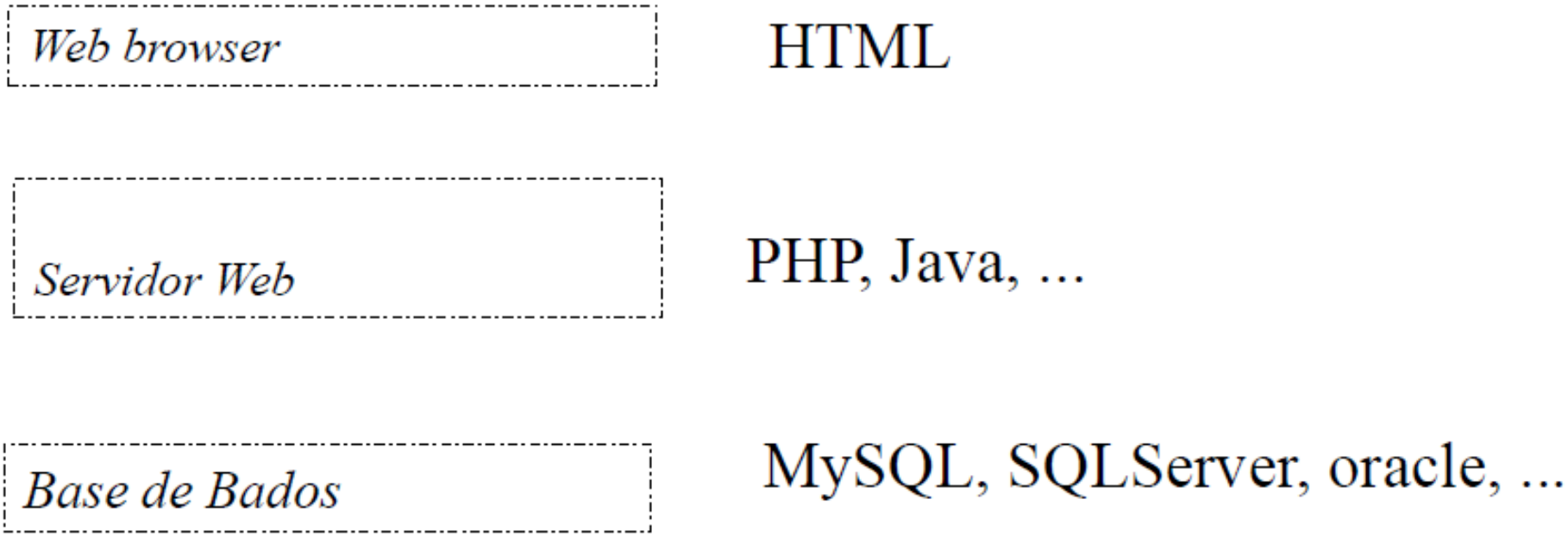




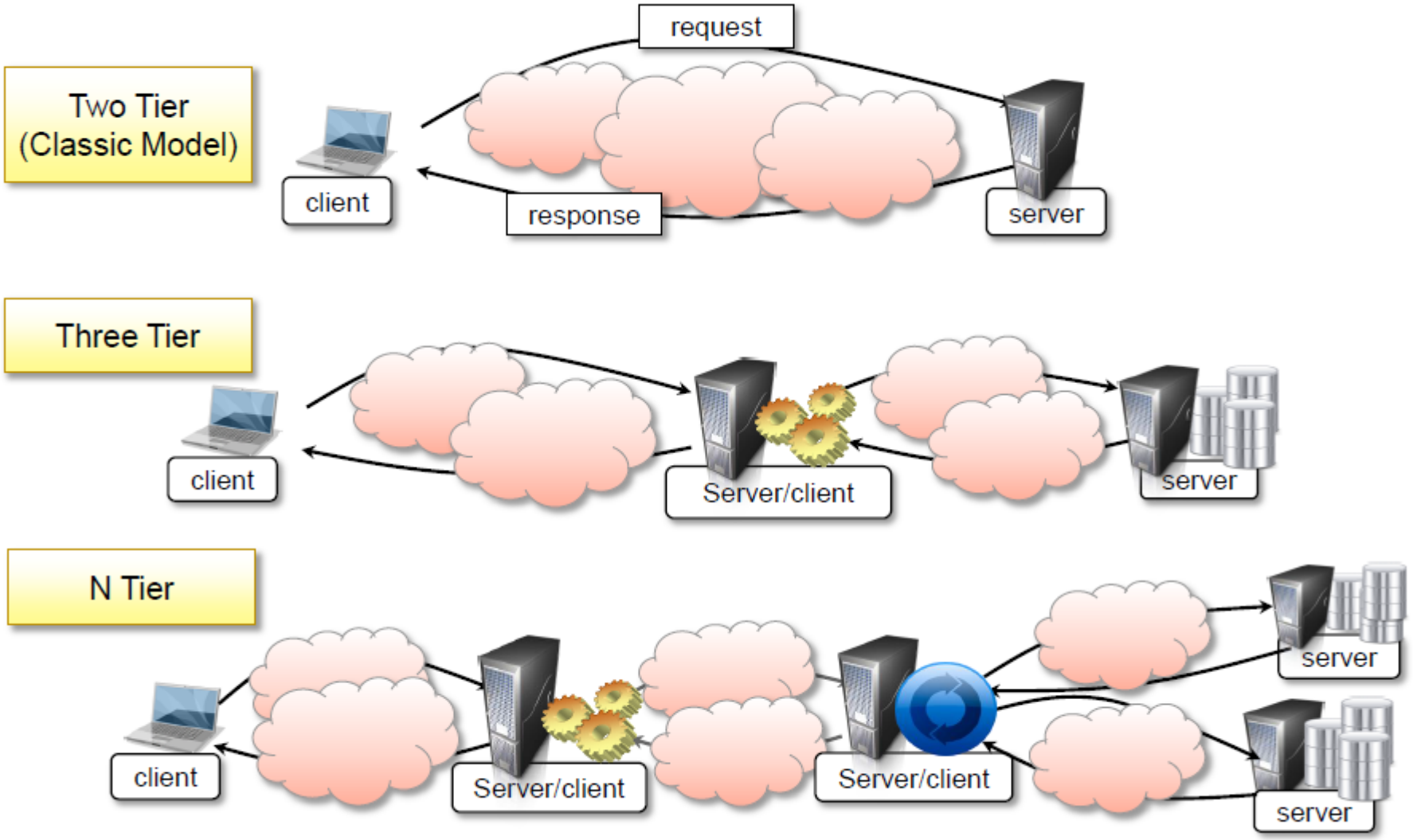
Modelo de 3 camadas (3-tiers)



Modelo de 3 camadas – Exemplo



Modelos Arquiteturais



Modelo Cliente-Servidor

- Modelo mais usado na prática
- Modelo de interação simples
- Segurança concentrada no servidor
- Servidor é um ponto de falha único
- Não é escalável para além de certos limites

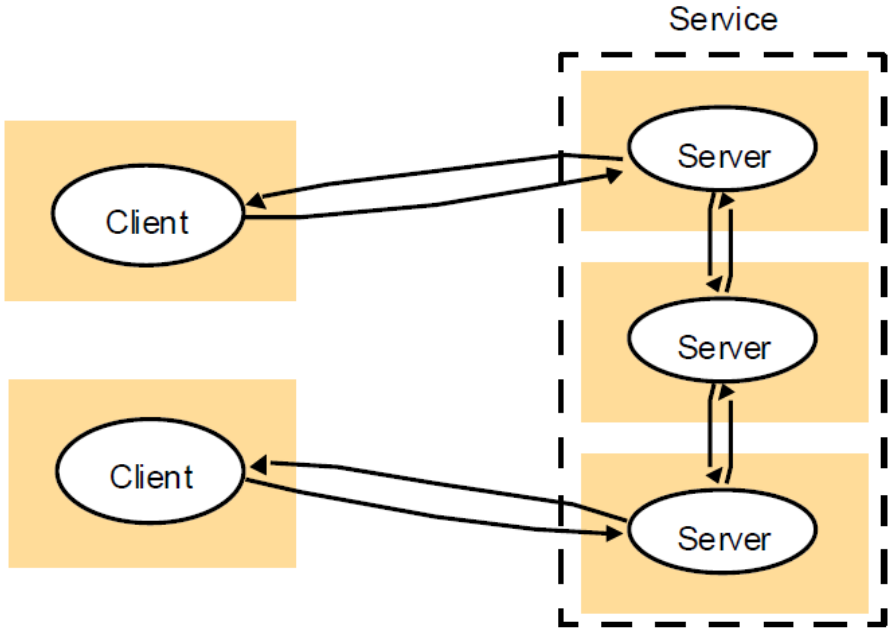
Modelo Cliente-Servidor

- Exemplos de grandes sistemas distribuídos com arquitetura cliente-servidor:
  - Google Search, Facebook, YouTube,...
- Podem aparecer alguns problemas como: particionamento, replicação, hierarquia, etc.

Múltiplos Servidores

- Um único serviço pode ser implementado por vários processos servidores, localizados em diferentes computadores

- Permite maior disponibilidade e tolerância a falhas



Cliente/Servidor Replicado

- Existem vários servidores capazes de responder aos mesmos serviços

- Vantagens:
  - Permite distribuir melhor a carga, melhorando o desempenho
  - Não existe um ponto de falha único

- Principal problema:
  - Manter o estado do servidor coerente em todas as réplicas

Cliente/Servidor Particionado

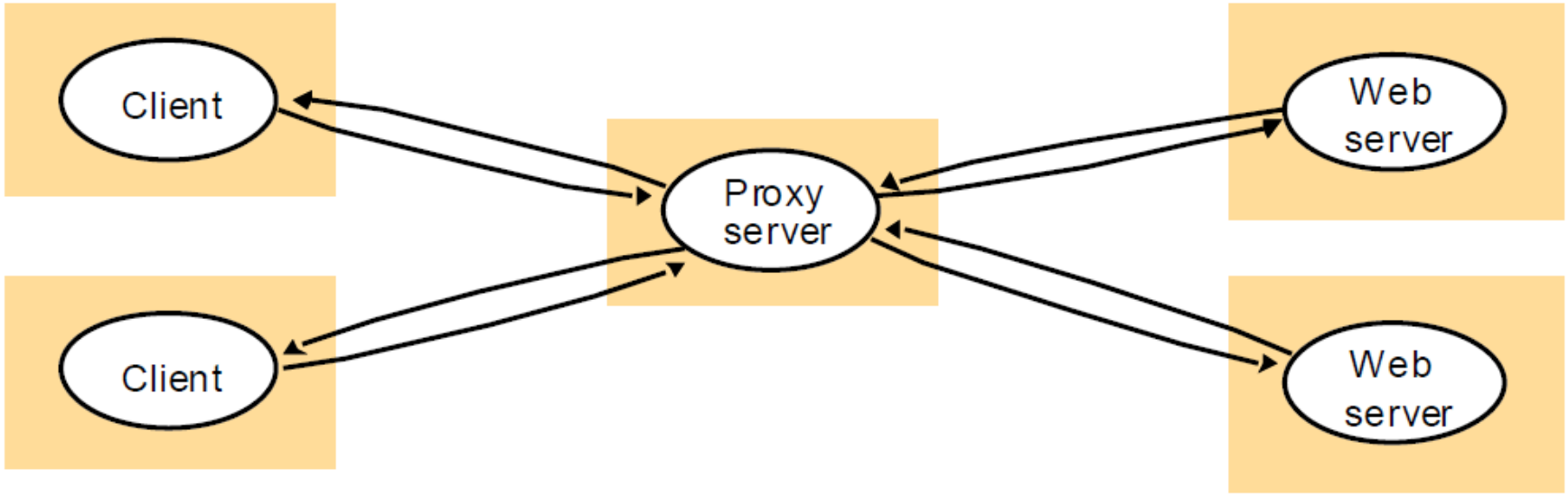
- Existem vários servidores com a mesma interface, cada um capaz de responder a uma parte dos pedidos (ex: DNS)
- Servidor redireciona o cliente para outro servidor (**iterativo**)
- Servidor invoca o pedido em um outro servidor (**recursivo**)

Cliente/Servidor Particionado

- Vantagens
  - Escalabilidade
  - Permite distribuir a carga, melhorando o desempenho
  - Não possui um ponto de falha único
- Problema
  - Falha de um servidor impede acesso aos dados presentes nesse servidor

Proxy Servers e Caches

- Um cache permite o armazenamento, em uma localização mais próxima, de dados/objetos recentemente usados

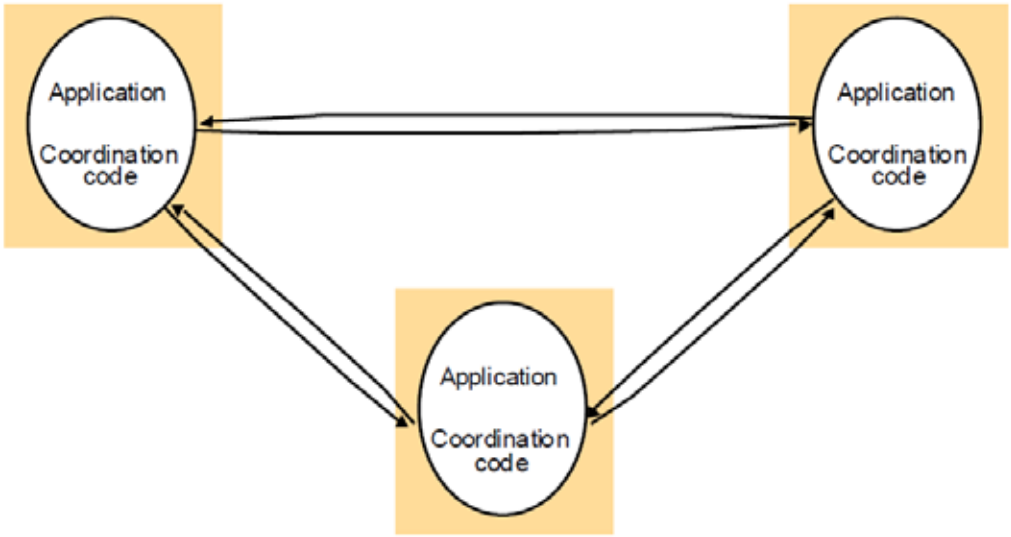


Proxy Servers e Caches

- Quando um cliente necessita de um objeto, o serviço de “caching” verifica se possui uma cópia atualizada do objeto, em caso afirmativo fornece esta cópia
- Um “cache” pode estar localizado no cliente ou em servidores “proxy”, que são utilizados por diversos clientes
- Objetivo: aumentar a disponibilidade e a performance do serviço

Processos Pares (peer processes)

- Todos os processos desempenham papeis similares.
- Cada processo é responsável pela consistência dos seus dados (recursos) e pela sincronização das várias operações



Processos Pares

- Cada processo pode assumir (simultaneamente ou alternadamente) o papel de cliente e servidor do mesmo serviço

- Paradigma de distribuição em que os serviços são suportados diretamente pelos seus clientes/usuários, sem recurso a uma infraestrutura criada e mantida explicitamente para este fim

Processos Pares

- A ideia base é conseguir explorar os recursos disponíveis nas máquinas ligadas em rede: CPU, disco, largura de banda, etc.
- Modelos de iteração e coordenação mais complexos (que sistemas cliente/servidor)
- Algoritmos mais complexos
- Não existe ponto único de falha

Processos Pares

- Grande potencial de escalabilidade
- Apropriado para ambientes em que todos os participantes querem cooperar para fornecer um dado serviço

Processos Pares – Exemplos

- Napster/Kazaa
- GNUtella
- Torrent
- BitTorrent
- Skype

Processos Pares

- Duas arquiteturas principais

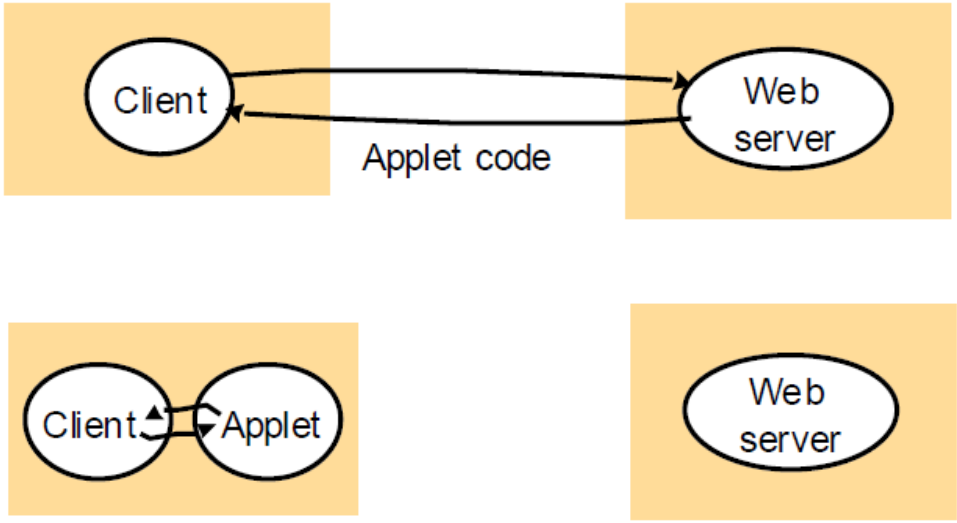
- Um servidor de diretório centralizado (ex: Napster)
  - Quando um novo processo se liga, informa qual o servidor central (do seu endereço, conteúdo)
  - A partir daí, pode se comunicar com outros pares

Processos Pares

- Serviço de Diretório Distribuído (ex: Kazaa, Torrent)
  - Quando um novo processo se liga, liga-se a um grupo
  - O líder do grupo registra o conteúdo de todos os elementos do grupo
  - Cada processo acessa o seu líder para localizar o que pretende
  - Cada líder pode acessar outros líderes

Web Applets

- Arquitetura onde um cliente realiza a instalação de parte de um serviço e, a partir deste, realiza as operações



Agentes Móveis

- Um agente é um programa executável que pode mover-se de uma máquina para outra

- Age em nome de um usuário específico e, no computador para o qual se transfere, realiza algum serviço para seu proprietário, podendo obter informações que mais tarde transmitirá ao local de origem

Agentes Móveis

- Entidade capaz de interagir autonomamente com o ambiente com o qual se rodeia, podendo apresentar características de adaptabilidade, mobilidade, cooperação ou competição

- Problemas de segurança

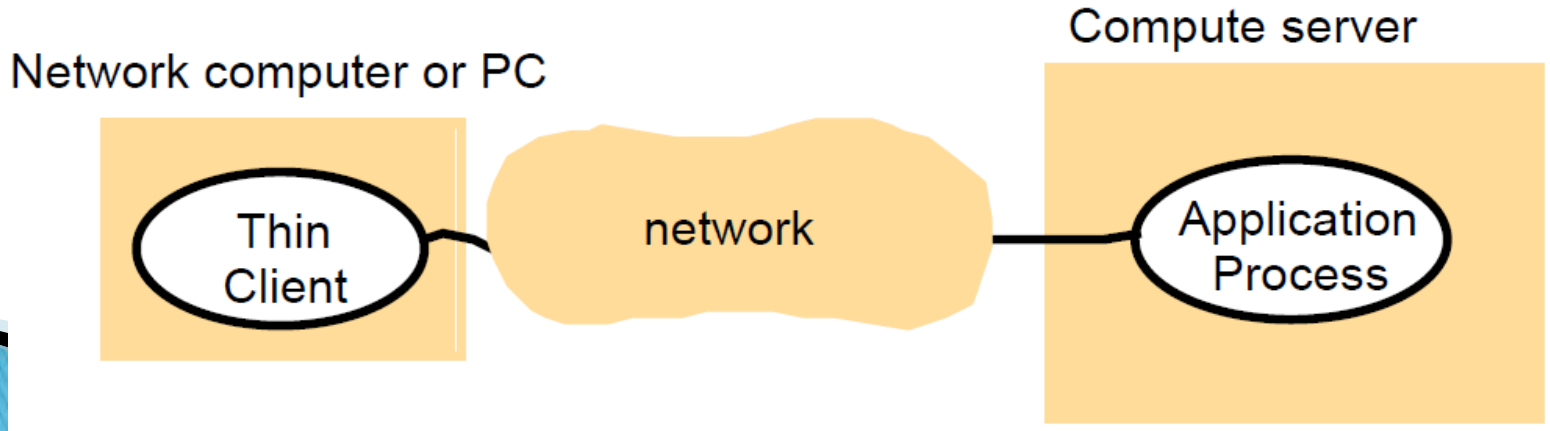
- Dificuldades de realizar o trabalho devido a problemas impossíveis de prever

Network Computers

- Computadores sem disco nem periféricos, contando apenas com o apoio da rede para fornecer os serviços para o usuário
- Aplicações **executam localmente**, mas os arquivos são gestionados por um servidor remoto
- Solução econômica para centros de computação com poucos recursos

Thin Clientes

- Interface gráfica, baseada em Windows, na máquina local do usuário
- As aplicações executam no servidor
- Ex: Citrix WinFrame



Equipamentos Móveis e Redes Espontâneas

- Laptops, PDAs, celulares, câmeras digitais, máquinas de lavar, relógios, etc
- Protocolos Wireless: BlueTooth, Infravermelho, HomeRF, RFID

- Principais características das redes sem-fio
  - Configuração é feita automaticamente sem intervenção humana
  - Os equipamentos digitais móveis descobrem por si os serviços disponíveis

Equipamentos Móveis e Redes Espontâneas

- Problemas
  - Conexão limitada (se os dispositivos se afastam demais do local de transmissão)
  - Segurança e privacidade

Redes Espontâneas

- Exigem um meio dos clientes (equipamentos móveis) descobrirem quais serviços estão disponíveis na rede a qual se conectam

- Um “discovery service” é um servidor (ou um ou mais processos) que mantém uma lista dos tipos e características dos serviços disponíveis dentro da rede local sem-fio

Discovery Service

- Oferecem 2 tipos de serviços:
  - Registro de serviços
    - Aceita pedidos para registrar em uma base de dados os detalhes de cada serviço disponível
  - Lookup de serviços
    - Aceita queries aos serviços disponíveis, fornecendo detalhes suficientes para que o cliente possa se conectar ao serviço que escolher