Gerenciamento de Dados em Sistemas P2P

XXI SBBD – Florianópolis

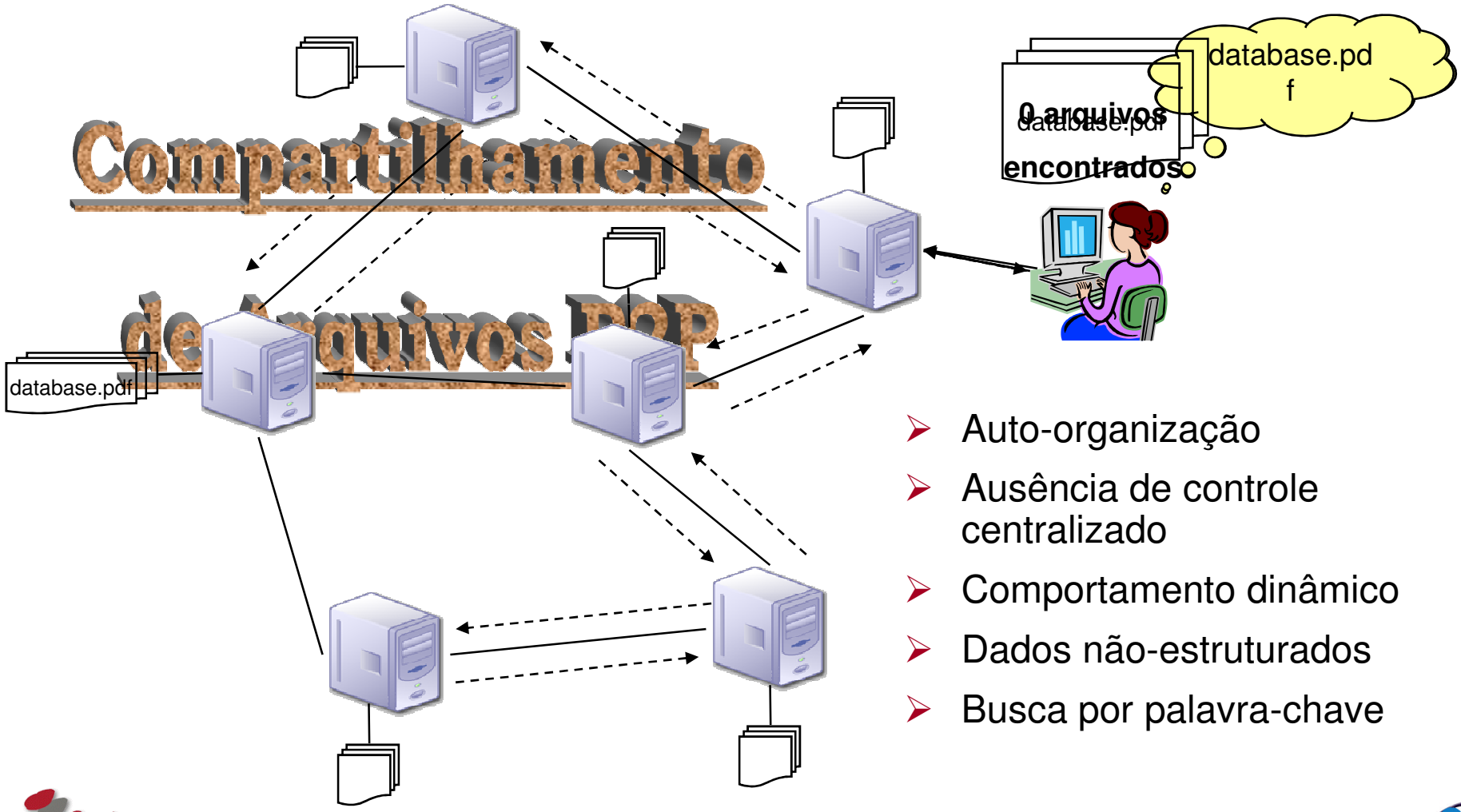
16 a 18 de Outubro de 2006

Ana Carolina Salgado,
Carlos Eduardo Santos Pires,
e Bernadette Farias Lóscio

{acs,cesp}@cin.ufpe.br,

bernafarias@lia.ufc.br

Cenário Ponto-a-Ponto



- Auto-organização
- Ausência de controle centralizado
- Comportamento dinâmico
- Dados não-estruturados
- Busca por palavra-chave

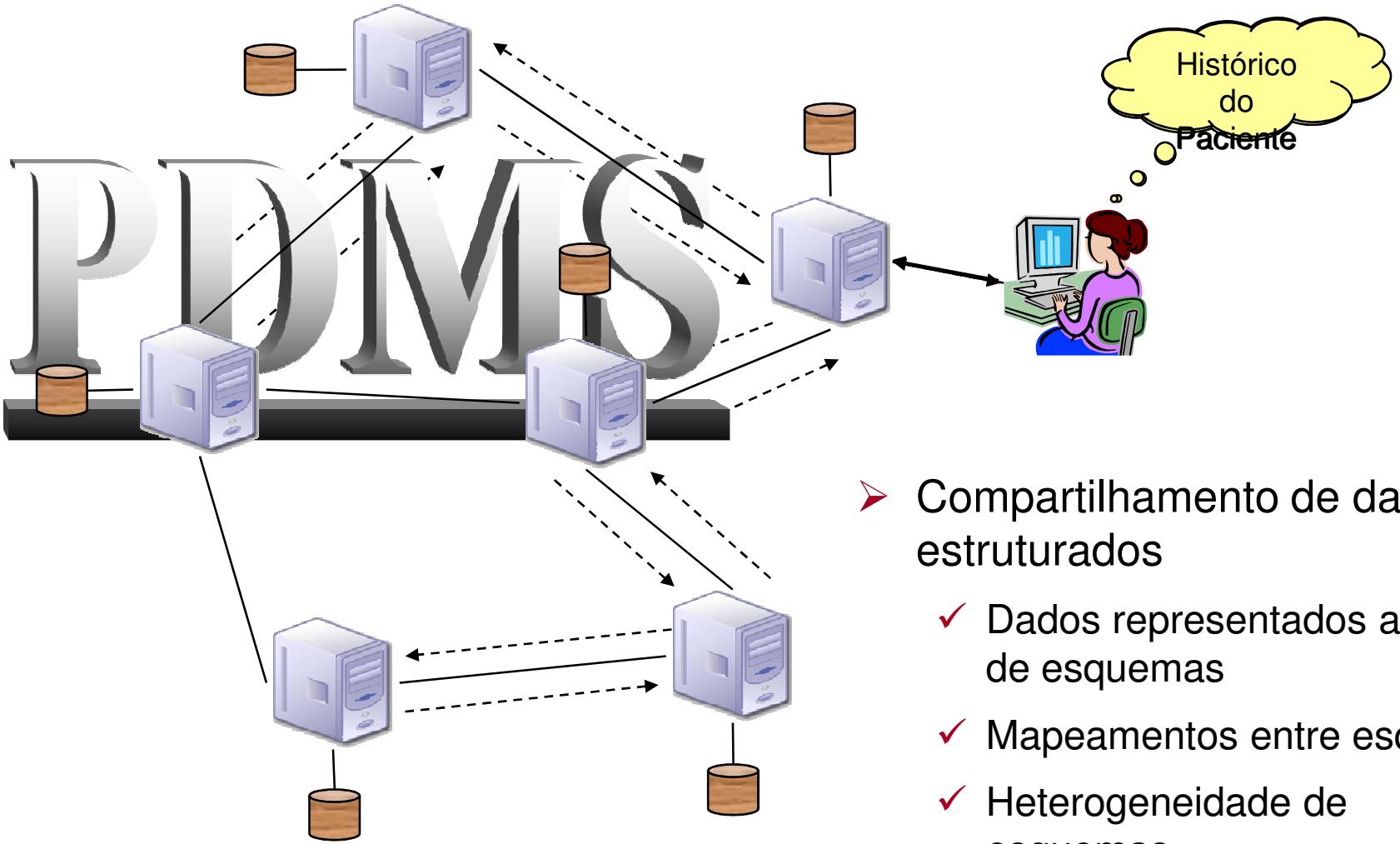
Paradigma Ponto-a-Ponto

Compartilhamento de
serviços e
recursos computacionais
diretamente entre sistemas

P2P – Questões Importantes

- Como reconhecer um outro ponto na rede?
- Como publicar informações para os demais pontos?
- Como identificar unicamente um ponto?
- Como compartilhar dados na rede?
- Como organizar os dados compartilhados?

Cenário PDMS (*Peer Data Management System*)



- Compartilhamento de dados estruturados
  - ✓ Dados representados através de esquemas
  - ✓ Mapeamentos entre esquemas
  - ✓ Heterogeneidade de esquemas

Novos Desafios

- Localização dos Dados
  - ✓ Pontos devem saber localizar dados em outros pontos
- Processamento de Consultas
  - ✓ Pontos devem localizar os dados e executar a consulta eficientemente
- Integração de Dados
  - ✓ A heterogeneidade da representação dos dados nos vários pontos deve ser resolvida
- Consistência dos Dados
  - ✓ A consistência deve ser mantida em caso de replicação e uso de *cache*

Agenda

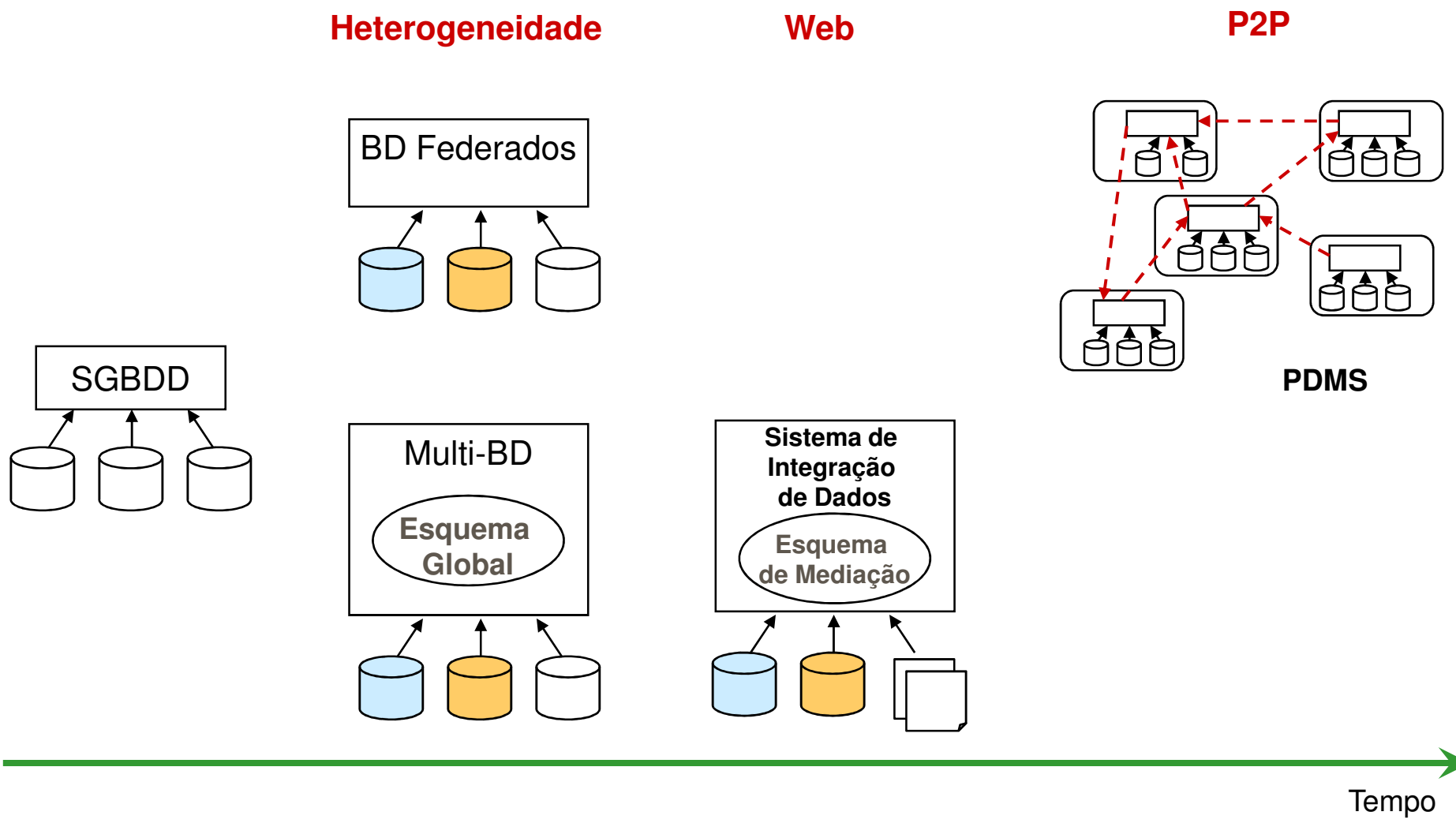
- Bancos de Dados X Sistemas P2P
- Paradigma Ponto-a-Ponto (P2P)
  - ✓ Características
  - ✓ Topologias de Rede
- P2P x Grades Computacionais
- PDMS – *Peer Data Management System*
  - ✓ Gerenciamento de Dados em PDMS
  - ✓ Principais PDMS
- Desafios

Bancos de Dados

X

Sistemas P2P

Evolução dos Sistemas de Gerenciamento de Dados Distribuídos



Bancos de Dados Distribuídos

- BD Distribuído

  É uma coleção de múltiplos BD logicamente inter-relacionados distribuídos através de uma rede de computadores

- Sistema de BD Distribuídos

  É definido como um sistema que permite o gerenciamento de BD distribuídos deixando a distribuição dos dados transparente ao usuário

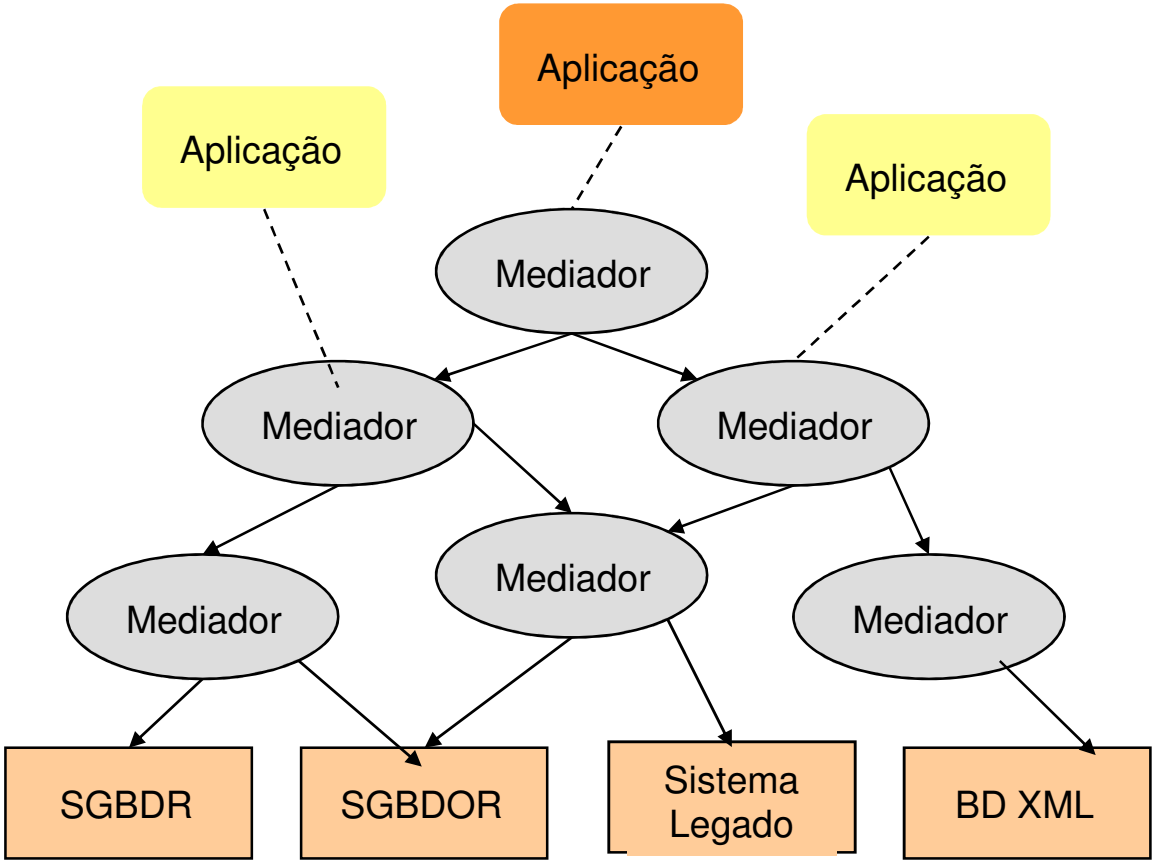
Evolução dos BD Distribuídos

- Bancos de Dados Distribuídos
  - ✓ Dados homogêneos
  - ✓ Acesso e atualização
- Multi-Bancos de Dados
  - ✓ Dados heterogêneos
  - ✓ Acesso e atualização usando um esquema global
- Bancos de Dados Federados
  - ✓ Dados heterogêneos
  - ✓ Importação e exportação de esquemas (não há esquema global)

Evolução dos BD Distribuídos

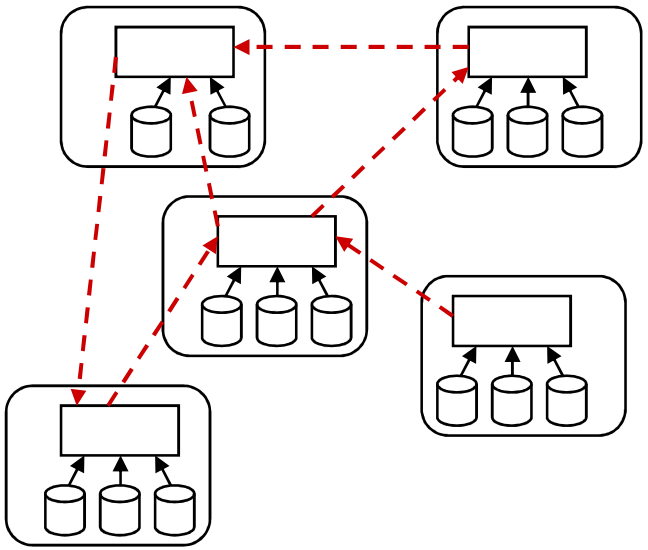
- Sistemas de Integração de Dados
  - ✓ Tipo particular de *middleware*
  - ✓ Integração de fontes de dados distribuídas, autônomas e heterogêneas
  - ✓ Visão lógica unificada de dados (esquema global)
    - Evita lidar com as inúmeras fontes, *interfaces* e representações dos dados diferentes
  - ✓ Apenas consultas aos dados
  - ✓ Abordagens para integração de dados: virtual e materializada
    - Dados atuais x Tempo de resposta

Vários Mediadores



[Wiederhold 92]

PDMS



Sistemas de Gerenciamento de Dados Ponto-a-Ponto (PDMS)

- Compartilhamento de dados descentralizado
- Processamento e armazenamento de dados distribuídos em pontos autônomos
- Comportamento dinâmico dos pontos: podem entrar e sair do sistema a qualquer momento
- Escalabilidade sem servidores poderosos
- Mapeamentos semânticos dos dados armazenados nos pontos

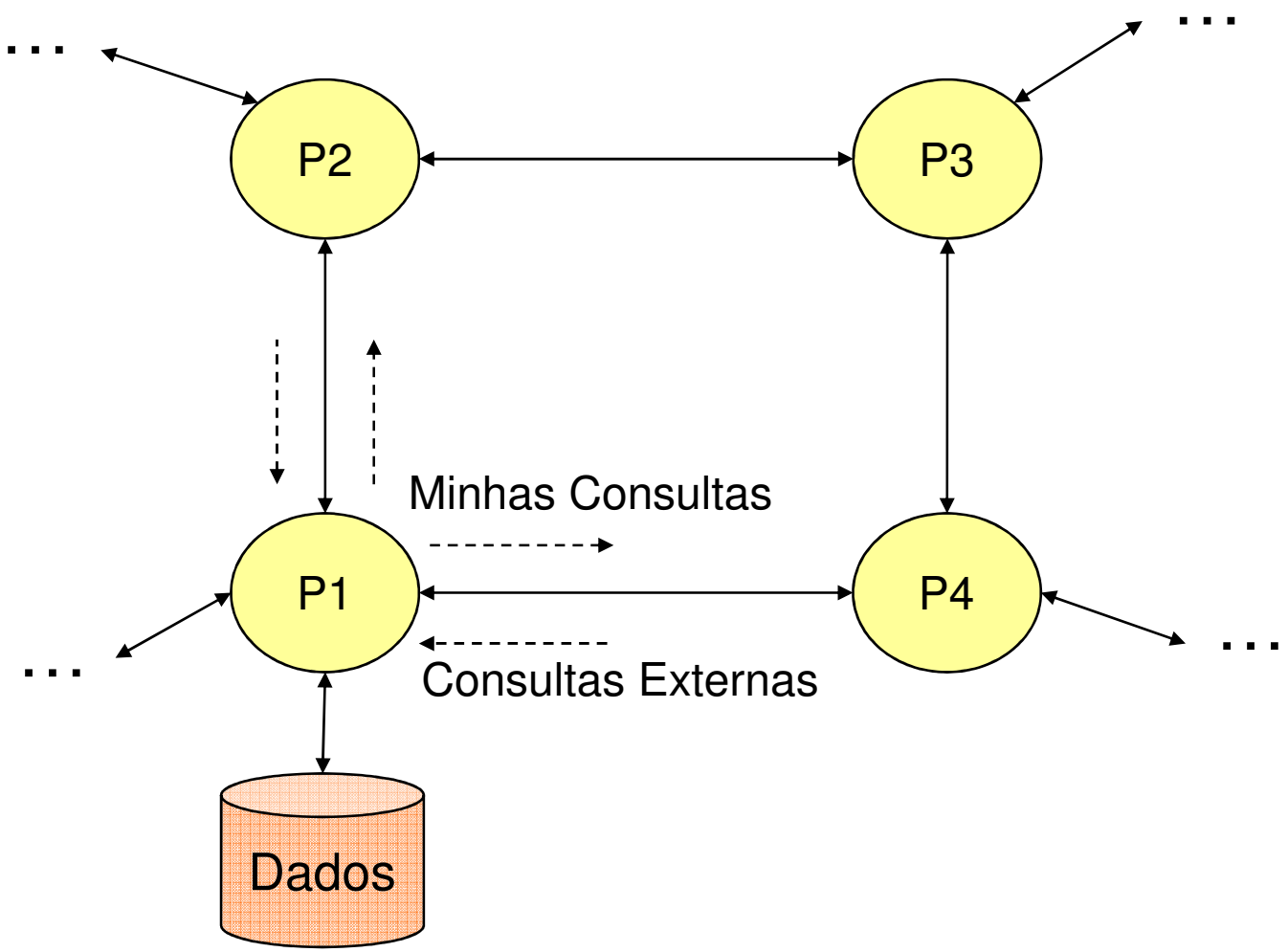
DB Distribuídos X PDMS

- BD Distribuídos
  - ✓ Pequeno número de fontes
  - ✓ Nós controlados na rede
  - ✓ Dados consistentes
    - Coordenação
  - ✓ Dados estruturados (e.g., modelo relacional)
  - ✓ Transações
  - ✓ Restrições de integridade
  - ✓ Consultas complexas
  - ✓ Esquemas criados pelo administrador
  - ✓ Topologia relativamente fixa
- PDMS
  - ✓ Grande número de fontes
  - ✓ Nós entram e saem da rede
  - ✓ Dados não confiáveis
    - Autonomia
  - ✓ Dados semi-estruturados (e.g., XML)
  - ✓ Sem transações
  - ✓ Sem restrições de integridade
  - ✓ Consultas simples
  - ✓ Esquemas criados pelo usuário
  - ✓ Rede imprevisível

Banco de Dados X P2P

- No paradigma P2P falta semântica, transformação de dados e relacionamento entre dados
- BD oferecem consultas, visões e restrições de integridade
- BD possibilitam consultas complexas e reuso de resultados de consultas anteriormente processadas

Banco de Dados P2P



Paradigma Ponto-a-Ponto

Terminologia

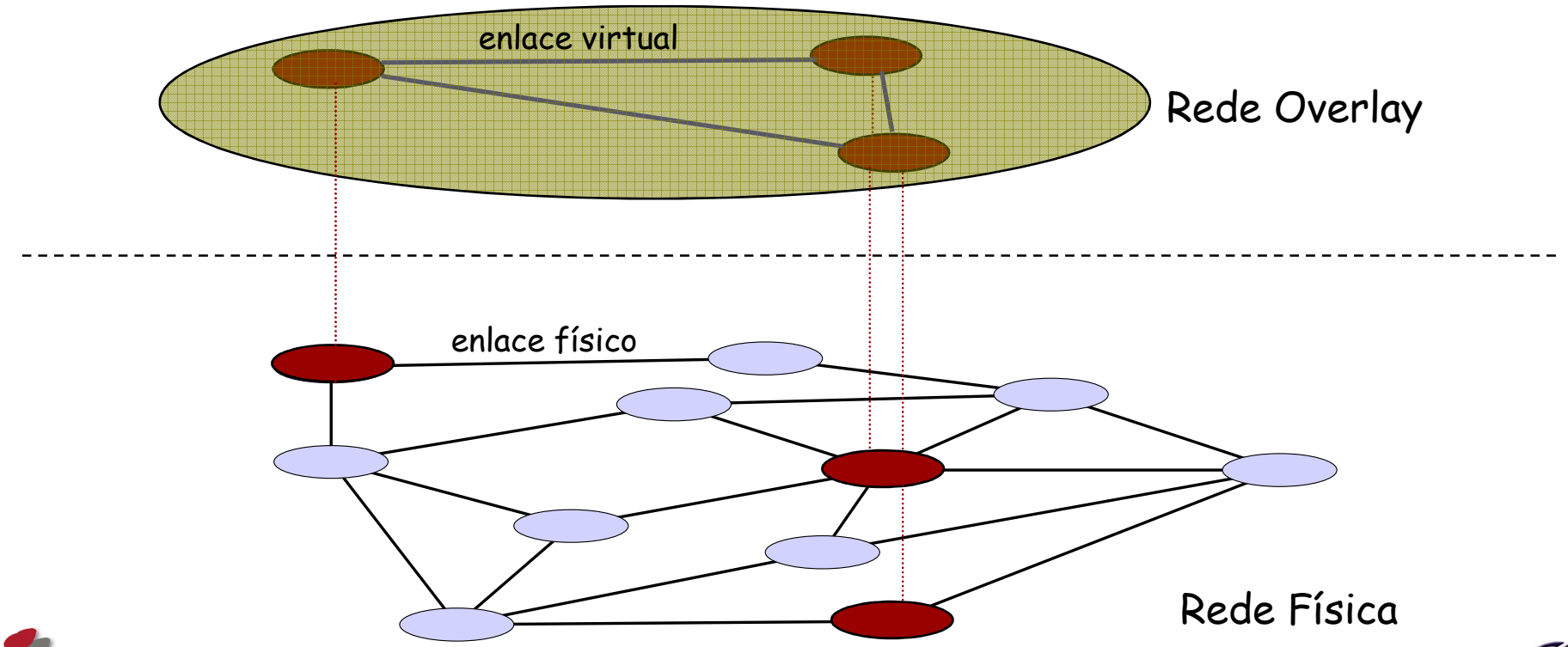
- Peer ≡ Ponto ≡ Nó
  - ✓ Componente de uma rede P2P
  - ✓ Pode assumir o papel de cliente e servidor
- Cluster
  - ✓ Agrupamento de pontos com interesses específicos
  - ✓ Exemplo: cluster semântico
- Topologia de rede e localização dos dados
  - ✓ Estruturada
  - ✓ Não-estruturada

Terminologia

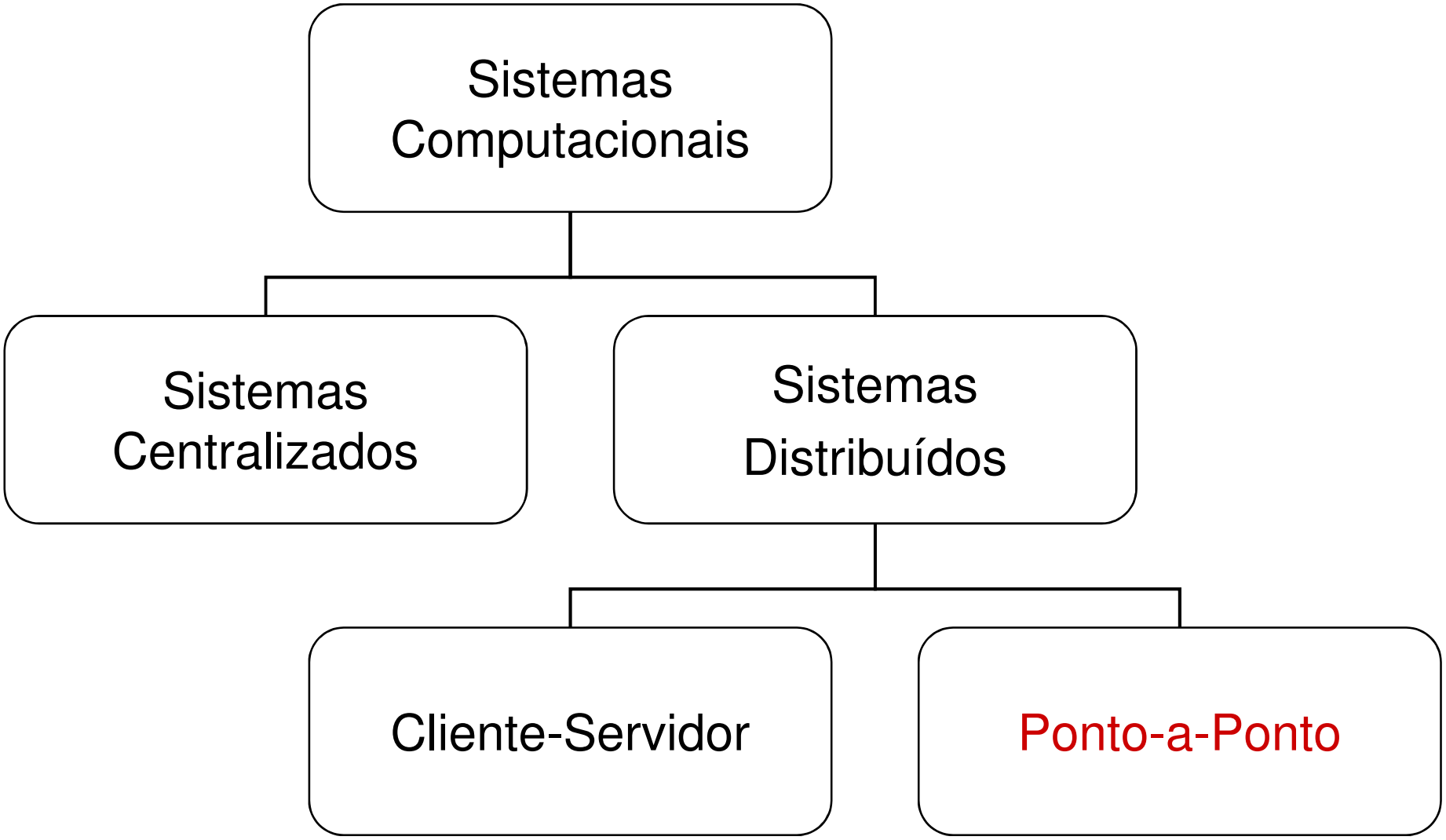
- Serviço
  - ✓ Funcionalidades oferecidas os pontos
    - Transferência de conteúdo
    - Disponibilização de status
  - ✓ Motivação para agrupamento de pontos em uma rede P2P
- Anúncio
  - ✓ Forma de comunicar a disponibilidade de um recurso por um ponto

Terminologia

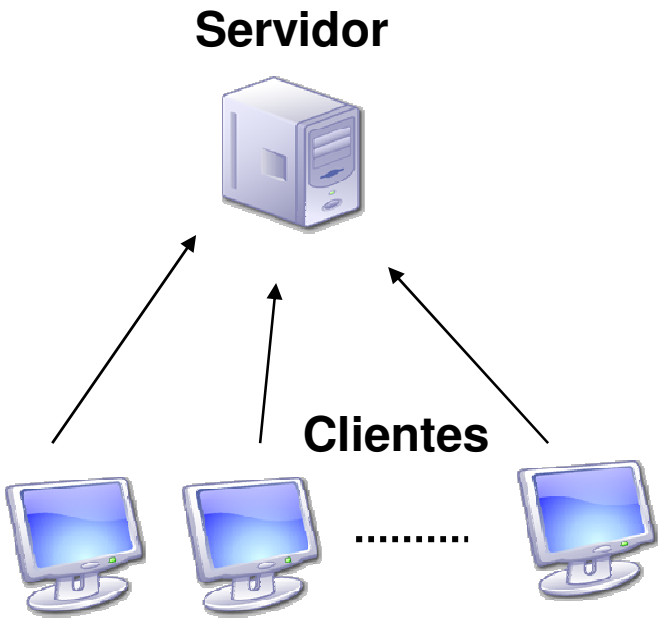
- Rede overlay
  - ✓ Rede virtual criada sobre uma rede já existente
  - ✓ Não é exatamente igual à rede física



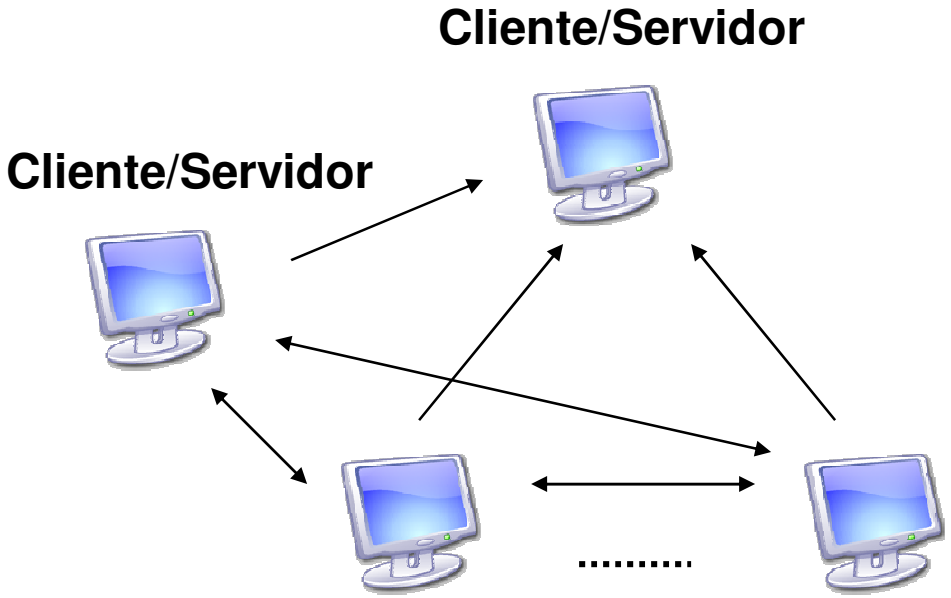
Classificação dos Sistemas Computacionais



Sistemas Distribuídos



(a) Cliente/Servidor Típico



(b) Ponto-a-Ponto Típico

Paradigma Ponto-a-Ponto (P2P)

Consiste de uma ampla rede de
pontos computacionais
(ou nós de informação)
interconectados
que cooperam uns com os outros,
trocando serviços e informações

Cada ponto compartilha recursos com os outros pontos e se beneficia dos recursos dos demais

Paradigma Ponto-a-Ponto (P2P)

- Recursos compartilhados
  - ✓ Recursos computacionais
    - Espaço em disco
    - Processamento
  - ✓ Recursos de rede
  - ✓ Conteúdo

Paradigma Ponto-a-Ponto (P2P)

- Sistemas P2P
  - ✓Pontos possuem relativamente as mesmas características e funções
  - ✓Pontos trocam mensagens através dos seus *links* lógicos sem a interferência de um coordenador (ponto servidor)
  - ✓Pontos são organizados através de uma rede lógica (*Overlay Network*) no nível da aplicação

Paradigma Ponto-a-Ponto (P2P)

- Principais características
  - ✓ Sem coordenação central
  - ✓ Sem repositório central
  - ✓ Sem local único de falha ou gargalo
  - ✓ Nenhum ponto tem visão global do sistema
  - ✓ Todos os dados e serviços são acessíveis de qualquer ponto
  - ✓ Pontos são autônomos
  - ✓ Pontos e conexões não são confiáveis

Sistemas Ponto-a-Ponto (P2P)

- Tipos de Sistemas
  - ✓ Não Estruturados
  - ✓ Estruturados
- Não Estruturados
  - ✓ Sem restrição de localização dos dados
  - ✓ Principal aplicação: compartilhamento de arquivos
  - ✓ Busca por palavra-chave
  - ✓ Alta disponibilidade de arquivos (réplicas nos pontos)

Sistemas Ponto-a-Ponto (P2P)

- Estruturados
  - ✓ Referenciados como *Distributed Hash Tables (DHT)*
  - ✓ Alta escalabilidade
  - ✓ Boa cobertura e alta precisão
  - ✓ Dois aspectos importantes
    - Busca aos dados
    - Acesso aos dados

} camada virtual de rede *(overlay network)*

Paradigma Ponto-a-Ponto (P2P)

- Vantagens
  - ✓ Poder computacional (recursos dos demais pontos)
  - ✓ Pontos com diferentes papéis (cliente ou servidor)
  - ✓ Compartilhamento de recursos
    - Melhor desempenho, tolerância a falhas (replicação)
  - ✓ Autonomia dos pontos participantes
    - Ausência de administração
  - ✓ Escalabilidade (e.g. KaZaA com ~3-4 milhões de usuários)

Paradigma Ponto-a-Ponto (P2P)

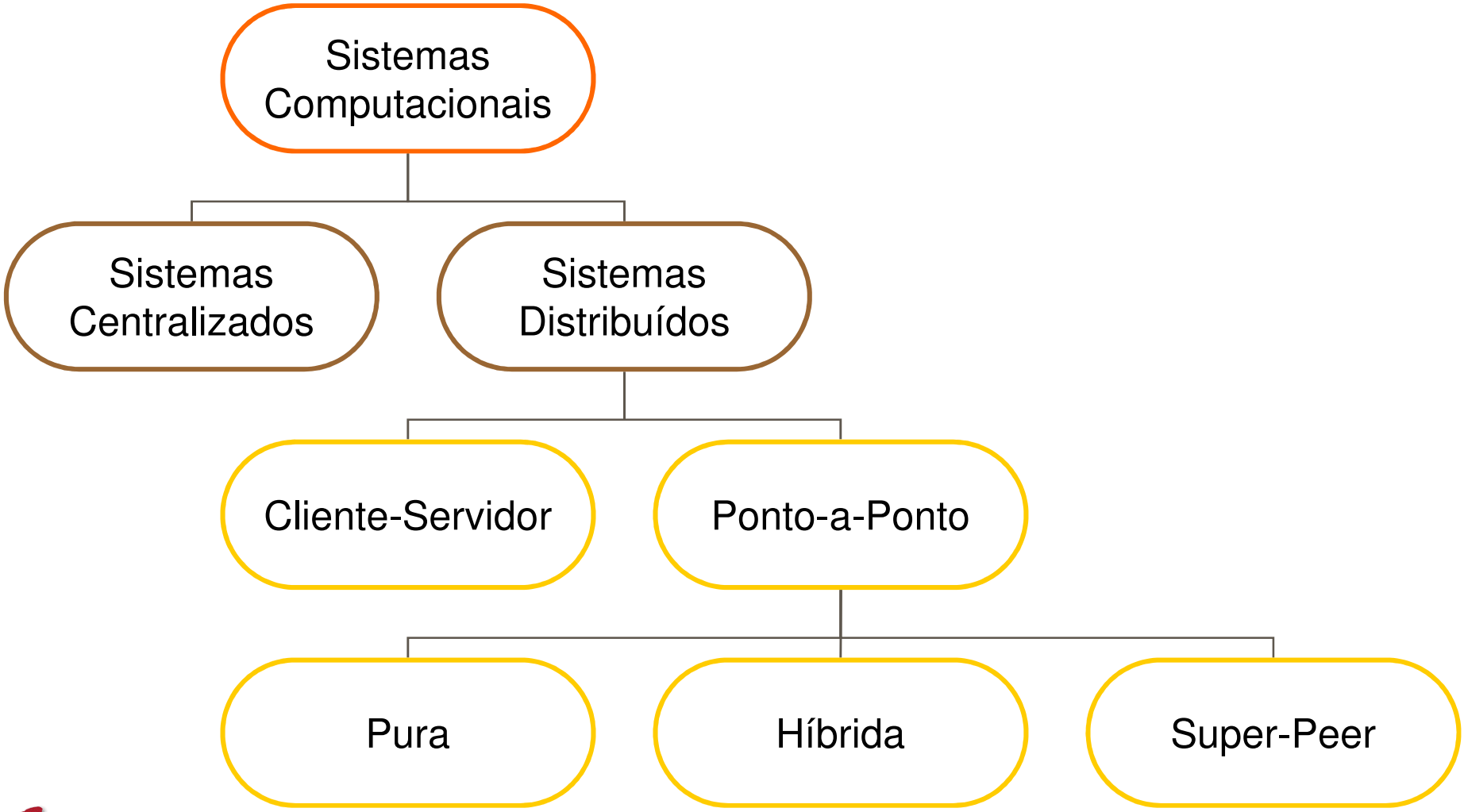
- Desvantagens
  - ✓ Ausência de tratamento semântico na troca de dados
  - ✓ Problemas com disponibilidade e consistência
  - ✓ Falta de estratégia para distribuição dos dados
  - ✓ Pode prejudicar o desempenho de pontos
  - ✓ Ausência de administração centralizada
  - ✓ Usuários responsáveis por gerenciar seus próprios recursos
  - ✓ Segurança

Topologias de Redes Ponto-a-Ponto

Topologias de Redes P2P

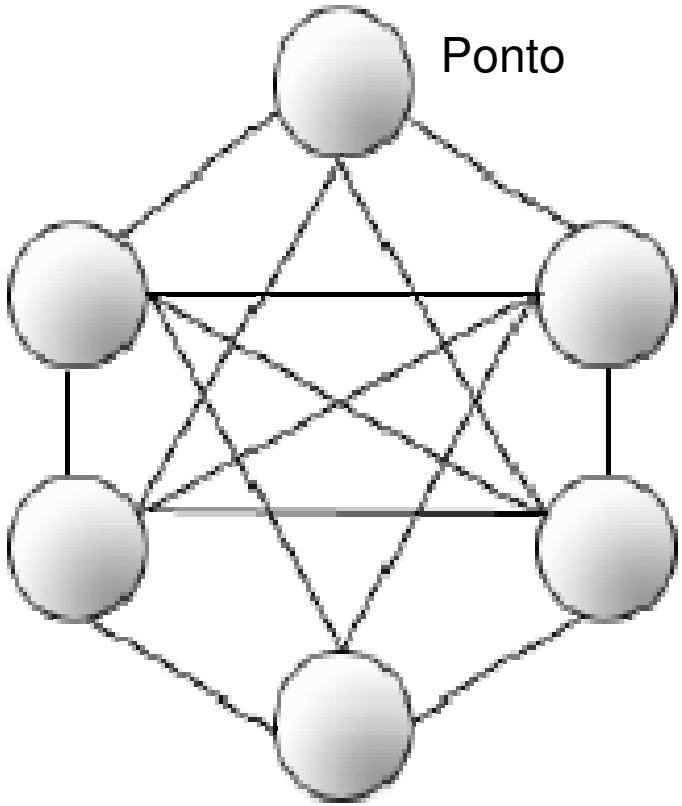
- Topologia
  - ✓ Define a organização lógica dos pontos na rede
- Tipos [Fiorano 2003]
  - ✓ Pura
  - ✓ Híbrida
  - ✓ *Super-Peer*

Topologias de Redes P2P



Topologia Pura

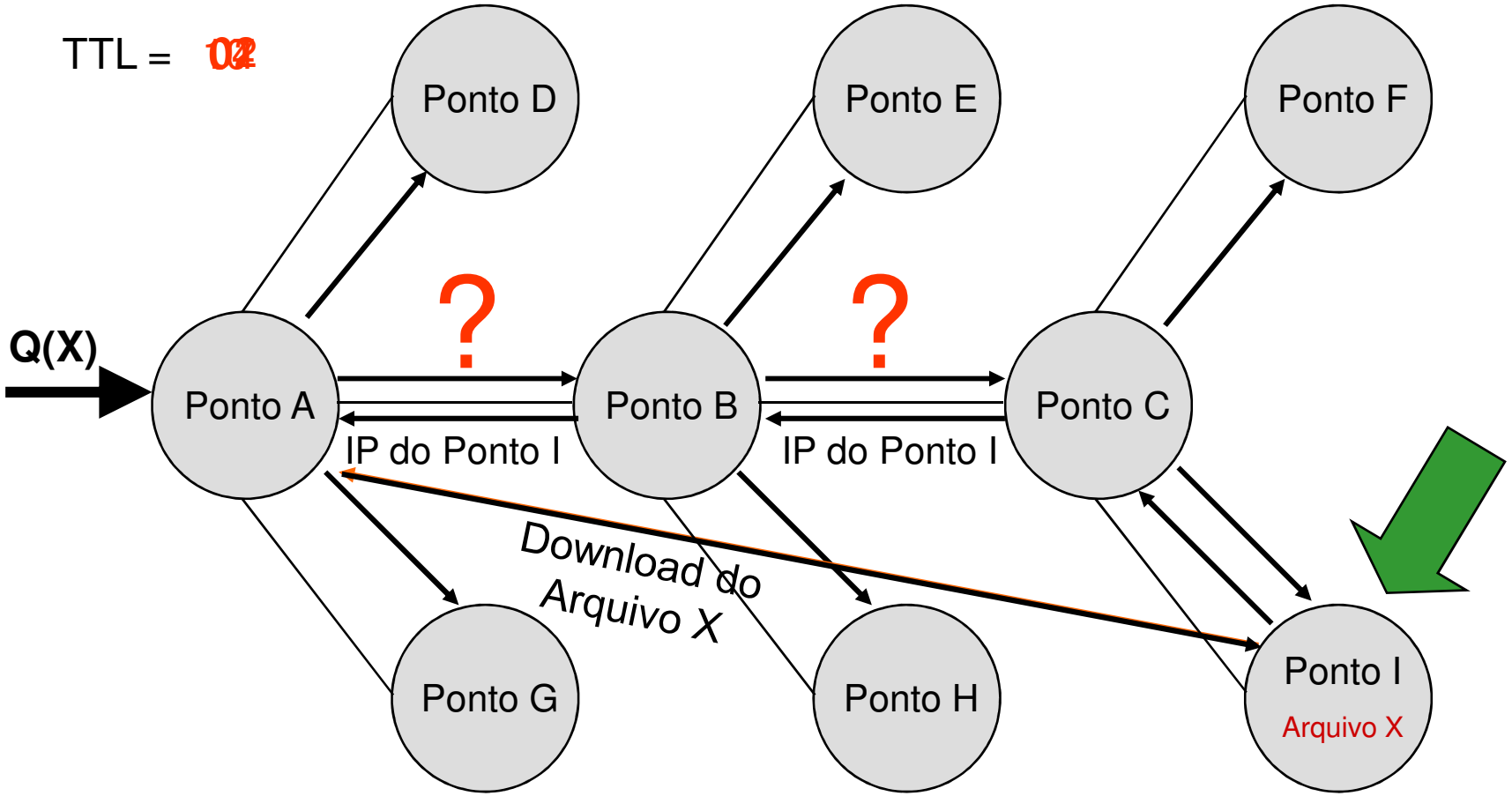
- Inexistência de um servidor ou repositório centralizado
- Todos os pontos são “iguais” e conectados entre si
- Busca
  - ✓ Não-Estruturada
    - *Flooding*
    - TTL (*time-to-live*)
  - ✓ Estruturada: DHT
- Sistemas: Gnutella, Freenet



Topologia Pura

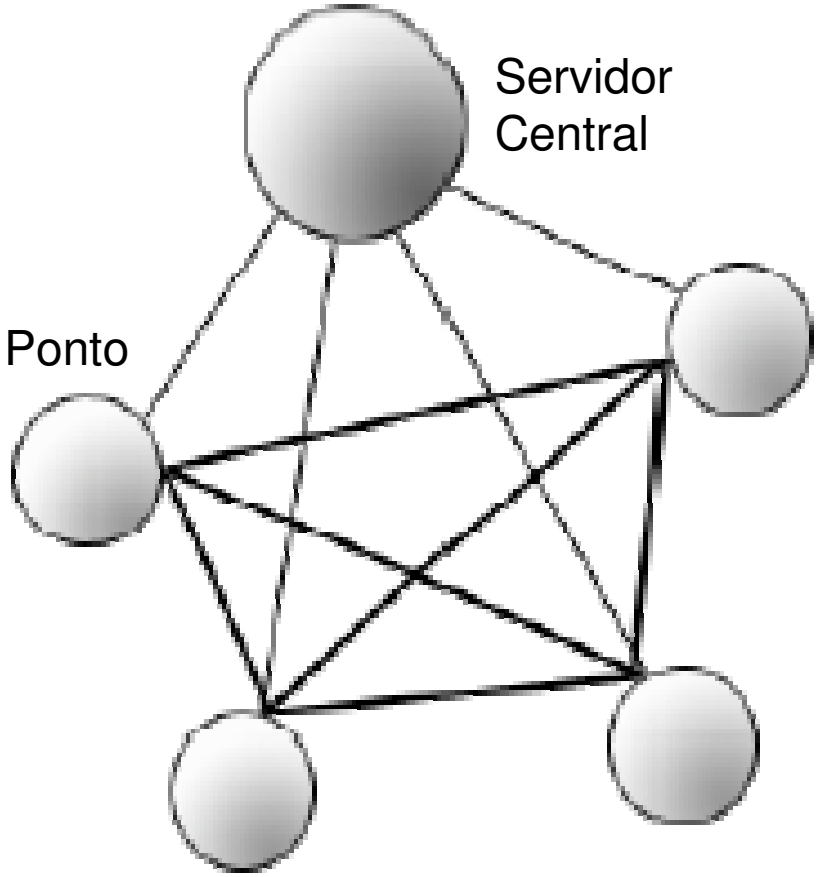
- Responsabilidades do ponto, como cliente:
  - ✓ Enviar pedidos de serviço a outros pontos
  - ✓ Receber as respostas dos pedidos feitos
- Responsabilidades do ponto, como servidor:
  - ✓ Receber pedidos de serviço de outros pontos
  - ✓ Processar os pedidos e executar um serviço requerido
  - ✓ Enviar a resposta com os resultados do serviço requerido
  - ✓ Propagar os pedidos de serviço a outros pontos

Busca na Topologia Pura Não-Estruturada



Topologia Híbrida

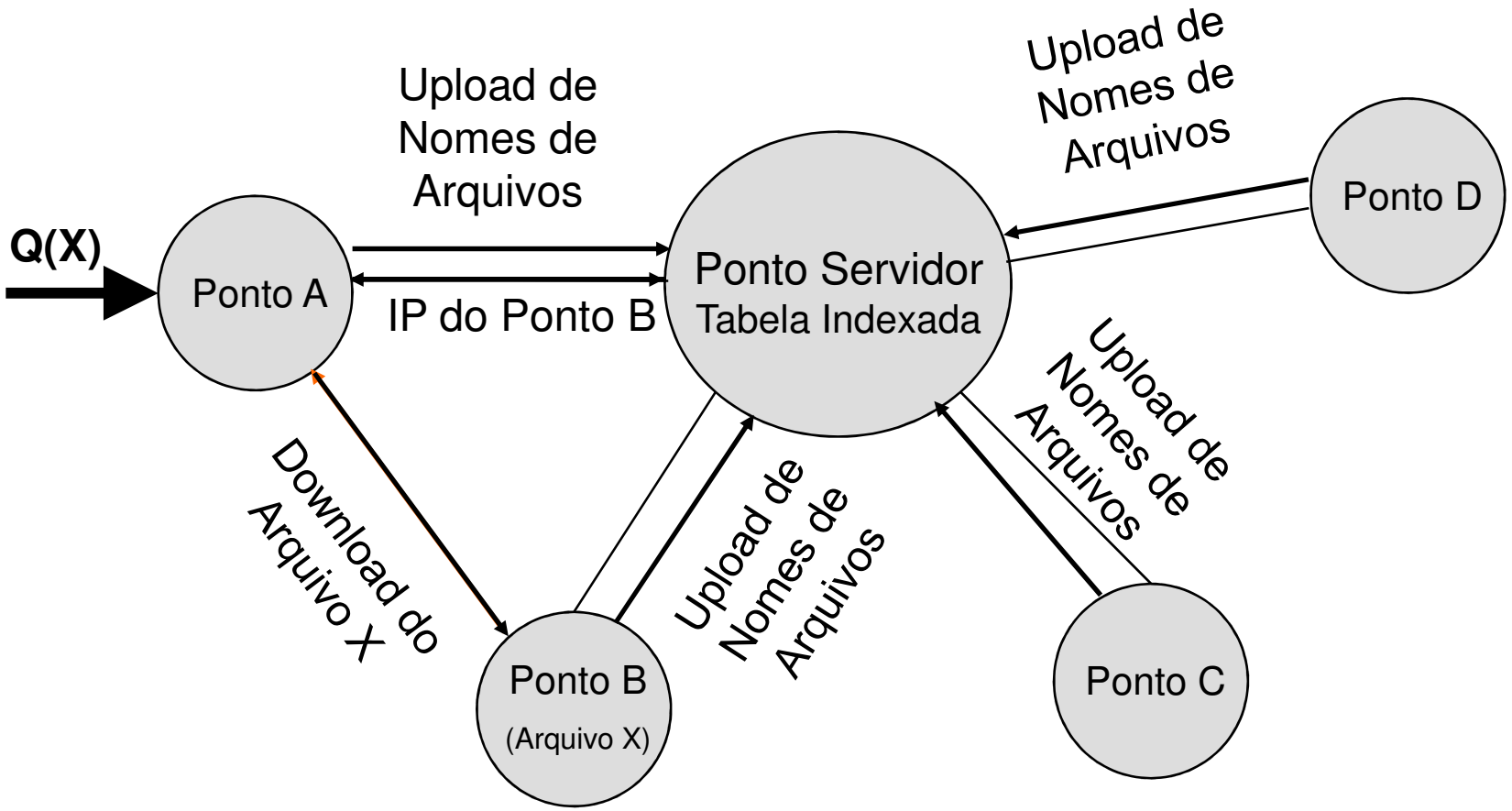
- Existência de um ou mais servidores centrais
- Informações de controle são armazenadas e fornecidas por um servidor central
- Gerência facilitada
- Servidor central representa um ponto único de falha
- Sistema: Napster



Topologia Híbrida

- Responsabilidades do ponto, como cliente:
  - ✓ Registrar no servidor seus serviços disponíveis
  - ✓ Enviar ao servidor pedidos de busca por serviços e receber respostas contendo listas de pontos com os serviços desejados
  - ✓ Enviar a outros pontos pedidos de serviço e receber as respostas destes pedidos
  - ✓ Processar e executar os serviços requerido e enviar repostas a quem fez o pedido
- Responsabilidades do ponto, como servidor:
  - ✓ Registrar serviços disponíveis nos pontos
  - ✓ Receber pedidos de busca por serviços disponíveis, buscar por esses serviços e enviar respostas com as localizações dos serviços desejados

Busca na Topologia Híbrida



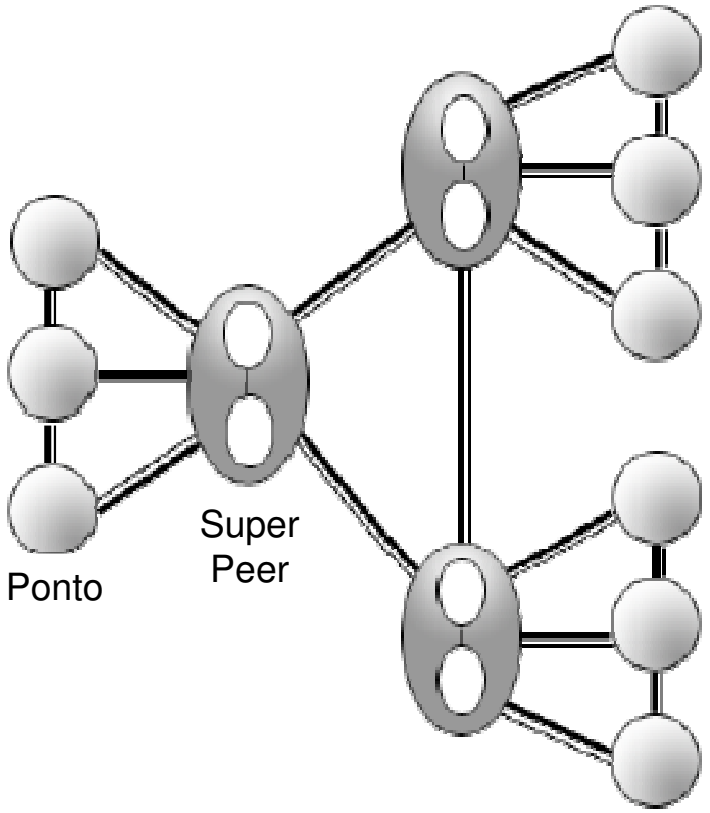
Topologia *Super-Peer*

Considera:

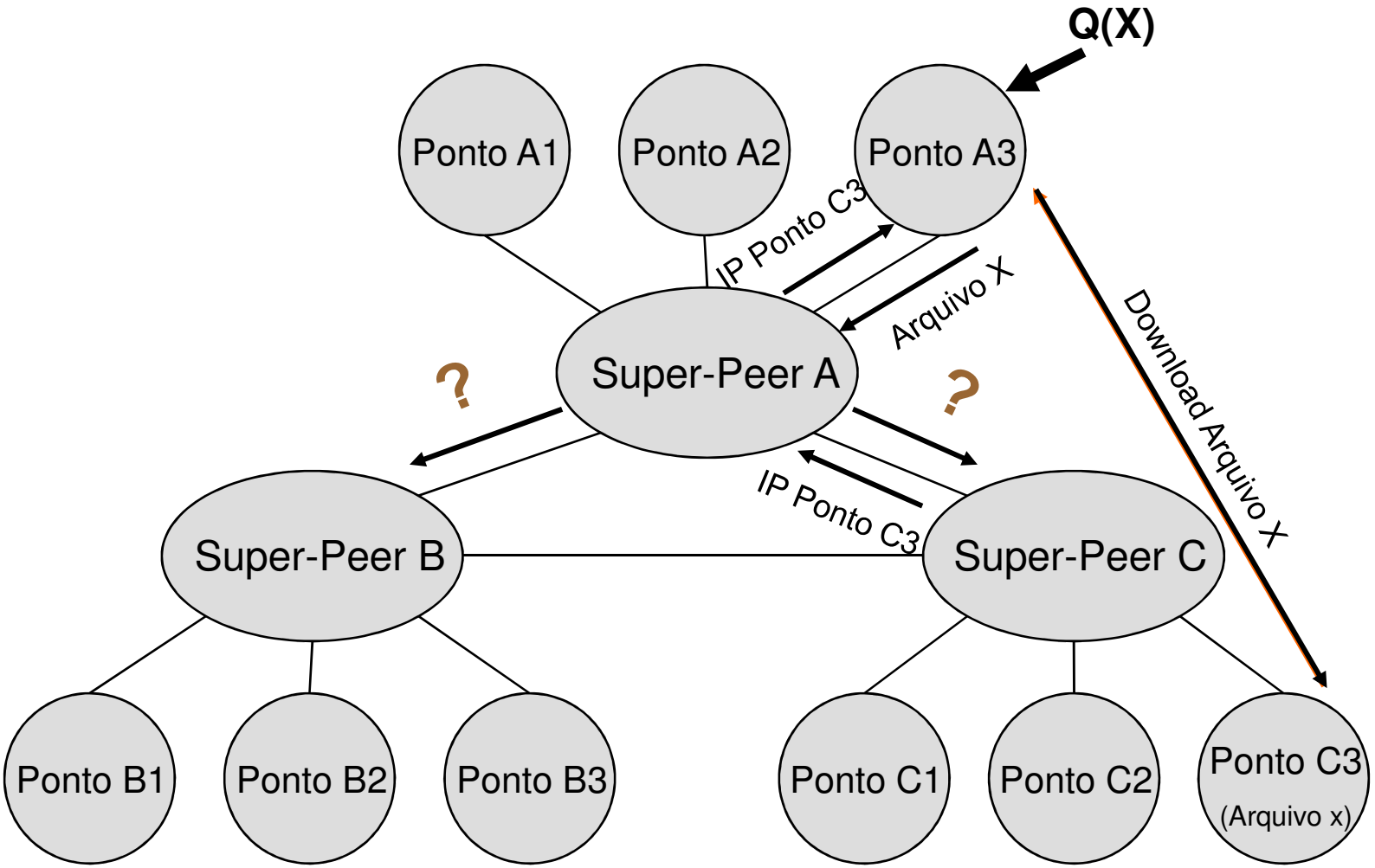
- Heterogeneidade dos pontos
- Muitos pontos em conexões de baixa capacidade e alta instabilidade
- Poucos pontos em conexões de alta capacidade e baixa instabilidade

Topologia *Super-Peer*

- Pontos heterogenêneos
- Organização hierárquica
- Grupos de pontos comunicam-se com outros grupos através de *super-peers*
- Cada *super-peer* indexa as informações armazenadas no seu conjunto de pontos
- Sistemas: KaZaA, Morpheus

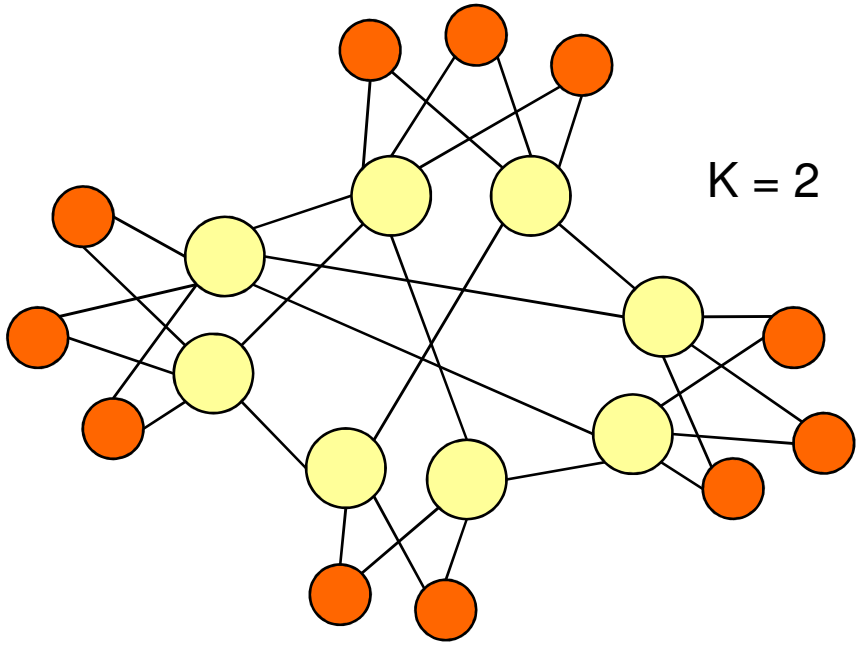


Busca na Topologia *Super-Peer*



Desafios da Topologia *Super-Peer*

- Qual a taxa ideal de pontos por *super-peer*?
- Como os *super-peers* devem conectar-se entre si?
  - ✓ Topologia estruturada ou não-estruturada?
- Variação:
  - ✓ K-redundant



Comparativo entre Topologias

Arquitetura	Segurança (Pontos Maliciosos)	Consistência (Dados)	Escalabilidade (Entrar e Sair)	Confiabilidade (Ponto de Falha)
P2P Pura				
P2P Híbrida				
Super-Peer				

[Fiorano 2003]

Propriedades dos Sistemas P2P

Principais Propriedades dos Sistemas P2P

- Conectividade
- Auto-Organização
- Descentralização
- Escalabilidade
- Roteamento
- …

[Walkerdine 2002]

Conectividade

- *Ad-hoc* e dinâmica
- Envolve
  - ✓ Conexão
  - ✓ Desconexão (normal, falha)
- Conexão de um ponto na rede
  - ✓ Feita através de outro que já esteja participando
- Alguns pontos podem atuar como *entry points*
- Pontos relacionados devem ficar “próximos” uns dos outros

Auto-Organização

- Capacidade dos pontos se realocarem na rede após a ocorrência de um evento
  - ✓ Conexão
  - ✓ Desconexão e/ou Falha
  - ✓ *Timeout*
- A inexistência de uma administração centralizada faz com que a reorganização de rede P2P fique ao encargo dos próprios pontos

Descentralização

- Dados e metadados estão distribuídos entre os pontos
- Não existe um servidor central responsável por tarefas como
  - ✓ Reorganização da rede
  - ✓ Armazenamento de metadados
- Próprios pontos devem ser responsáveis por tais tarefas
- Inexistência de ponto único de falha

Escalabilidade

- Capacidade da rede P2P crescer sem ficar sobrecarregada
- Sistema cliente-servidor
  - ✓ Administradores podem estender ou rebalancear os recursos computacionais para compensar o crescimento da rede
- Sistema P2P
  - ✓ Soluções devem estar embutidas em cada ponto

Escalabilidade

- Depende da topologia adotada
  - ✓ Híbrida
    - Dificuldade em tratar a escalabilidade
    - Pontos centrais podem necessitar de balanceamento e/ou expansão física do *hardware* para compensar o crescimento da rede
    - Preocupação com os custos de manutenção dos pontos centrais
    - Contra-exemplo: Napster mostrou-se robusto e eficiente
  - ✓ Pura
    - Sobrecarga de troca de mensagens para descoberta de novos pontos e buscas na rede
  - ✓ *Super-Peer*
    - Divisão e/ou fusão (*coalesce*) de *clusters*

Roteamento

- Principais mecanismos de roteamento para redes P2P
  - ✓ Híbrido
  - ✓ *Flooding* (ou inundação): modelo descentralizado não-estruturado
  - ✓ Tabela Hash Distribuída (DHT): modelo descentralizado estruturado
  - ✓ *Semantic Overlay Network* (SON)

Roteamento – Flooding

- Problemas
  - ✓ Excesso de mensagens
  - ✓ Mensagens duplicadas
  - ✓ Valor ideal de TTL
    - TTL alto: sobrecarga na rede
    - TTL baixo: nenhum resultado encontrado
- Variações
  - ✓ Busca informada: uso de cache local
  - ✓ Busca informada com replicação
  - ✓ Aprofundamento iterativo: múltiplos valores crescentes para TTL

Roteamento – Modelo DHT

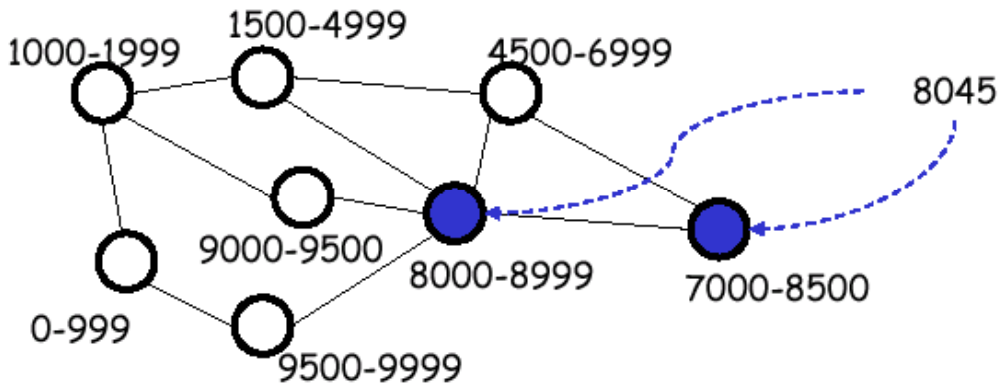
- Tentativa de melhorar os algoritmos de roteamento dos sistemas P2P não-estruturados
- Itens (arquivos) são distribuídos entre os pontos de acordo com um algoritmo
  - ✓ Pontos não escolhem os itens à vontade
  - ✓ Uso de replicação para garantir disponibilidade

Roteamento – Modelo DHT

- Função hash
  - ✓ Mapeia um ponto em um identificador único
    - h('172.17.166.99') → 8400
  - ✓ Mapeia um item (arquivo) em um identificador único
    - h('TutorialP2P.ppt') → 8045
  - ✓ Qualquer função aleatória de *hash* "boa" é suficiente
    - Padrão SHA-1 (colisão praticamente impossível)
- Faixa de resultados da função *hash* é distribuída pela rede

Roteamento – Modelo DHT

- Cada ponto é responsável por armazenar itens cujo identificador é igual ou próximo ao identificador do ponto

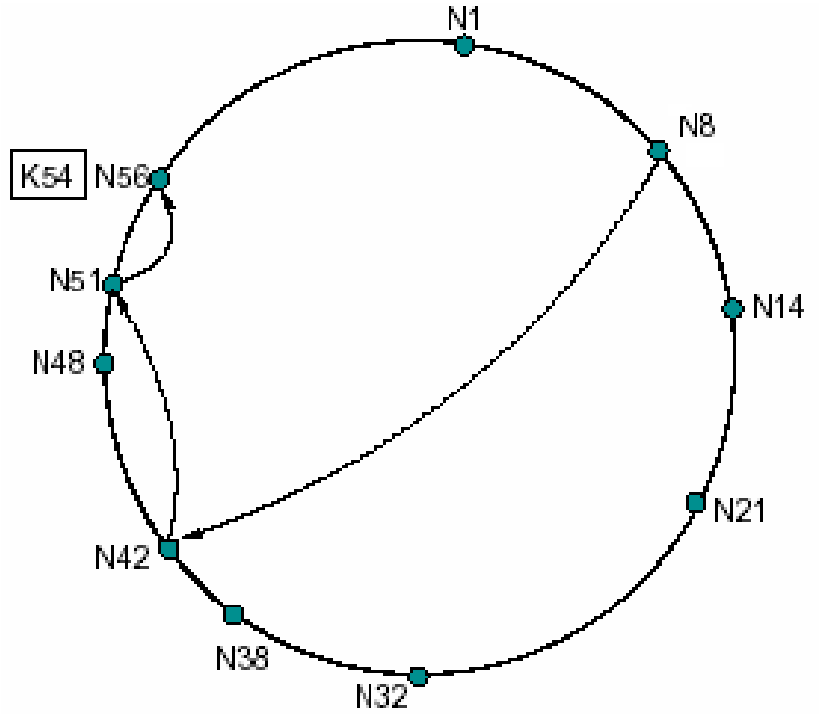


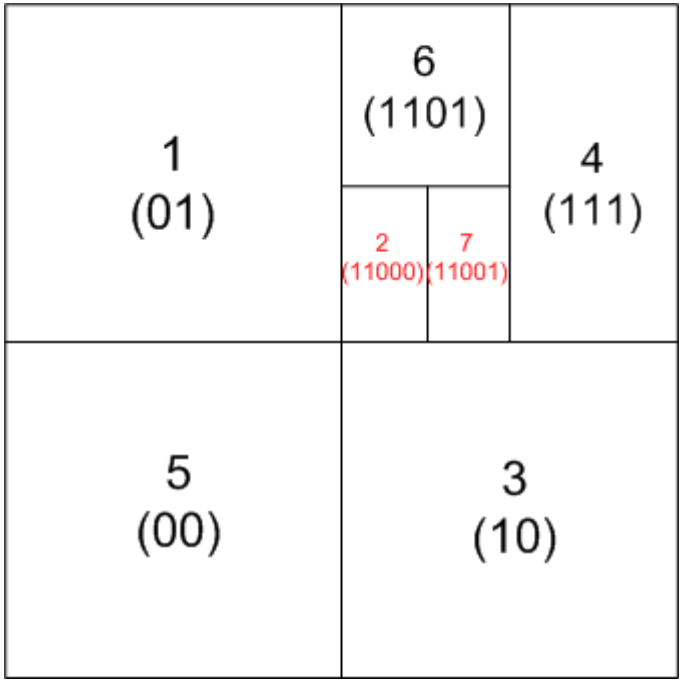
- Dado um identificador, um ponto deve ser capaz de encaminhar a consulta para o ponto cujo identificador mais se aproxima

Roteamento – Modelo DHT

- Para cada objeto, o(s) ponto(s) cuja faixa “cobre” o objeto deve ser alcançável por um caminho “curto”
  - ✓ De qualquer outro ponto
- Abordagens
  - ✓ Chord, CAN, Pastry, Tapestry, ...
  - ✓ Diferem na escolha do algoritmo de roteamento (determina a geometria da rede)
- Geometrias
  - ✓ Anel: Chord
  - ✓ Árvore: Pastry, Tapestry
  - ✓ XOR: Kademlia
  - ✓ Hipercubo: CAN
  - ✓ Híbrida: Pastry (pode trabalhar como anel)

Roteamento – Modelo DHT





(a) Chord **(b) CAN**

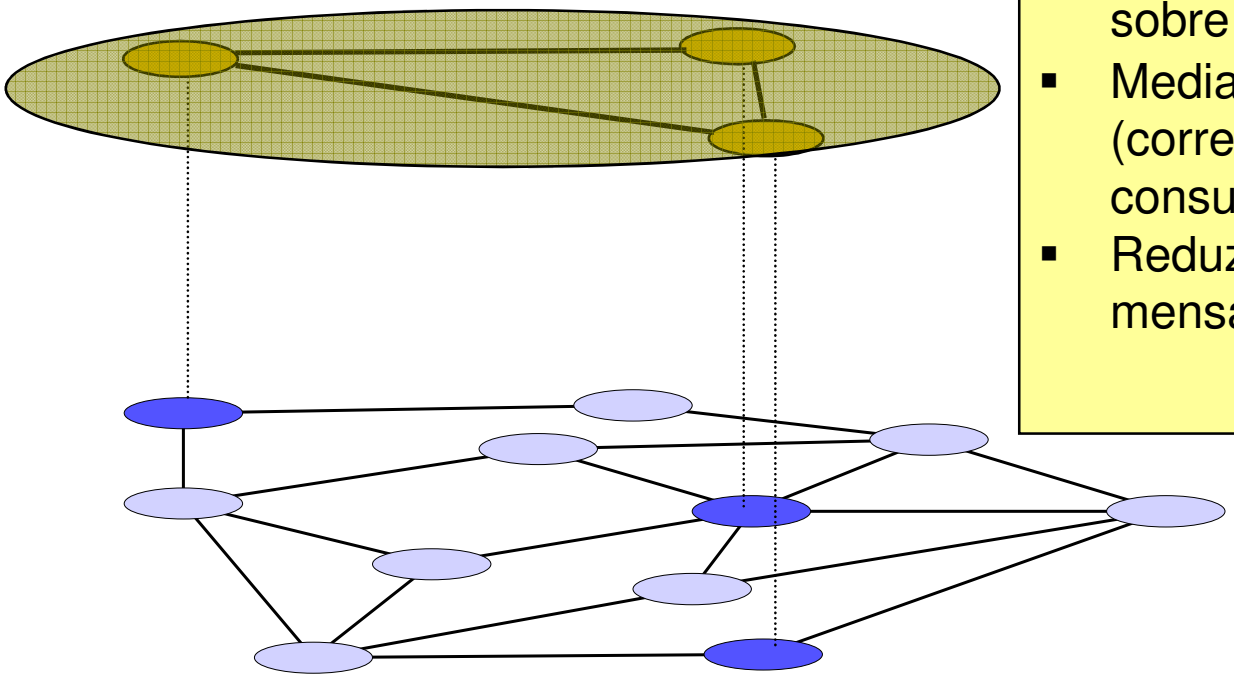
Roteamento

- Ineficiência de consultas no modelo de inundação (escalabilidade)
- Consultas no modelo DHT
  - ✓ Escalonável, porém “pobre”
  - ✓ Não permite
    - Consultas por aproximação
    - Consultas por faixa
- Uma consulta deve ser enviada apenas para os pontos aptos a respondê-la
- Em geral, é possível representar o conteúdo compartilhado através de ontologias
  - ✓ Música, filmes, artigos científicos, …

Roteamento - SON

Semantic Overlay Network

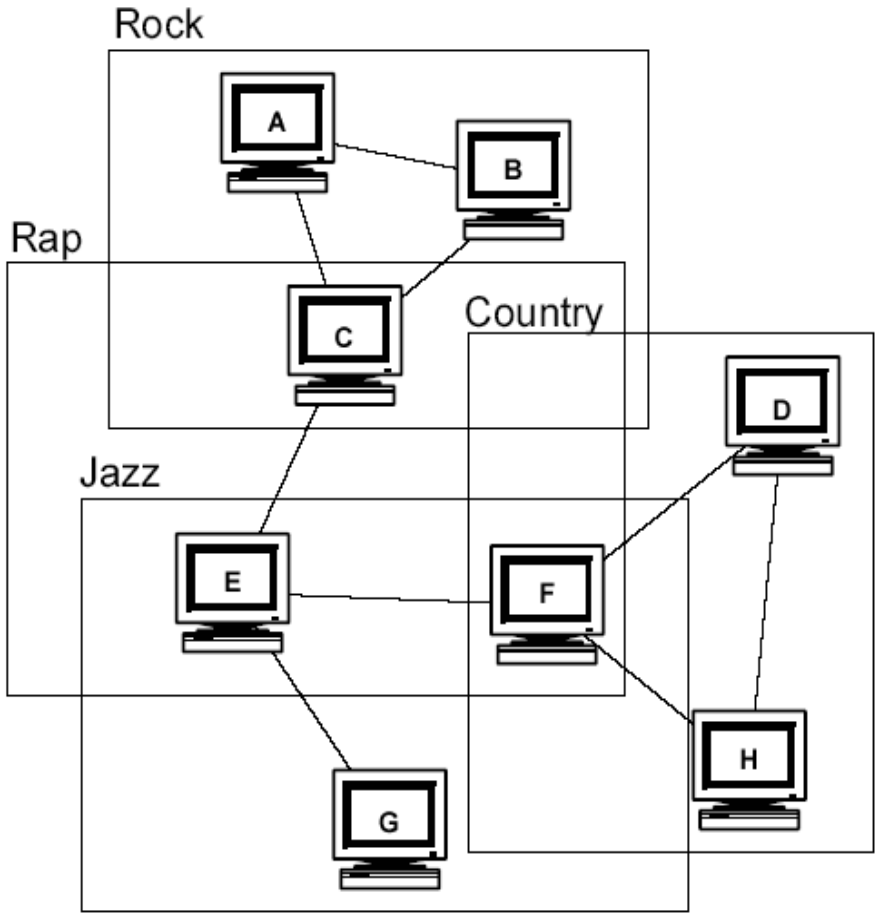
- Virtual, abstrata, camada independente de pontos selecionados



Vantagens

- Introduz visões semânticas sobre a rede física
- Mediação e integração (correspondências, reescrita de consultas)
- Reduz a quantidade de mensagens na rede

Roteamento - SON

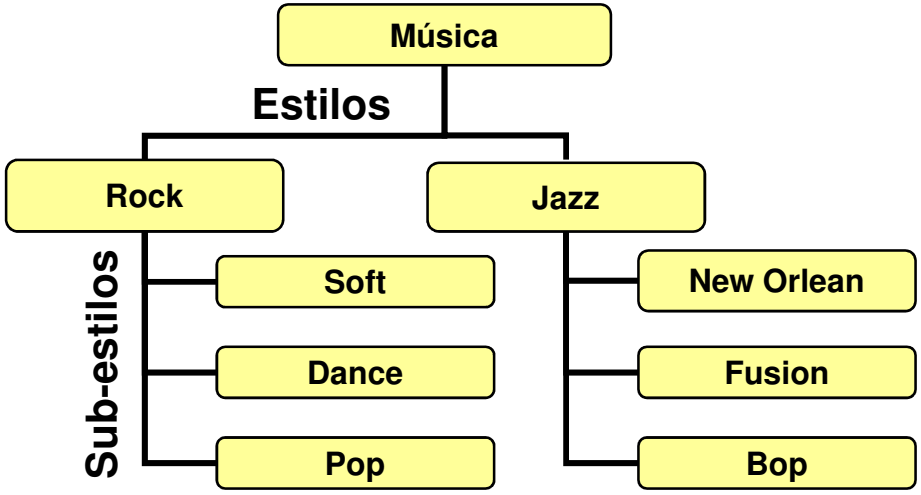


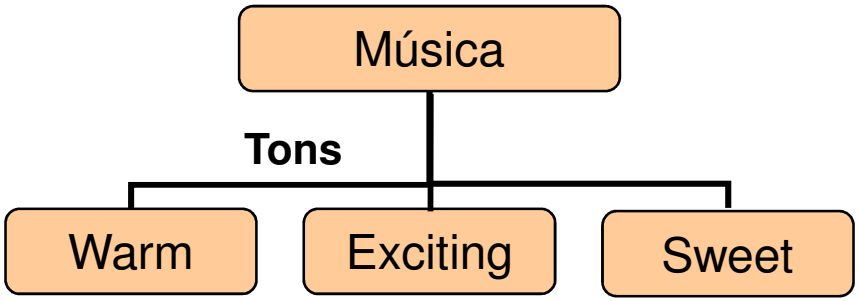
- Pontos agrupados em *clusters*
- *Overlap* de *clusters*
- Consultas enviadas apenas para *clusters* relevantes
- *Clusters* irrelevantes são descartados

[Garcia-Molina et al. 2002]

Roteamento - SON

- SON associada ao conceito de hierarquia de classificação
- Exemplos
  - ✓ 9 SON para classificação de músicas por estilo
  - ✓ 4 SON para classificação de músicas por ton





SON: Critérios para Definição da Hierarquia

- Documentos devem ser associados a conceitos para que o ponto seja associado a(s) SON correspondente(s)
- Pontos devem possuir documentos em um número reduzido de categorias
- Algoritmo de classificação eficiente

SON: Estratégias para Alocação de Pontos

- Estratégia conservadora
  - ✓ Aloque um ponto em SON_c, se o mesmo possui algum documento classificado no conceito *c*

Produz muitos links

- Estratégia menos conservadora
  - ✓ Aloque um ponto em SON_c, caso o mesmo possua uma quantidade de documentos "significativa" classificados no conceito *c*

Evita que sejam encontrados todos os documentos

Outras Propriedades de Sistemas P2P

- Autonomia
  - ✓ Um ponto não deve ter controle sobre os recursos compartilhados por outro ponto
- Anonimato
  - ✓ Capacidade do sistema ocultar a identificação do usuário
- Descoberta de Recursos
  - ✓ Os pontos devem comunicar a disponibilidade de recursos através de anúncios

Plataformas de Desenvolvimento P2P

Plataformas de Desenvolvimento

- Desenvolvimento de sistemas P2P
  - ✓ Implementação de protocolo de comunicação próprio para permitir a interoperabilidade entre os pontos da rede
- Conseqüências
  - ✓ Esforço de implementação adicional
  - ✓ Aplicações que oferecem o mesmo serviço não podiam comunicar-se entre si
  - ✓ Exemplo
    - Arquivos compartilhados por sistemas como o KaZaA, e-Mule e Gnutella
    - São acessíveis exclusivamente dentro de suas próprias redes
    - Instalação de múltiplos softwares

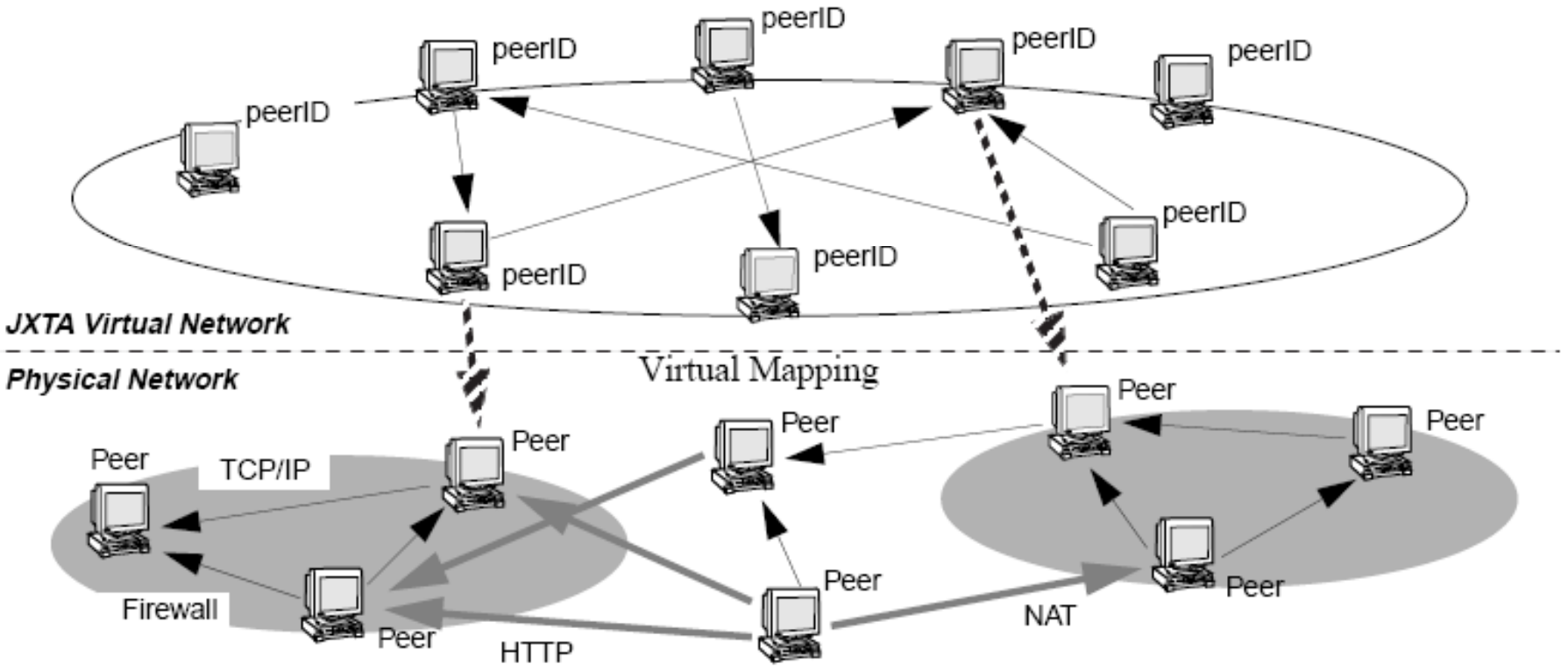
Plataformas de Desenvolvimento

- Popularização dos sistemas P2P
  - ✓ Esforço maior em construir plataformas e *frameworks* de desenvolvimento
  - ✓ Aplicações distintas podem comunicar-se entre si naturalmente
- Exemplos de plataformas
  - ✓ JXTA
  - ✓ Web Services
  - ✓ .NET
  - ✓ Groove

Plataforma JXTA

- Plataforma aberta para desenvolvimento de sistemas P2P
- Começou como um projeto de pesquisa incubado na *Sun Microsystems*
- Oferece um conjunto de protocolos para comunicação, colaboração e compartilhamento de recursos, entre diferentes tipos de dispositivo, e.g. celulares, estações de trabalho, PDA e servidores
- Principais objetivos
  - ✓ Interoperabilidade (entre diferentes sistemas e comunidades P2P)
  - ✓ Independência de plataforma (diversas linguagens de programação, sistemas operacionais e redes)
  - ✓ Generalidade (qualquer tipo de dispositivo digital)
  - ✓ Segurança

Exemplo de Rede Virtual JXTA



Classes de Aplicações P2P

Classes de Aplicações P2P

- Computação Distribuída
- Troca de Mensagens
- Trabalho Colaborativo
- Compartilhamento de Dados
  - ✓ Não-estruturado
  - ✓ Estruturado/Semi-estruturado

Computação Distribuída

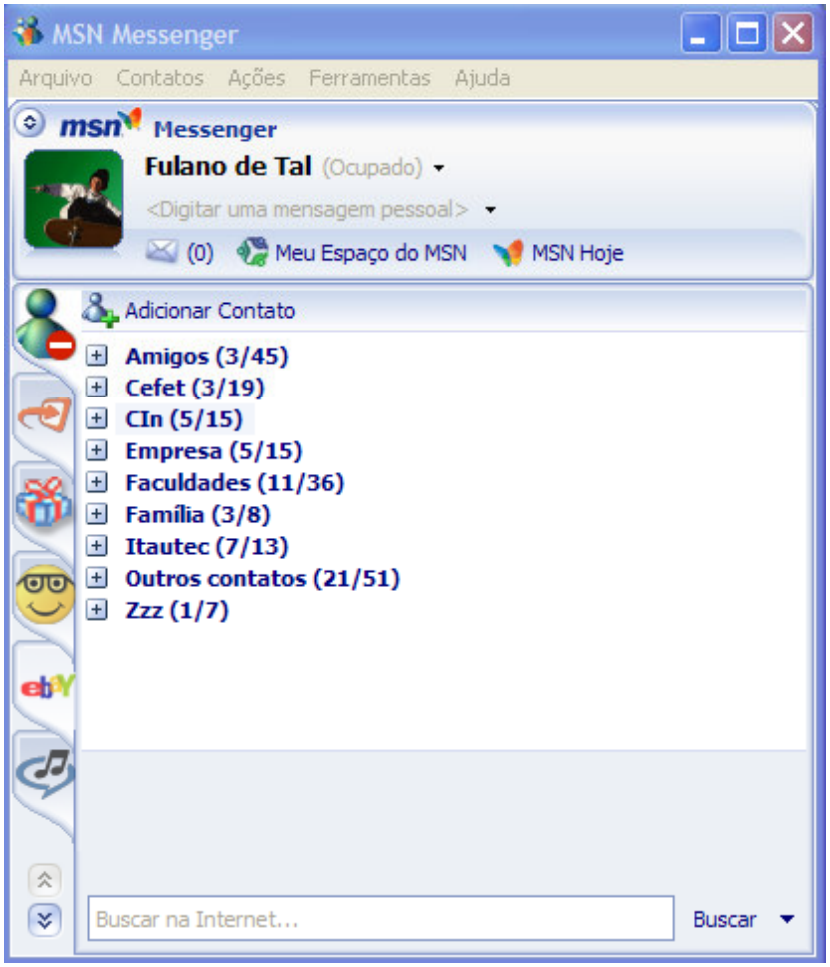
- A idéia de aproveitar recursos computacionais ociosos não é nova
- Grade Computacional
  - ✓ Solução de computação distribuída para engenharia e ciências, baseada em compartilhamento de recursos em larga escala
  - ✓ Discutiremos adiante
- SETI@Home (http://setiathome.ssl.berkeley.edu)
  - ✓ *The Search for Extraterrestrial Intelligence*
  - ✓ Usuários executam partes da "busca"

Computação Distribuída

Troca de Mensagens

- *Instant Messaging* (IM)
- Aplicação popular na Internet, pela facilidade de enviar mensagens *on-line*
- Exemplos
  - ✓ Yahoo! Messenger (http://messenger.yahoo.com)
  - ✓ AIM (AOL Instant Messenger - http://www.aim.com)
  - ✓ ICQ – (http://web.icq.com)
  - ✓ MSN Messenger (http://messenger.msn.com)
  - ✓ Trillian Messenger – (http://www.trilian.de)

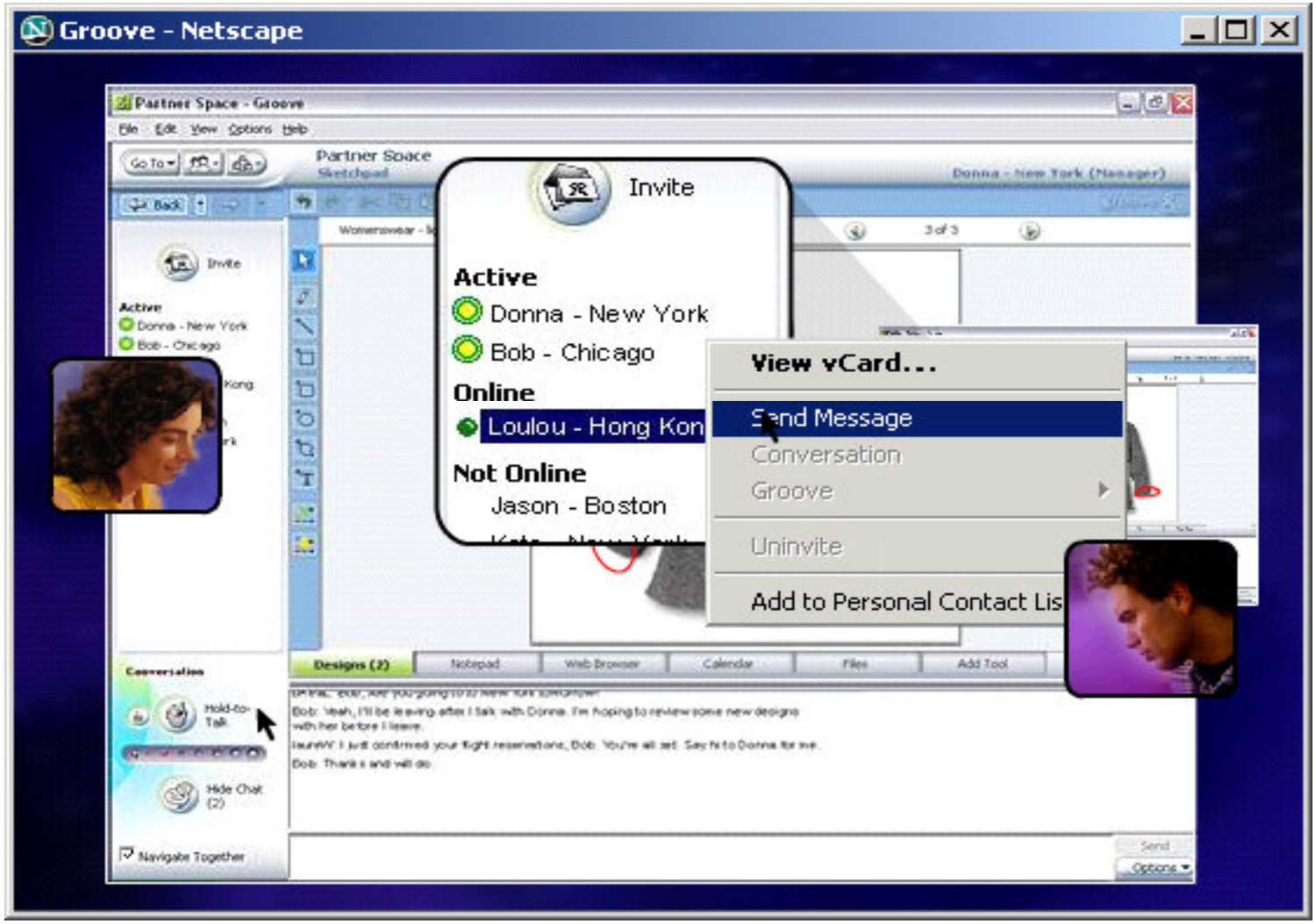
Troca de Mensagens



Trabalho Colaborativo

- Groupware
  - ✓ Aplicação que suporta colaboração, comunicação e coordenação de vários usuários em uma rede
  - ✓ Exemplo Cliente/Servidor: Lotus Notes
- A combinação de *groupware* com computação P2P trouxe novas oportunidades
- Groove (http://www.groove.net)
  - ✓ Usuários criam e compartilham espaços de trabalho
  - ✓ Oferece servidores centralizados para maior garantia
  - ✓ Oferece uma plataforma de desenvolvimento (GDK)

Trabalho Colaborativo



Compartilhamento de Arquivos

- Aplicação de maior sucesso na Internet
- Os usuários podem compartilhar diretamente seus arquivos, músicas, vídeos, entre outros
- Pode apresentar problemas de violação de direitos autorais
- Características
  - ✓ Área de armazenamento
  - ✓ Disponibilidade de informações
  - ✓ Anonimato
  - ✓ Gerenciamento

Compartilhamento de Arquivos

- Napster (http://www.napster.com)
- KaZaA (http://www.kazaa.com)
- Gnutella (http://www.gnutella.com)
- Freenet (http://www.freenetproject.org)
- BitTorrent (http://www.bittorrent.com)
- Morpheus (http://www.morpheus.com)

P2P e Grades Computacionais

Grades Computacionais

Paradigma para compartilhamento coordenado de recursos e resolução de problemas em organizações virtuais, multi-institucionais e dinâmicas

Grades Computacionais

- Computação em grade
  - ✓ Plataforma de computação distribuída geograficamente
    - Máquinas heterogêneas
    - Interface única de acesso
  - ✓ Disponibiliza camadas virtuais
    - Recursos computacionais variados e dispersos em um único "supercomputador virtual"
    - Aplicações de alto desempenho
  - ✓ Possibilita a execução de várias aplicações paralelas

P2P e Grades Computacionais

- Computação em Grade
  - ✓ Caso especial de computação P2P
  - ✓ Nós participantes dispõem de muitos recursos
  - ✓ Colaboração entre os nós é restrita e organizada
- Computação P2P é mais amigável
  - ✓ Usuários compartilham recursos e conteúdo de forma simples e gratuita
- Semelhanças
  - ✓ Modelos de computação distribuída
  - ✓ Computadores ligados em rede colaboram entre si, de modo descentralizado
  - ✓ Cada computador pode consumir e oferecer serviços

Comparativo P2P x Grades Computacionais

Critério	P2P	Grade Computacional
Nós Participantes	-Muitos (usuários comuns) -Conexões variadas -PCs	-Poucos (Organizações) -Redes de alta capacidade -Servidores (Clusters)
Segurança	-Comunidades abertas -Pouco explorado -Anonimato -Difícil de gerenciar	-Comunidades Fechadas -Muito explorado -Autenticação/Autorização/Confiabilidade -Mais simples de gerenciar
Conectividade	Instável	Estável
Consultas	Resultados incompletos	Resultados completos
Aplicações	Simples	Alto Desempenho
Descoberta de Recursos	Informações disponíveis em um ponto	Nós enviam mensagens periodicamente
Tolerância à Falhas	Bastante explorado	Pouco explorado

[Ooi et al. 2003b, Talia et al. 2003, Foster et al. 2002]

PDMS

Peer Data Management System

Sistemas de Gerenciamento de Dados Ponto-a-Ponto (PDMS)

Sistema de Gerenciamento de Dados com arquitetura descentralizada, facilmente extensível, na qual qualquer usuário pode contribuir com novos dados, novos esquemas, ou mapeamentos entre os esquemas dos pontos

Sistemas de Gerenciamento de Dados Ponto-a-Ponto (PDMS)

São uma evolução natural dos

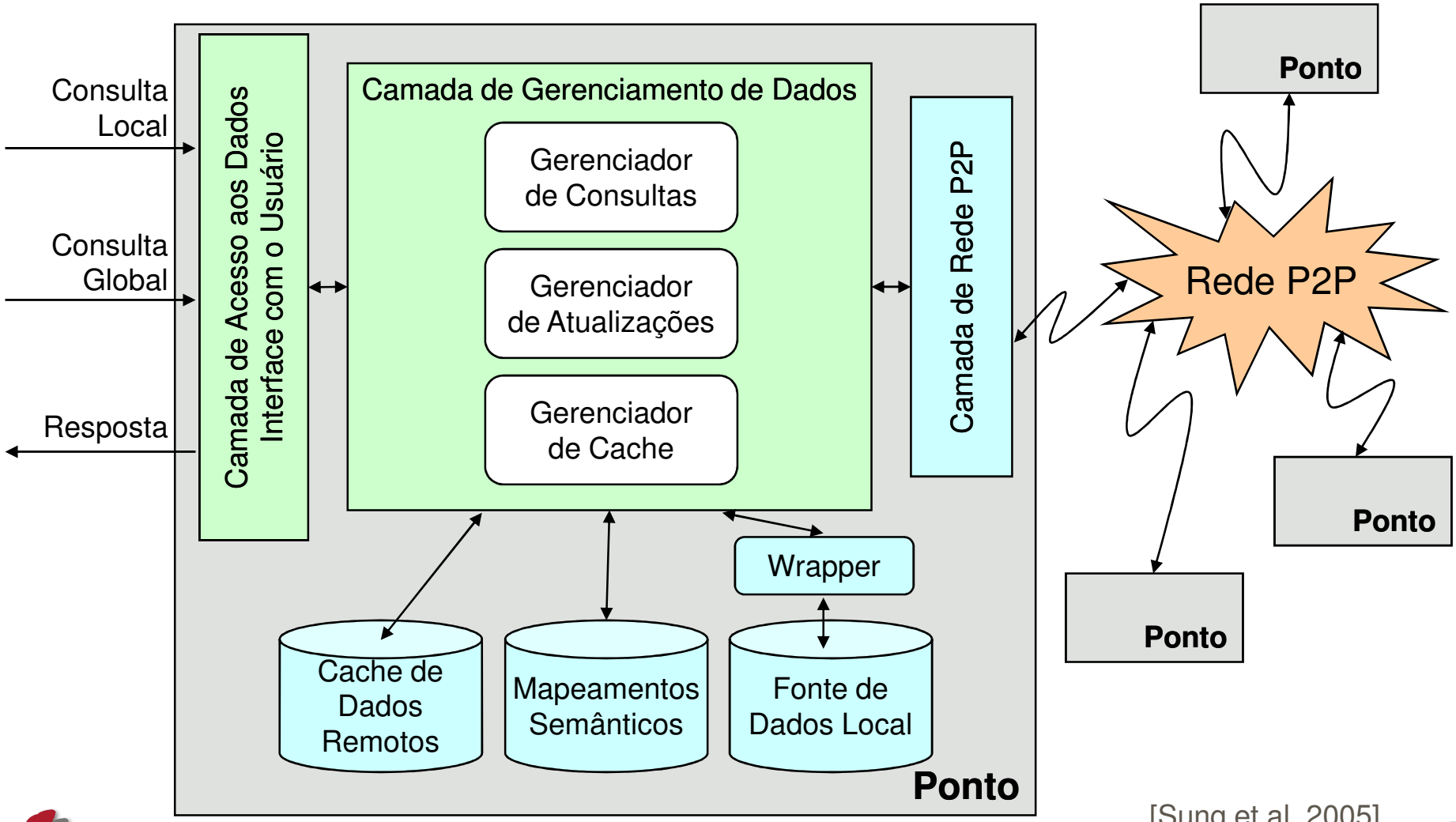
Sistemas de Integração de Dados

substituindo seu único esquema lógico (mediação) por uma coleção de mapeamentos semânticos entre os esquemas individuais de cada ponto

Sistemas de Gerenciamento de Dados Ponto-a-Ponto (PDMS)

- Algumas características
  - ✓ Autonomia: controle sobre os dados locais
  - ✓ Dinamismo: pontos e recursos podem entrar e sair a qualquer momento
  - ✓ Descentralização: cada ponto é independente dos outros
  - ✓ Cooperação: compartilhamento de recursos e serviços entre os pontos
  - ✓ Esquema local do BD: cada ponto tem seu esquema (ausência de esquema global)
  - ✓ Dados: podem estar incompletos, indisponíveis ou inconsistentes

Arquitetura Genérica de um Ponto



[Sung et al. 2005]

Sistemas de Gerenciamento de Dados Ponto-a-Ponto (PDMS)

- Gerenciamento de dados distribuídos
- Compartilhamento de dados em larga escala
- Solução depende fortemente da topologia adotada
- Impraticável a existência de esquema de mediação único

Problemas de um Esquema de Mediação Único

- Conflito com as propriedades dos sistemas P2P
- Dinamismo
  - ✓ Atualização do esquema de mediação a cada conexão e/ou desconexão
- Autonomia
  - ✓ Nem todos os pontos querem compartilhar todos os dados
- Escalabilidade
  - ✓ Onde armazenar um esquema de mediação único?
    - Centralizado
      - Ponto único de falha
      - Investimento em hardware e conectividade
    - Distribuído
      - Técnicas para garantir uma visão integrada do esquema de mediação
      - Esquema replicado: problemas de armazenamento e consistência

Pontos Positivos dos PDMS

- Não existe esquema global
  - ✓ Manutenção
- Mapeamentos definidos da forma mais conveniente (pontos “próximos”)
- Consultas são elaboradas de acordo com o esquema do ponto
  - ✓ Resultados vêm de qualquer lugar do sistema
- PDMS x Compartilhamento de Arquivos
  - ✓ Dados possuem semântica mais “rica”
  - ✓ Não são tão dinâmicos (conexão/desconexão)

Pontos Positivos dos PDMS

- Inexistência de Ponto Único de Falha
  - ✓ Dados altamente distribuídos usando *caching* local
  - ✓ Se um ponto falha, as consultas podem continuar
- Pouca administração
- Escopo dos dados
  - ✓ Grande quantidade de dados de diferentes tipos
- Replicação
  - ✓ Dados podem ser replicados nos pontos
  - ✓ Resultados de consultas podem ir para *cache* local
  - ✓ Acesso rápido aos dados locais se pontos originais não disponíveis

Gerenciamento de Dados em PDMS

Aspectos do Gerenciamento de Dados em Sistemas P2P

- Mapeamentos entre Esquemas
- Processamento de Consultas
- Consistência de Dados
- Localização de Dados
- Conectividade entre Pontos
- Tolerância a Falhas
- Qualidade de Dados

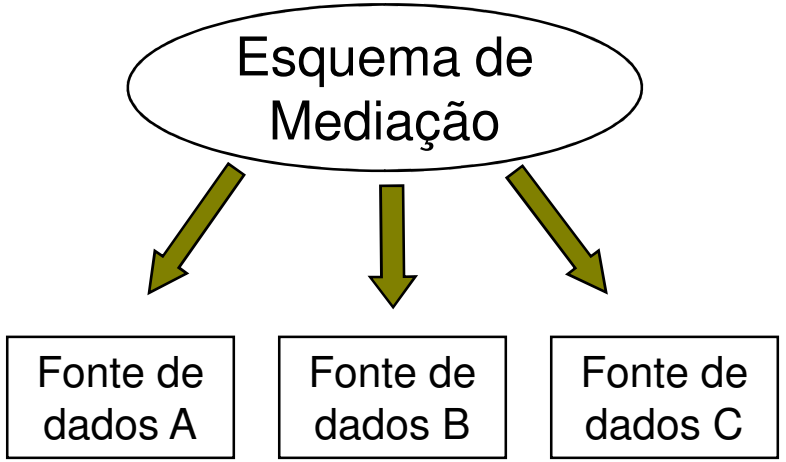
PDMS – Mapeamentos entre Esquemas

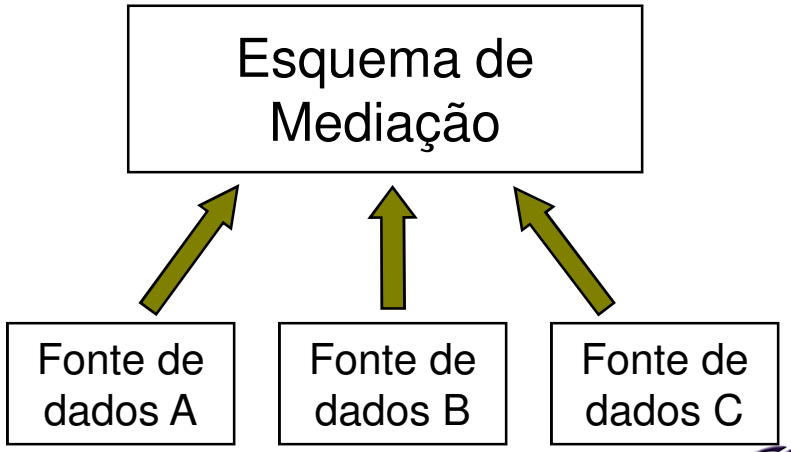
- Mapeamentos
  - ✓ Estabelecem relacionamentos entre esquemas
  - ✓ Em sistemas de integração de dados são estabelecidos entre o esquema de mediação e as fontes de dados
  - ✓ Em PDMS são estabelecidos entre os pontos
  - ✓ A qualidade dos mapeamentos possui forte influência no resultado das consultas

- Principais formalismos
  - ✓ Global-as-view (GAV)
  - ✓ Local-as-view (LAV)

PDMS – Mapeamentos entre Esquemas

- Mapeamentos em Sistemas de Integração de Dados
  - ✓ Global-as-view
    - Relações no esquema global são descritas como visões sobre a coleção de relações locais
  - ✓ Local-as-view
    - Cada relação em uma visão local é descrita como visão sobre o esquema de mediação





PDMS – Mapeamentos entre Esquemas

- Mapeamentos em Sistemas de Integração de Dados - GAV
- Exemplo
  - ✓ Esquema de Mediação
    - CD(Álbum, Artista, Ano)
    - Contrato(Artista, Ano, Gravadora)
    - Música(Álbum, Nome)
  - ✓ Esquemas Locais
    - BD1(Álbum, Artista, Ano)
    - BD2(Álbum, Artista, Ano, Gravadora)
    - BD3(Álbum, Música)

PDMS – Mapeamentos entre Esquemas

- Mapeamentos em Sistemas de Integração de Dados – GAV

  - ✓ **Mapeamento 1 (CD):**

    CD(Álbum, Artista, Ano) ← BD1(Álbum, Artista, Ano)

    CD(Álbum, Artista, Ano) ← BD2(Álbum, Artista, Ano, Gravadora)

    A relação CD é definida como a união das projeções de BD1 e BD2

    $CD = \pi_{\text{Álbum, Artista, Ano}}(BD1) \cup \pi_{\text{Álbum, Artista, Ano}}(BD2)$

  - ✓ **Mapeamento 2 (Contrato):**

    Contrato(Artista, Ano, Gravadora) ← BD2(Álbum, Artista, Ano, Gravadora)

    $Contrato = \pi_{\text{Artista, Ano, Gravadora}}(BD2)$

  - ✓ **Mapeamento 3 (Música):**

    Música(Álbum, Nome) ← BD1(Álbum, Artista, Ano), BD3(Álbum, Música)

    $Música = \pi_{\text{Álbum,Nome}}(BD1 \bowtie BD3)$

PDMS – Mapeamentos entre Esquemas

- Mapeamentos em Sistemas de Integração de Dados – LAV

  - ✓ Mapeamento 1 (Fonte 1)

    V1(Álbum,Artista, Ano) ← CD(Álbum, Artista, Ano), Contrato(Artista, Ano, EMI), Ano ≥ 2000

    A tabela (V1) armazena os dados dos álbuns que foram produzidos depois do ano 2000 pela gravadora EMI.

  - ✓ Mapeamento 2 (Fonte 2)

    V2(Álbum, Música) ← Música(Álbum, Música)

    A tabela (V2) armazena as músicas e os álbuns

PDMS – Mapeamentos entre Esquemas

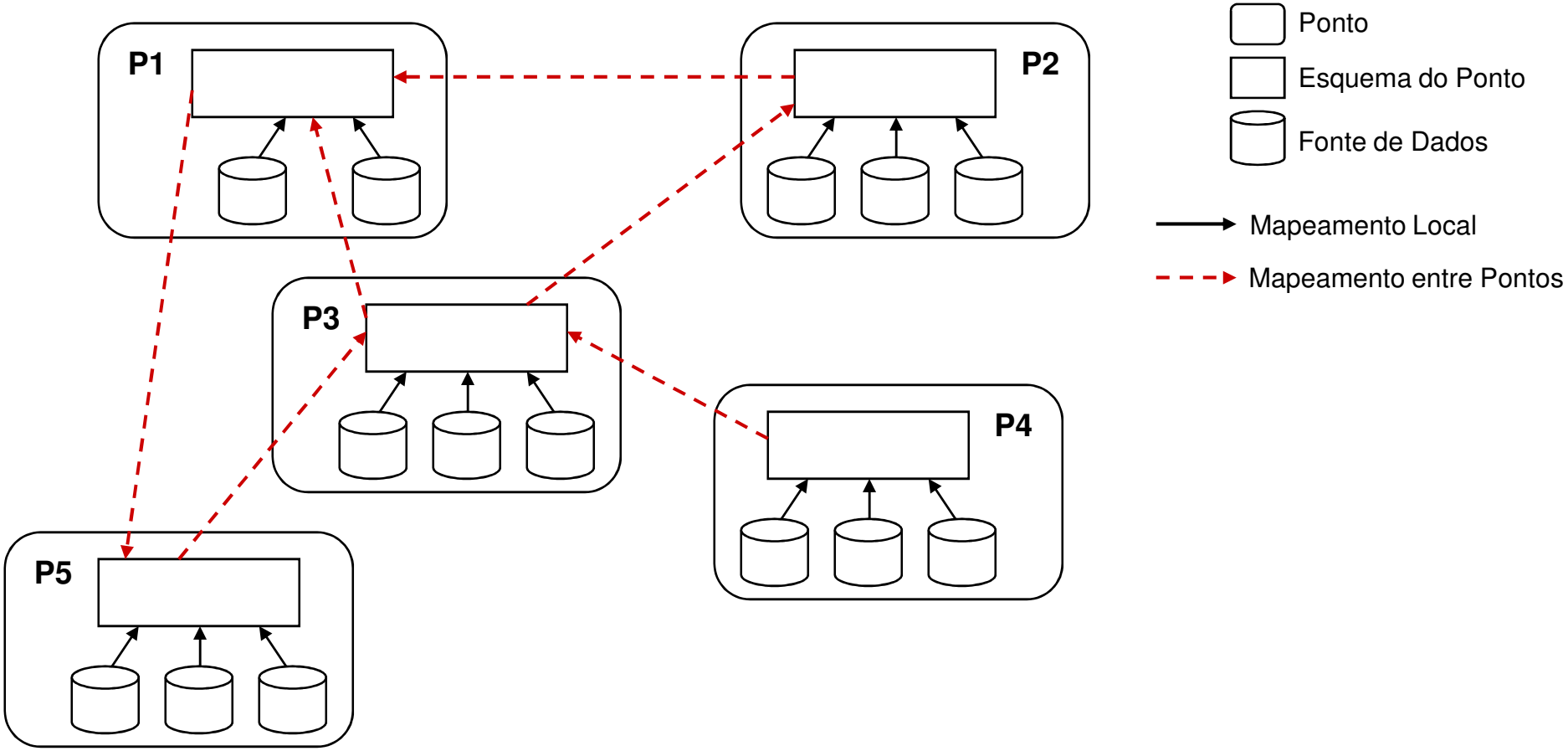
- Mapeamentos em um PDMS
  - ✓ Pontos em um PDMS são relacionados através de mapeamentos
  - ✓ O conjunto de mapeamentos define a rede semântica de um PDMS
  - ✓ A otimização da rede semântica considera
    - Eliminação de mapeamentos redundantes
    - Redução do diâmetro do PDMS (para reduzir a perda de informação na reformulação de consultas)
    - Identificação de pontos semânticos inacessíveis

PDMS – Mapeamentos entre Esquemas

- Mapeamentos em um PDMS
  - ✓ Vantagem de um PDMS
    - Quando um novo ponto entra no sistema será preciso fornecer mapeamentos para um pequeno número de pontos já existentes
    - Em termos de esquema, os pontos existentes devem ser similares ao novo ponto
  - ✓ Dada uma consulta submetida a um ponto, o sistema reformulará a consulta de acordo com os esquemas dos pontos vizinhos

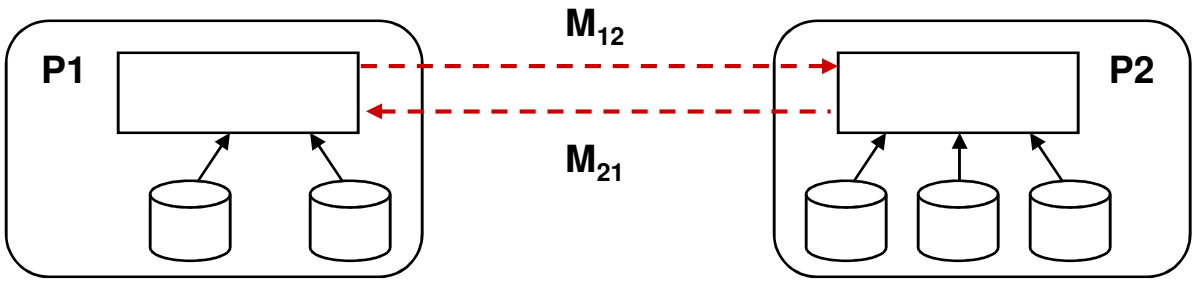
PDMS – Mapeamentos entre Esquemas

- Mapeamentos em um PDMS



PDMS – Mapeamentos entre Esquemas

- Diferentes formalismos podem ser usados para a definição dos mapeamentos (GAV, LAV)
  - ✓ Exemplos
    - Mapeamentos locais: LAV
    - Mapeamentos entre pontos: GAV
- Os mapeamentos podem ser direcionados



PDMS – Mapeamento de Dados

- Mapeamentos entre dados são necessários quando seus valores (formatos) são diferentes
- Tabelas de mapeamento podem ser usadas para especificar a correspondência entre valores de atributos
  - Associam valores dentro de um único domínio ou entre domínios
  - Representam o conhecimento de especialistas
  - Ex.:

Para	Origem
Fortaleza	FOR
Porto Alegre	POA
Recife	REC

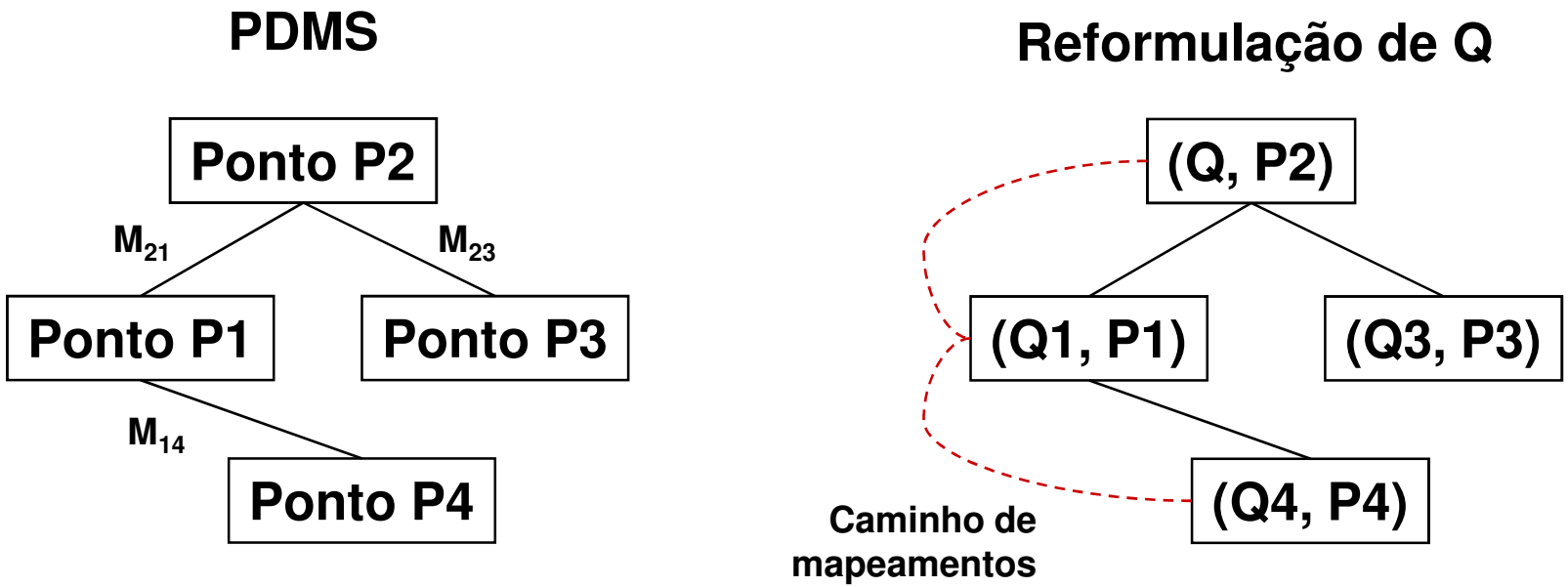
PDMS – Processamento de consultas

- Cada ponto tem um esquema associado
- Pontos são conectados através de “caminhos de mapeamentos”
- Pontos podem ser servidores de dados, mediadores entre pontos e clientes que submetem consultas

PDMS – Processamento de Consultas

1. Uma consulta Q é submetida a um ponto P1 de acordo com o esquema de P1
2. Se P1 possui seus próprios dados, então o PDMS recupera a resposta de Q a partir de P1
3. Em seguida, de acordo com os mapeamentos correspondentes, Q será reformulada para os vizinhos de P1
4. As consultas reformuladas são submetidas aos vizinhos de P1 e assim sucessivamente

PDMS – Processamento de Consultas



PDMS – Processamento de Consultas

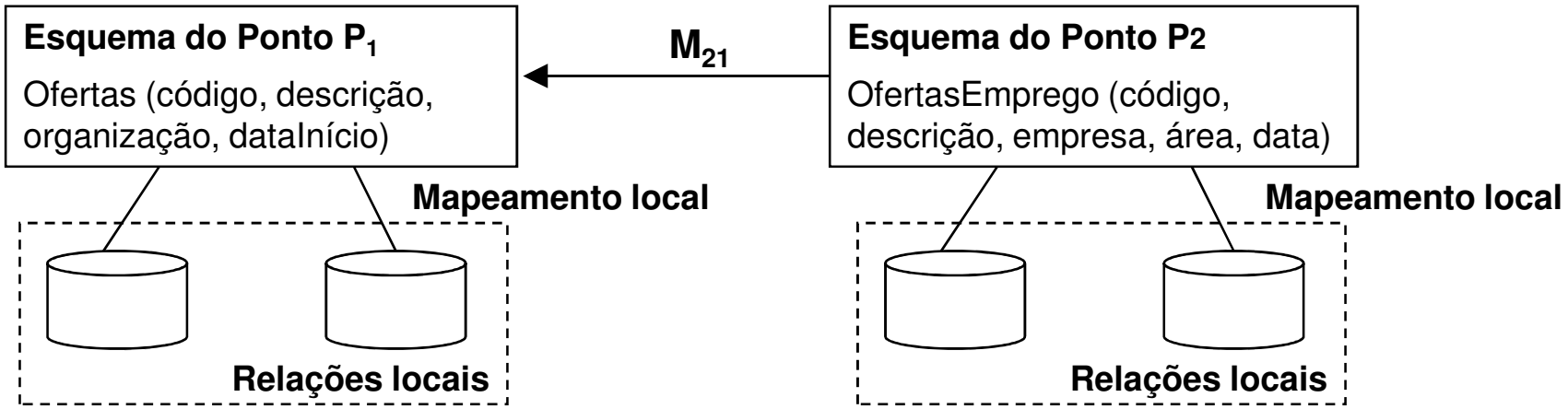
- A técnica de reformulação de consulta dependerá do formalismo para definição dos mapeamentos
  - ✓ *View Unfolding* (Abordagem GAV)
  - ✓ *View Rewriting* (Abordagem LAV)

PDMS – Processamento de Consultas

- Exemplo
  - ✓ Ponto P1
    - Ofertas (código, descrição, organização, dataInício)
  - ✓ Ponto P2
    - OfertasEmprego (código, descrição, empresa, área, data)
  - ✓ Mapeamento entre P2 e P1 (M_{21})
    - P2.OfertasEmprego (código, descrição, empresa, data) $\subseteq$ P1.Ofertas (código, descrição, organização, dataInício)

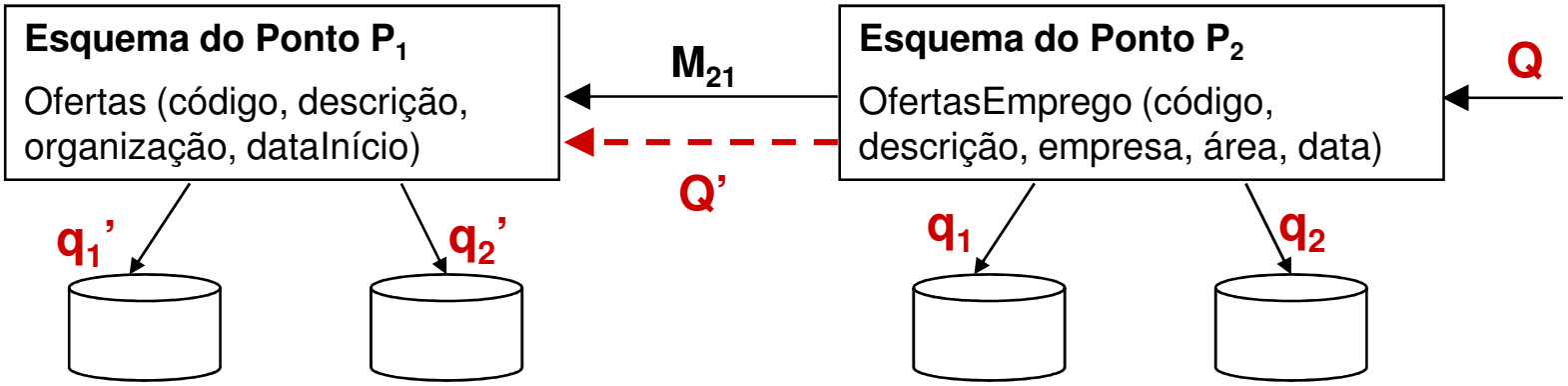
PDMS – Processamento de Consultas

- Exemplo



PDMS – Processamento de Consultas

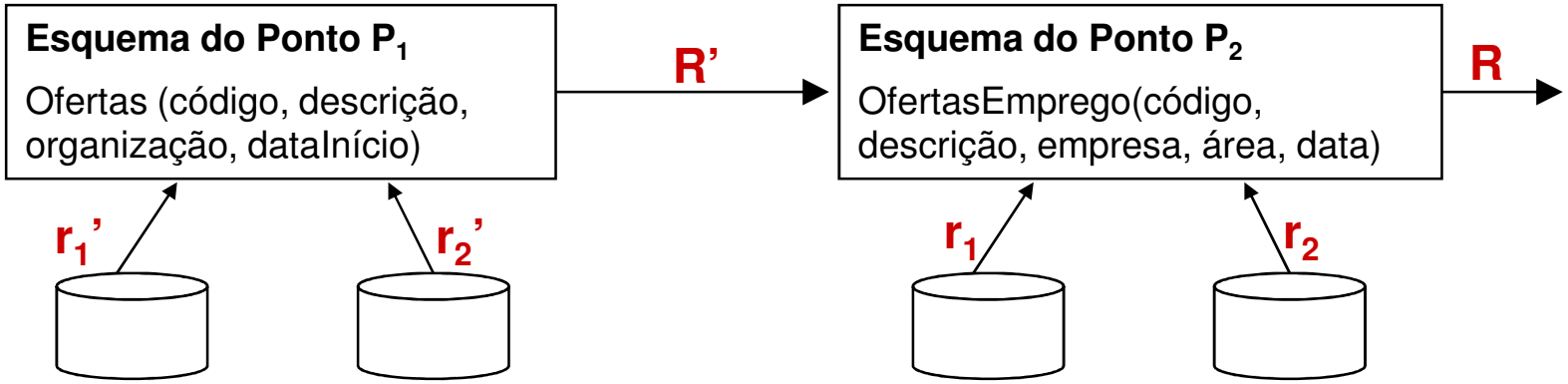
- Exemplo 1
- Consulta submetida em P2:
  - ✓ Q ← OfertasEmprego(código, 'Desenvolvimento', 'Microsoft', 'TI', '01/10/2006')
- Passos de execução da consulta:
  1. P2 usa os mapeamentos locais para reformular Q de acordo com as relações locais de P2
  2. P2 passa Q para P1 de acordo com o mapeamento (M_{21})
  3. A reformulação de consultas é *view unfolding*
  4. P1 usa os mapeamentos locais para reformular Q' de acordo com as relações locais de P1



PDMS – Processamento de Consultas

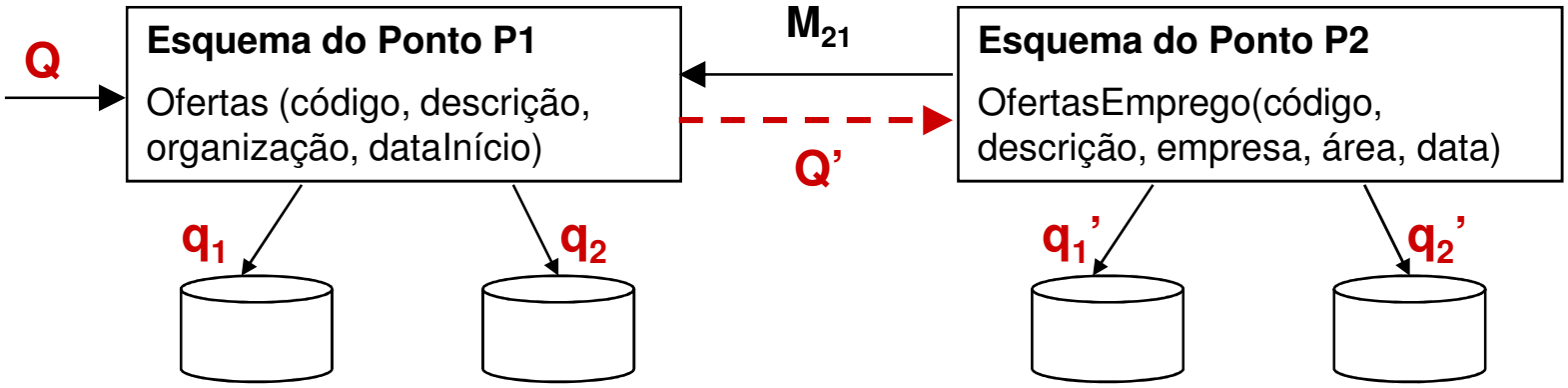
- Exemplo 1
- Passos de execução da consulta:

5. Como não existem mais mapeamentos o processo de reformulação é finalizado
6. Em seguida as consultas são avaliadas nas relações locais, seus resultados integrados e retornados para P2



PDMS – Processamento de Consultas

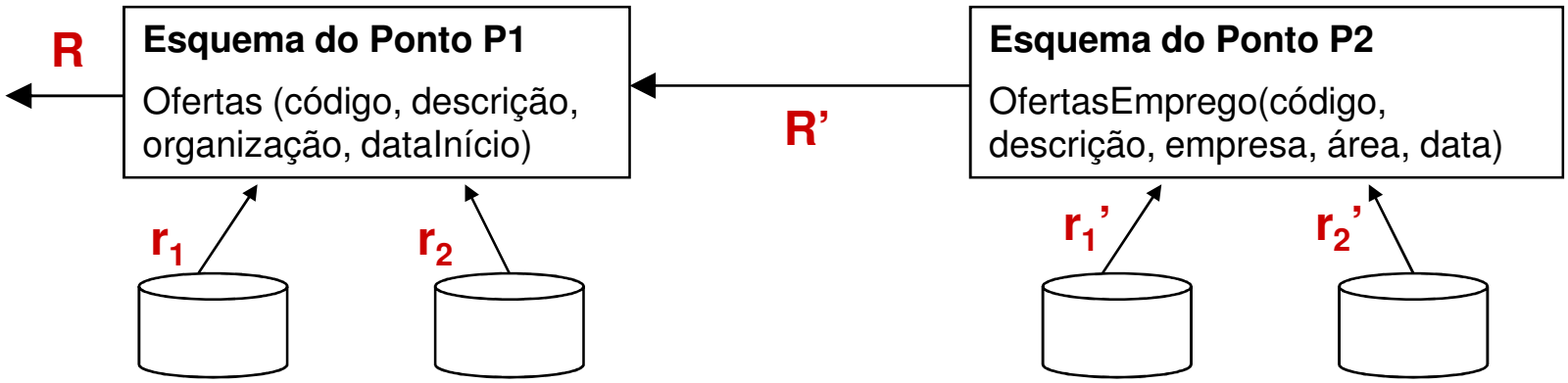
- Exemplo 2
- Consulta submetida em P1:
  - ✓ Q ← Ofertas(código, 'Desenvolvimento', 'Microsoft', '01/10/2006')
- Passos de execução da consulta:
  1. P1 usa os mapeamentos locais para reformular Q de acordo com as relações locais de P1
  2. P1 passa Q para P2 de acordo com o mapeamento (M_{21})
  3. A reformulação de consultas é *view rewriting*
  4. P2 usa os mapeamentos locais para reformular Q' de acordo com as relações locais de P2



PDMS – Processamento de Consultas

- Exemplo 2
- Passos de execução da consulta:

5. Como não existem mais mapeamentos o processo de reformulação é finalizado
6. Em seguida as consultas são avaliadas nas relações locais, seus resultados integrados e retornados para P1

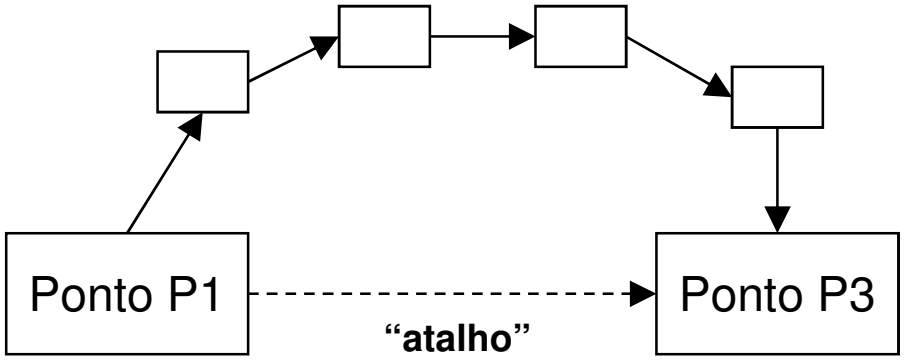


PDMS – Processamento de Consultas

- Em um PDMS um ponto só é acessível através dos mapeamentos
- Com o aumento do número de pontos no sistema o número de esquemas e mapeamentos aumenta
  - ✓ Caminhos de mapeamentos muito extensos (muitas reformulações)
  - ✓ Perda de semântica à medida que as consultas são repassadas (reformuladas)
  - ✓ Restrições de tempo de resposta

PDMS – Processamento de Consultas

- Ontologias podem ser usadas para reduzir o tamanho dos caminhos de mapeamentos
  - ✓ Pontos similares devem ser detectados
  - ✓ “Atalhos” podem ser definidos entre os pontos



PDMS – Processamento de Consultas

- Otimização
  - ✓ Seguir por todos os caminhos leva a ineficiências
    - O algoritmo pode seguir muitos caminhos que resultam em reformulações redundantes (consultas desnecessárias)
    - A aplicação repetitiva de *query unfolding* freqüentemente resulta em consultas redundantes.
  - ✓ Evitar a execução de uma consulta de forma redundante – que retorne um subconjunto de resultados de uma consulta executada anteriormente (*query containment*)

PDMS – Consistência de Dados

- Problema que surge em qualquer cenário onde exista a duplicação de dados
- Principais cenários: *caching* e replicação
- Benefícios da abordagem P2P trazem novos desafios
  - ✓ *Caching*: garantir que os dados da *cache* de um ponto estejam consistentes com os dados dos seus pontos vizinhos
  - ✓ Replicação: a propagação de atualizações nos dados torna-se uma tarefa bastante complexa
    - Grande número de pontos
    - Indisponibilidade dos pontos

PDMS – Localização de Dados

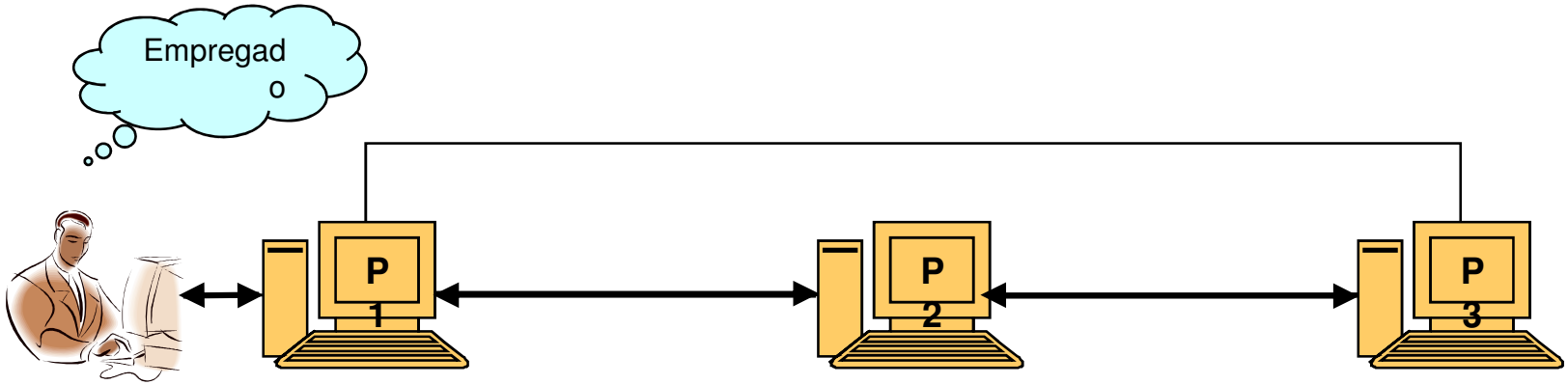
- Difícil de ser prevista
  - ✓ Conectividade dinâmica
  - ✓ Descentralização (conhecimento parcial)
- Topologia Híbrida
  - ✓ Pontos descrevem seus dados durante a conexão
  - ✓ Metadados armazenados em um repositório central

PDMS – Localização de Dados

- Topologia Pura
  - ✓ Não-Estruturada
    - Descoberta de Recursos
      - Transitividade (mapeamentos)
    - Aproximação de Pontos
      - Estatísticas e probabilidade para aproximação de pontos “distantes”
  - ✓ Estruturada
    - Semantic Overlay Networks
- Topologia Super-Peer
  - ✓ Pontos são agrupados em *clusters* (e comunidades)
  - ✓ Utilização de índices (SP-P e SP-SP)

PDMS – Localização de Dados

- Topologia Pura Não-Estruturada



Empregado
Número
…

Funcionário
Código
…

Emp
Código
…

Vizinhos = {P2} P3}

Empregado [P1] = Funcionário [P2]

Empregado [P1] = Emp [P3]

Vizinhos = {P1, P3}

Funcionário [P2] = Empregado [P1]

Funcionário [P2] = Emp [P3]

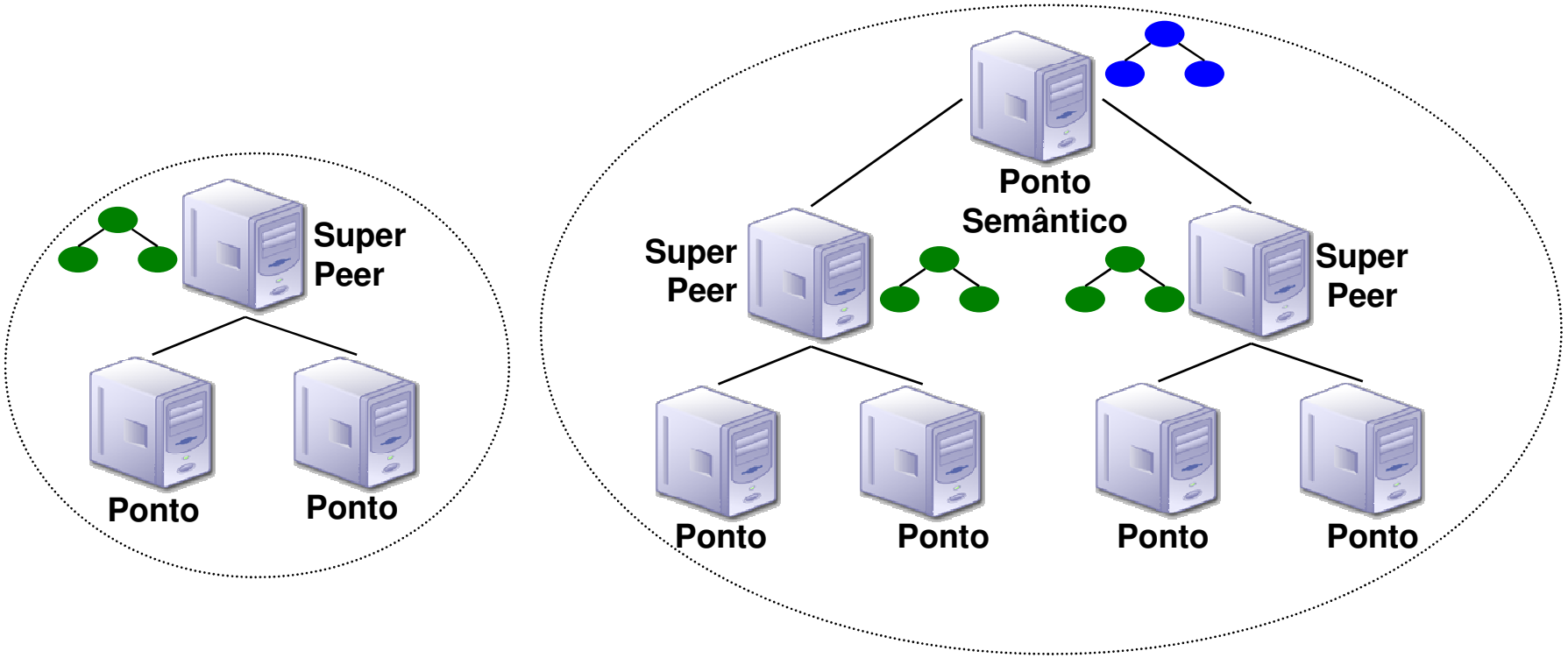
Vizinhos = {P2} P1}

Emp [P3] = Funcionário [P2]

Emp [P3] = Empregado [P1]

PDMS – Localização de Dados

- Topologia Super-Peer
  - ✓ Índices semânticos



(a) Cluster Semântico

(b) Comunidade Semântica

PDMS – Conectividade

- Dinâmica e *ad-hoc*
  - ✓ Dinamismo é menor do que em outros sistemas P2P, e.g. compartilhamento de arquivos
- Importância da alocação eficiente de pontos
  - ✓ Mapeamentos entre esquemas
    - Qualidade dos mapeamentos
  - ✓ Processamento de consultas
    - Escolha dos pontos aptos a responderem consultas
- Topologia Pura
  - ✓ Definição dos vizinhos iniciais
- Topologia *Super-Peer*
  - ✓ Clusters e comunidades semânticas para agrupar

PDMS – Tolerância a Falhas

- Substituição de *super-peers*
  - ✓ Eleição
- Fusão de *clusters*
- Servidores de Backup
  - ✓ Alternativa para evitar políticas de substituição
  - ✓ Selecionado entre os pontos
  - ✓ Cópia periódica dos metatados (*Super-peer* → Servidor de Backup)

PDMS – Qualidade de Dados

- Aspectos de qualidade
  - ✓ Dados das fontes
  - ✓ Mapeamentos entre esquemas
    - Incompletos
    - Incorretos
  - ✓ Plano da Consulta
  - ✓ Resultado das consultas

PDMS – Qualidade de Dados

- Alguns Critérios:
  - ✓ Disponibilidade
  - ✓ Consistência
  - ✓ Relevância
  - ✓ Completude
    - Extensão: conjunto de objetos
    - Intenção: esquema
  - ✓ Cobertura
  - ✓ Densidade

  } **Web**

  - ✓ Tempo de Resposta

Principais PDMS

Principais PDMS

- Piazza, Halevy, Univ. of Washington, 2001
- PeerDB, National Univ. of Singapore, 2002
- XPeer, Univ. Montpellier, 2003
- Hyperion, Univ. Toronto/Univ. Ottawa, 2003
- SPEED, UFPE, Brasil, 2006
- Edutella, Univ. Hannover/Univ. Berlin, entre outras, 2003
- APPA, Univ. de Nantes, 2004
- Rosa-P2P, IME-RJ, Brasil, 2005
- Hyperion, Univ. Toronto/Univ. Ottawa, 2003
- Mapster, Univ. of Cape Town, 2006
- PARIS, Univ. of Calabria/Univ. Bologna, 2006

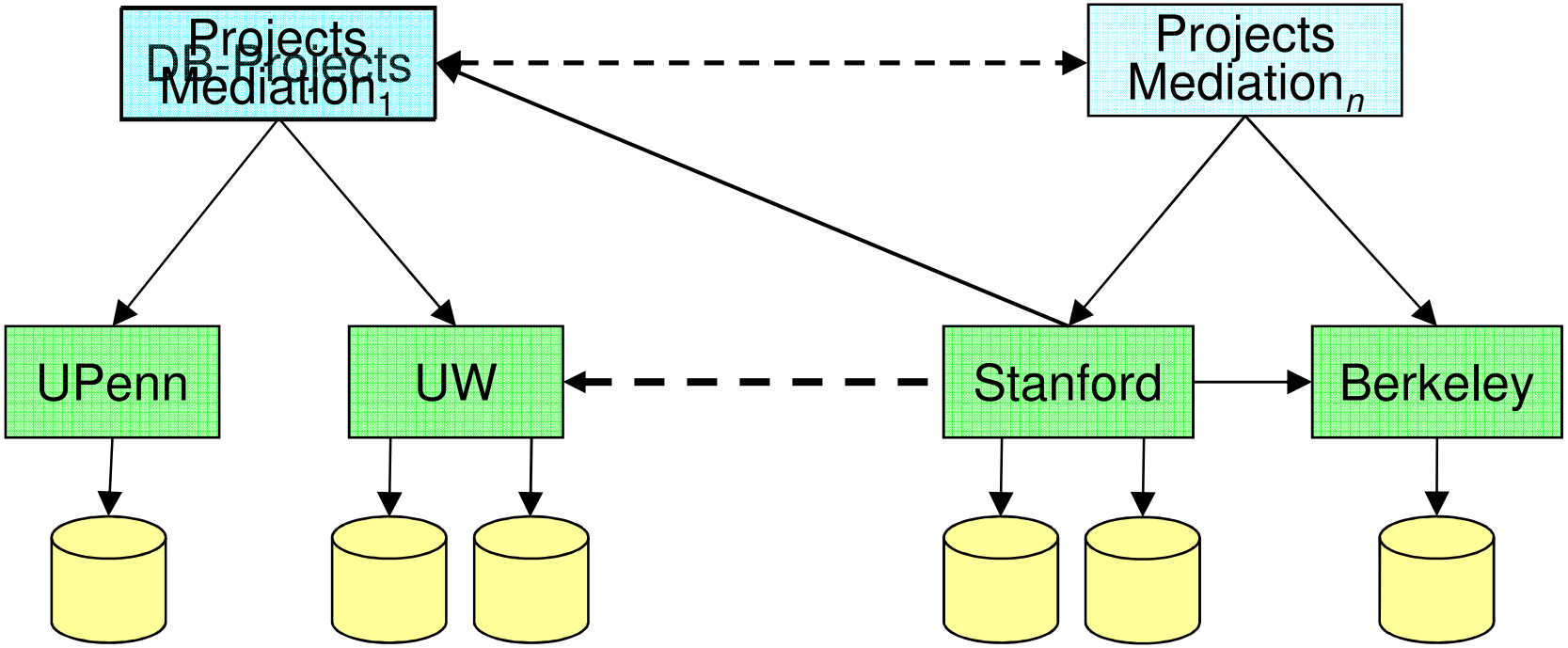
PDMS – Piazza

- **Instituição**
  - ✓ *U. of Washington*
- **Topologia de Rede**
  - ✓ *Pura Não-estruturada*
- **Modelo de Dados**
  - ✓ *Relacional, XML, RDF e DAML-OIL*
- **Mapeamentos entre Esquemas**
  - ✓ *Estabelecidos aos pares entre os pontos*
- **Processamento de Consultas**
  - ✓ *Consulta reescrita com base em mapeamentos*

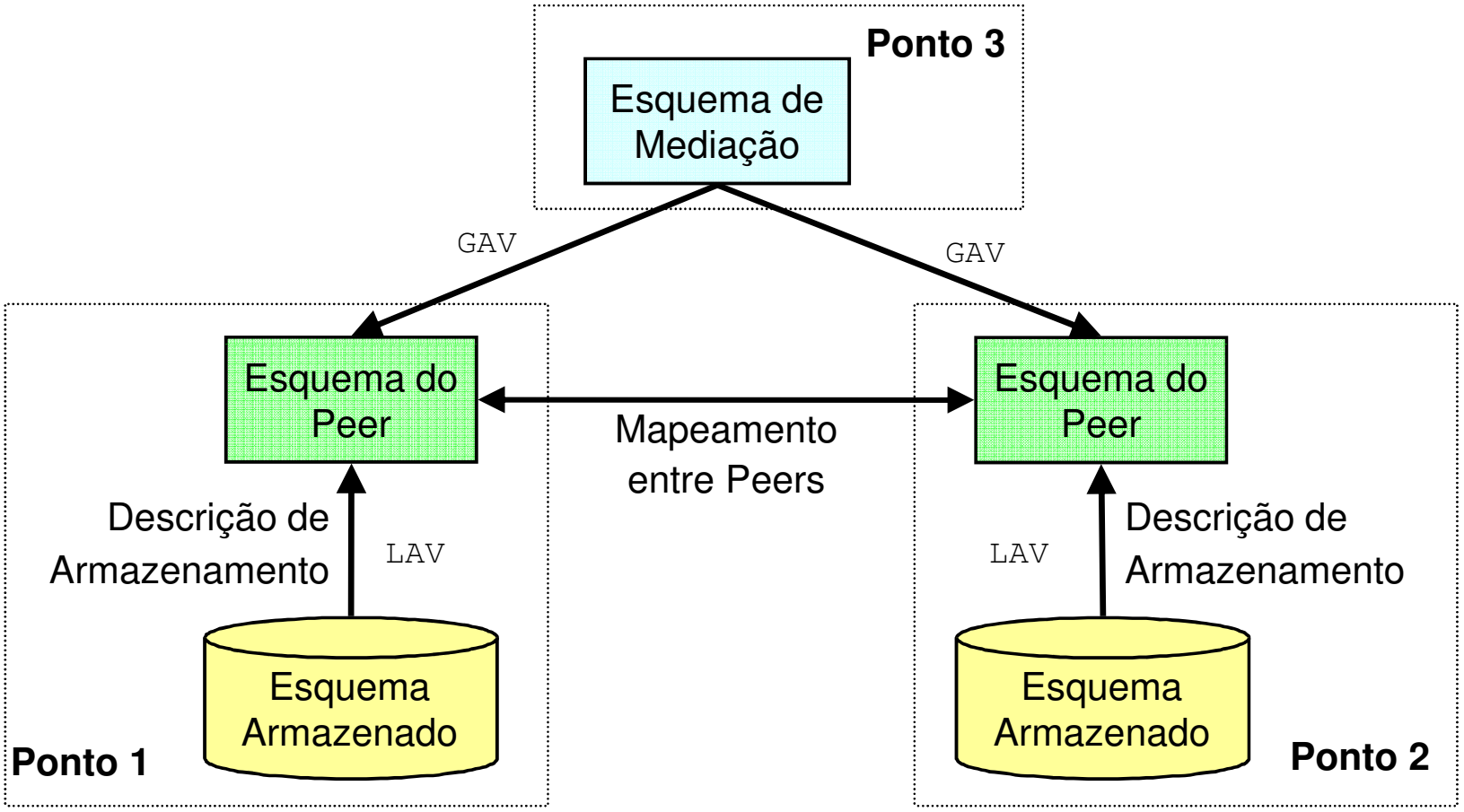
Piazza

- Proposta
  - ✓ Integrar fontes de dados heterogêneas e distribuídas de forma escalonável
- Pontos participantes
  - ✓ Exportam seu esquema, compartilhado-o com os demais pontos
  - ✓ Oferecem visões sobre seus dados
  - ✓ Atuam como mediadores para outros pontos

Piazza – Exemplo



Piazza – Mapeamentos entre Esquemas



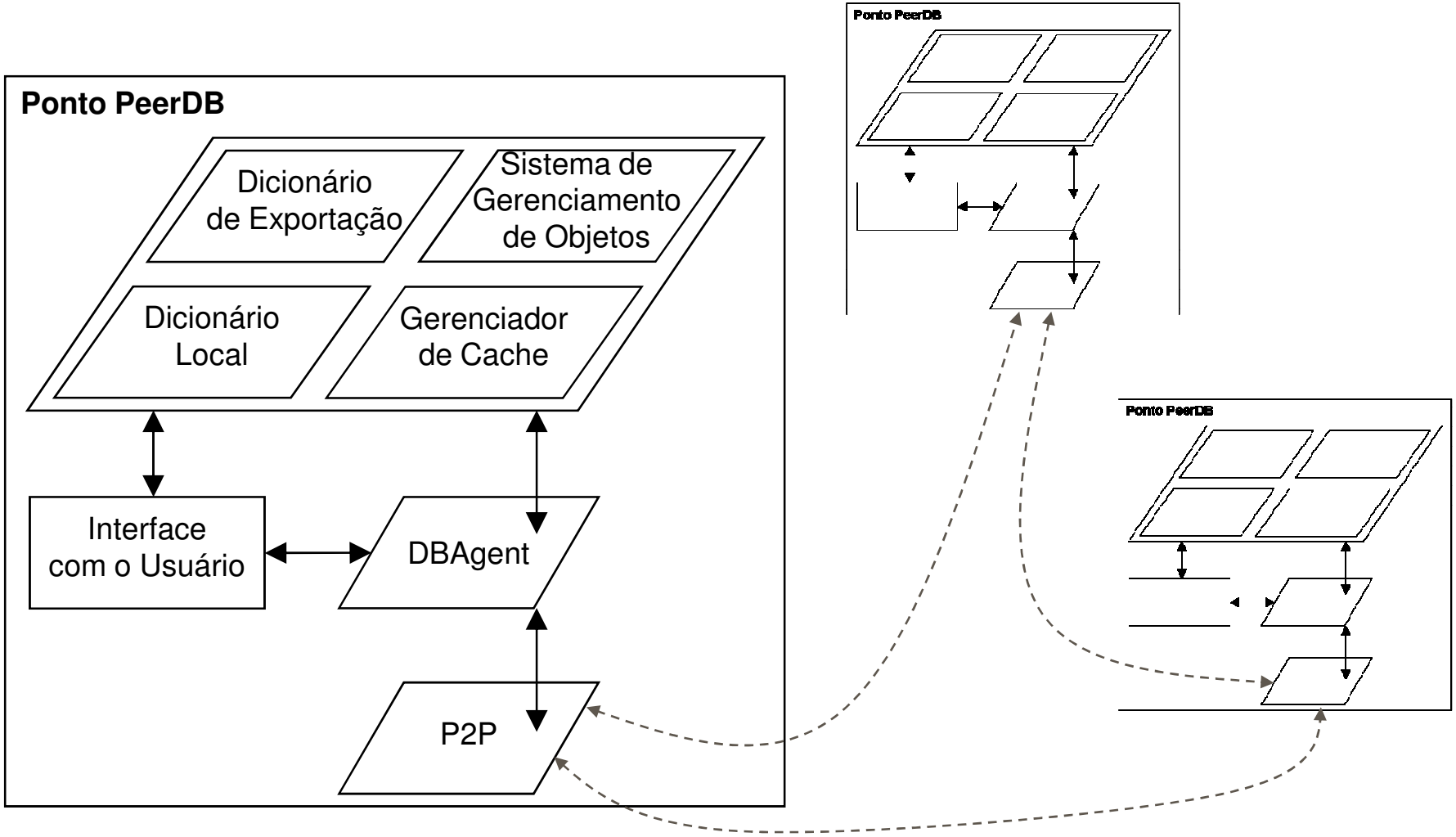
Referências Bibliográficas [Piazza]

- Tatarinov, I., Ives, Z., Madhavan, J., Halevy, A., Suciu, D., Dalvi, N., Dong, X., Kadiyska, Y. Miklau, G., and Mork, P. (2003) “The Piazza Peer Data Management Project”. In Proc. of the ACM SIGMOD Record, Volume 32 , Issue 3, 47-52.
- Halevy, A. Y., Ives, Z. G., Mork, P., and Tatarinov, I. (2003) “Piazza: Data Management Infrastructure for Semantic Web Applications”. In Proc. of the 12th International World Wide Web Conference (WWW’03), Budapest, Hungary.
- Halevy, A., Ives, Z., Mork, P., and Tatarinov, I. (2004) “Piazza: Mediation and Integration Infrastructure for Semantic Web Data”. Journal of Web Semantics, Vol. 1(2), pp. 155-175.
- Halevy, A. Y., Ives, Z., Suciu, D., and Tatarinov, I. (2003) “Schema Mediation in Peer Data Management Systems”. In Proc. of the International Conference on Data Engineering (ICDE’03). IEEE, pp. 505-516.

PDMS – PeerDB

- **Instituição**
  - ✓ *National U. of Singapore*
- **Topologia de Rede**
  - ✓ *Pura Não-estruturada (plataforma BestPeer)*
- **Modelo de Dados**
  - ✓ *Relacional*
- **Mapeamentos entre Esquemas**
  - ✓ *Relation-matching (Função de Similaridade)*
  - ✓ *Palavras-chave*
- **Processamento de Consultas**
  - ✓ *Agentes Inteligentes*
  - ✓ *Dependência e interatividade com o usuário*

PeerDB – Arquitetura



PeerDB – Arquitetura

1. *Camada P2P*
   - ✓ Oferece serviços para compartilhamento de dados e descoberta de recursos na rede
2. *Camada de Agentes Inteligentes*
   - ✓ *Master agents* e *worker agents*
3. *Camada de Gerenciamento de Dados*
   - ✓ Possibilita armazenamento e processamento dos dados

PeerDB – Camada de Gerenciamento de Dados

- *Sistema de Gerenciamento de Objetos*
  - ✓ Possibilita armazenamento, manipulação e recuperação dos dados no ponto
- *Dicionário Local*
  - ✓ Contém metadados sobre o esquema armazenado no ponto
- *Dicionário de Exportação*
  - ✓ Contém metadados das tabelas compartilhadas
  - ✓ Subconjunto dos metadados do *Local Dictionary*
- *Gerenciador de Cache*
  - ✓ Determina políticas de *caching* e substituição de dados e metadados

PeerDB – Camada de Agentes

- *Sistema de Agentes de Banco de Dados (DBAgent)*
  - ✓ Oferece o ambiente para os agentes trabalharem
  - ✓ Cada ponto possui um *Master Agent* que gerencia as consultas dos usuários
  - ✓ Responsável por “clonar” e despachar *Worker Agents* para os pontos vizinhos, receber respostas e apresentá-las ao usuário
  - ✓ Monitora as estatísticas e gerencia as políticas de reconfiguração de pontos na rede

PeerDB – Mapeamentos entre Esquemas

- Determinados em tempo de execução de consulta
- Abordagem baseada em *Information Retrieval* (IR)
  - ✓ Metadados gerados para tabelas/atributos
    - Palavras-chave + descrições
    - Servem como sinônimos
  - ✓ Agentes são enviados aos pontos em busca de associações candidatas
  - ✓ A busca baseia-se na comparação de *keywords* e *relation-matchings*

PeerDB – Mapeamentos entre Esquemas

Peer	Names	Keywords
P1	Kinases	protein, human
	SeqID	key, identifier, ID
	length	length
	proteinSeq	sequence, protein sequence
P2	Protein	protein, annexin, zebrafish
	SeqNo	number, identifier
	len	length
	sequence	sequence
P3	ProteinKLen	protein, kinases, length
	ID	number, identifier
	seqLength	length
	ProteinKSeq	protein, sequence
	ID	number, identifier
	sequence	sequence
P4	Protein	protein, kinases, annexin, . . .
	name	name
	char	characteristics, features, functions

- Pontos: P1, P2, P3 e P4
- Consulta Q é feita em P1

```
SELECT SeqId, ProteinSeq
FROM   Kinases
WHERE  length > 30
```

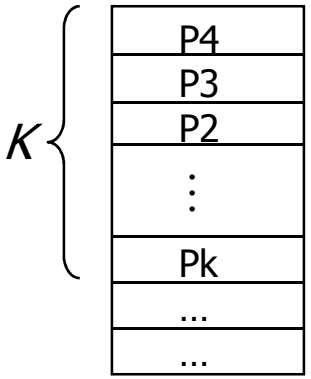
- Estratégia *relation-matchings*
  - ✓ P2, P3 e P4 são relevantes para Q
  - ✓ P4 só tem associação com nome da tabela (menor relevância que P2 e P3)
  - ✓ Semanticamente, dados de P2 não são de interesse de P1 (não são *Kinases*)
  - ✓ Usuário vai decidir

PeerDB – Processamento de Consultas

- Consulta ocorre em duas fases distintas
  1. Aplicação da estratégia de *relation-matching* para localizar tabelas candidatas
     - Tabelas candidatas são retornadas para o usuário
     - Ponto atualiza suas estatísticas para futuras buscas
     - Objetivos
       - Eliminar problema de homonímia
       - Minimizar o uso banda de rede
  2. Execução de consultas
     - Consultas são enviadas aos pontos determinados pelo usuário
     - Respostas são retornadas (e colocadas em *cache*)

PeerDB – Reconfiguração de Pontos

- Políticas para definição de vizinhança
  - ✓ Número de *keywords* em comum
  - ✓ Quantidade de dados retornados pelos pontos vizinhos
  - ✓ *Stack distance*
    - Pontos que tenham respondido consultas recentemente, devem ficar no topo da pilha
    - Os *k* primeiros permanecem diretamente conectados



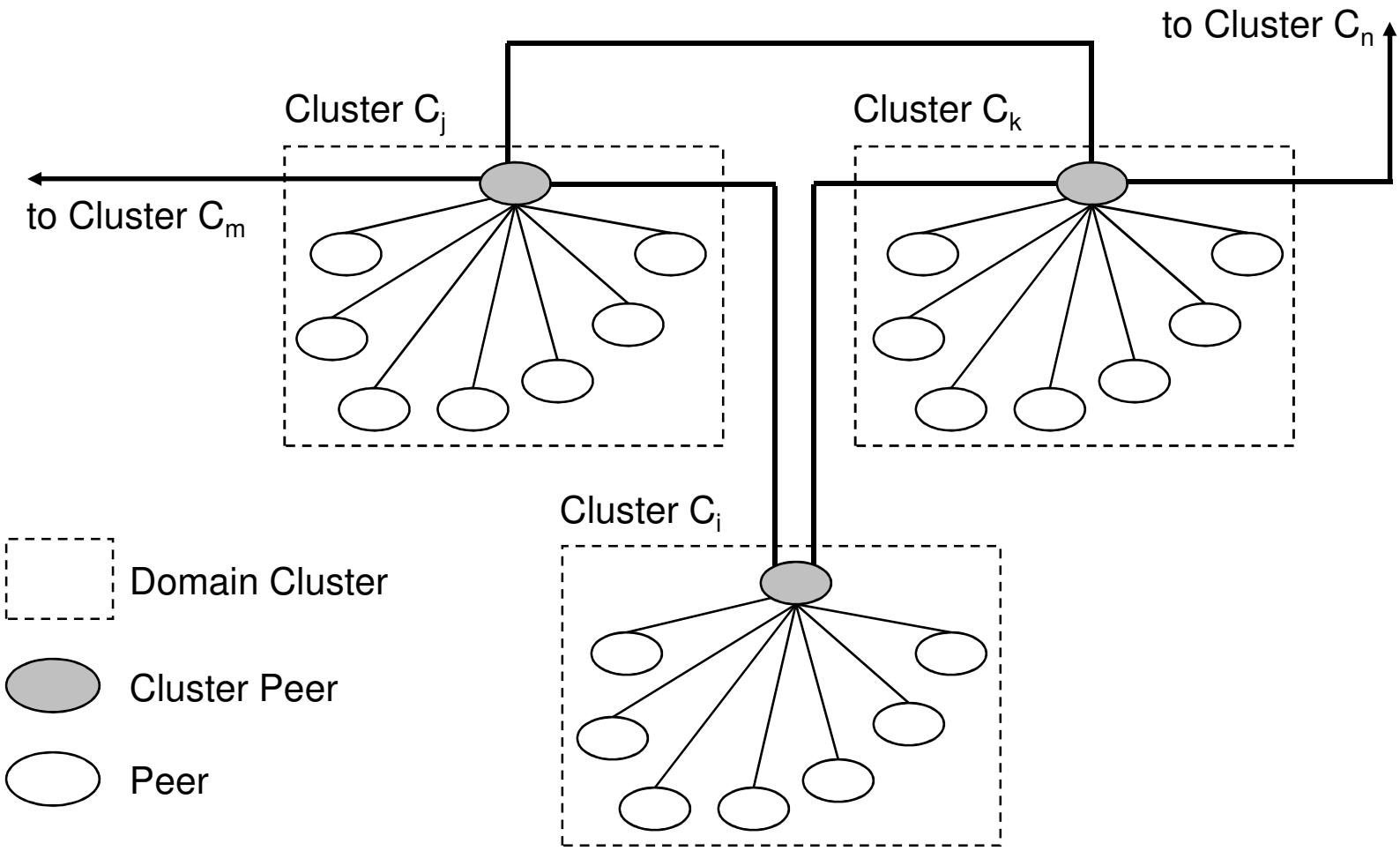
Referências Bibliográficas [PeerDB]

- Ooi, B. C., Shu, Y., and Tan, K.-L. (2003) “Relational Data Sharing in Peer-based Data Management Systems”. In ACM SIGMOD Record, 32 (3), 59-64.
- Ooi, B. C., Shu, Y., and Tan, K.-L. (2003) “DB-Enabled Peers for Managing Distributed Data”. In Proc. of the 5th Asia Pacific Web Conference (APWeb'03). Xian, China. LNCS 2642, 10-21.
- Ng, W. S. Ooi, B. C., Tan, K.-L., and Zhou, A. (2003) “PeerDB: A P2P-based System for Distributed Data Sharing”. In Proc. of the International Conference on Data Engineering 2003 (ICDE'03). 633-644.
- Ng, W. S., Ooi, B. C., and Tan, K.-L. (2002) “Bestpeer: A selfconfigurable peer-to-peer system”. In Proc. of the 18th International Conference on Data Engineering, San Jose, USA.
- Hellerstein, J., Franklin, M., Chandrasekaran, S., Deshpande, A., Hildrum, K., Madden, S., Raman, V., and Shah, M. (2000) “Adaptive query processing: Technology in evolution. In IEEE Data Engineering, 23(2).
- Papadimos V., and Maier, D. (2002) “Mutated query plans”. Information and Software Technology, 44(4):197–206, 2002.

XPeer

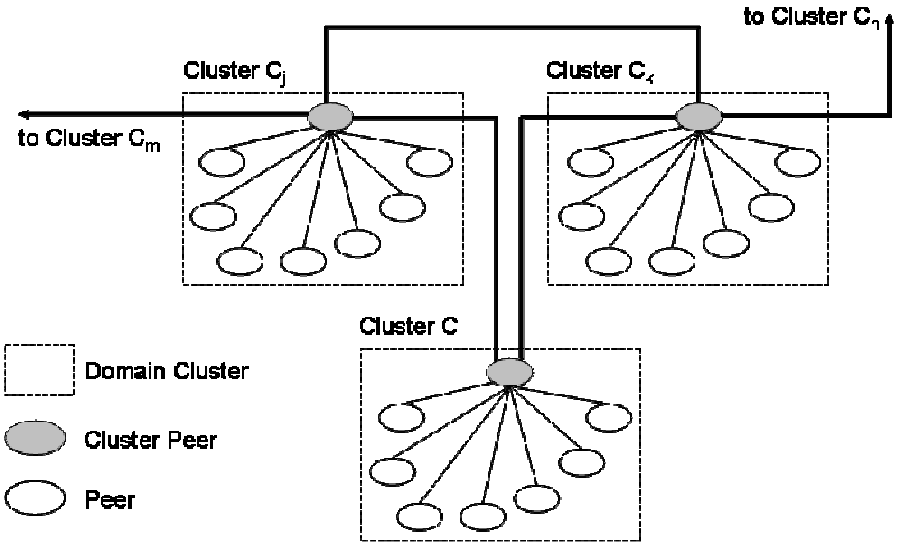
- **Instituição**
  - ✓ *Université de Montpellier*
- **Topologia de Rede**
  - ✓ *Super-Peer*
- **Modelo de Dados**
  - ✓ *XML*
- **Mapeamentos entre Esquemas**
  - ✓ *LAV e GAV*
- **Processamento de Consultas**
  - ✓ *Consultas são reescritas pelos cluster peers*

XPeer – Arquitetura



XPeer – Arquitetura

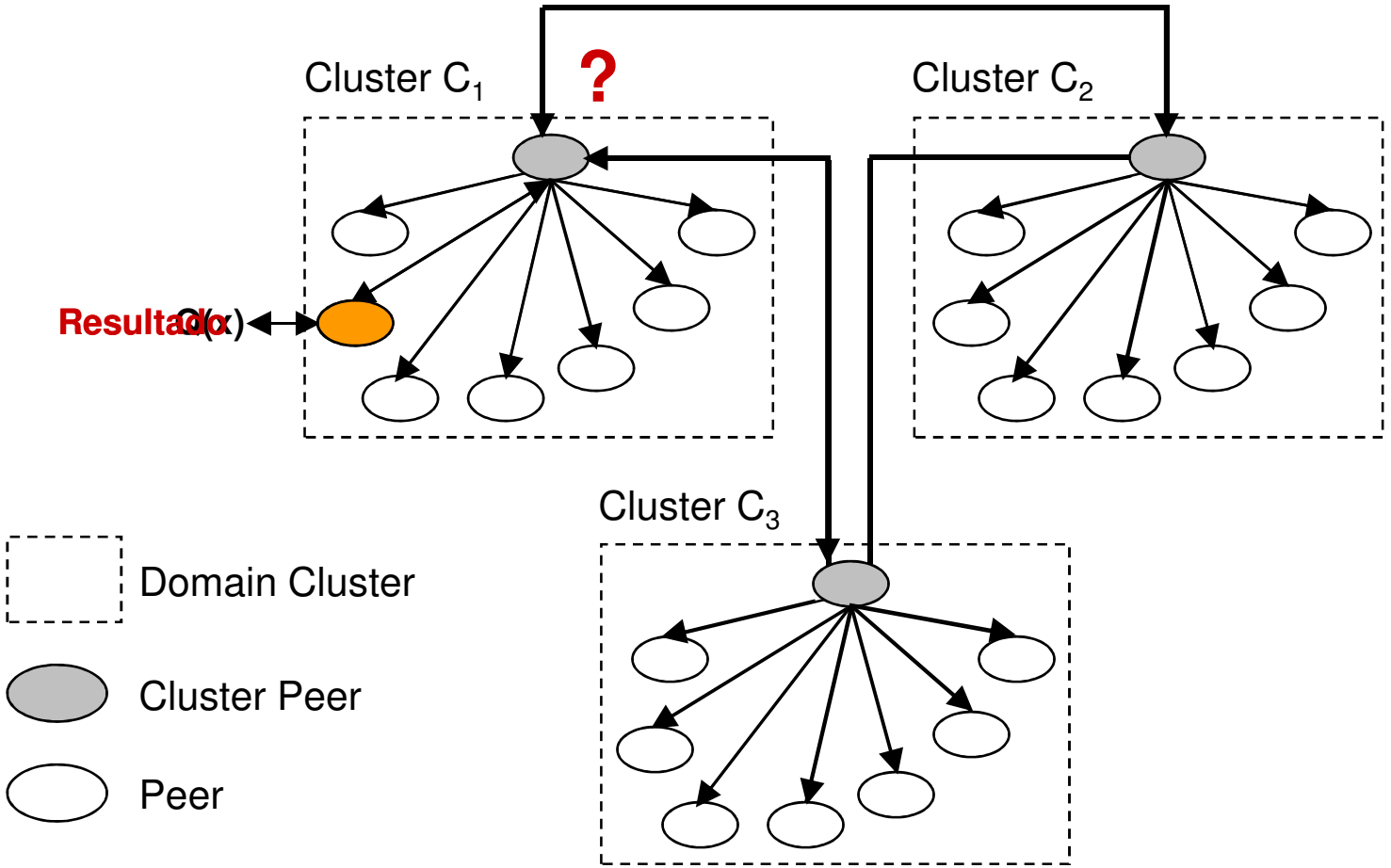
- Domínio
  - ✓ Conjunto de *clusters* compartilhando uma mesma categoria de informação (e.g. medicina)
- *Cluster Peer*
  - ✓ Contém um esquema de mediação
- *Domain Peer*
  - ✓ Gerencia vários *cluster peers*
  - ✓ Ponto de entrada no domínio
- *Global Peer*
  - ✓ Gerencia vários *domain peers*
  - ✓ Ponto de entrada no sistema



XPeer – Conectividade

- Conexão e desconexão de um peer
  - ✓ Afeta apenas o *cluster peer* local
- *Cluster peer*
  - ✓ Conexão
    - Registro no domínio adequado, passando sua identificação, localização e esquema de mediação
    - *Domain peer* repassa os dados para os outros *cluster peers*, que retornam suas identificações
  - ✓ Desconexão/Falha
    - *Domain peer* elege um dos pontos do mesmo *cluster* para ser o novo *cluster-peer*

Processamento de Consultas



5) Resultados recebidos, integrados e devolvidos ao peer solicitante

Referências Bibliográficas [Xpeer]

- Bellahsène R., King, N. and Roantree M. (2004) “Services for Large Scale P2P Networks”. European Research Consortium for Informatics and Mathematics News Journal (ERCIM'04)
- Bellahsène, Z., Roantree, M. (2004) “Querying Distributed Data in a Super-Peer Based Architecture”. In Proc. of the 15th International Conference on Database and Expert Systems Applications (DEXA'04), Volume 3180 of LNCS, pp. 296-305, Zaragoza, Spain.
- Bellahsène, Z., Lazanitis, C., Mc. Brien, P., and Rizopoulos, N. (2006) “iXPeer: Implementing layers of abstraction in P2P Schema Mapping using AutoMed”. In Proc. of the 2nd Workshop on Innovations in Web Infrastructure. Co-located with the 15th International World-Wide Web Conference, Edinburgh, Scotland.
- Tranier, J., Baraër, R., Bellahsène, Z., and Teisseire, M. (2004) “Where's Charlie: Family-Based Heuristics for Peer-to-Peer Schema Integration”. In Proc. of the 8th International Database Engineering and Applications Symposium (IDEAS 2004), Coimbra, Portugal. 227-235.

Hyperion

- **Instituições**
  - ✓ *U. of Toronto / U. of Ottawa / U. of Edinburgh / U. of Trento*
- **Topologia da Rede**
  - ✓ *Pura Não-estruturada*
- **Modelo de Dados**
  - ✓ *Relacional*
- **Mapeamentos entre Esquemas**
  - ✓ *Tabelas de Mapeamento e Expressões de Mapeamento*
- **Processamento de Consultas**
  - ✓ *Reescrita de consultas através das Tabelas e Expressões de Mapeamento*

Hyperion

- Proposta
  - ✓ SGBD coordenam, em tempo de execução, consultas, atualizações e compartilhamento de dados
- Coordenação de dados significa…
  - ✓ Reconciliação e integração de dados em tempo de consulta
  - ✓ Manutenção da consistência dos dados dos pontos

Hyperion – Exemplo

- Companhias de aviação
  - ✓ Alpha-Air e Beta-Air possuem sua própria clientela e calendário de vôos
- Compartilhamento de dados sobre vôos e passageiros
- Coordenação de vôos para destinos comuns

AA_Passenger

pid	name
1	Renee
2	Verena
3	Iluju

AA_Flight

pid	fno	meal
1	AA210	Meat
2	AA378	Veg.

AA_Ticket

fno	date	dest	sold	cap
AA210	01/05/03	L.A.	120	256
AA341	01/15/03	N.Y.	160	160
AA378	01/21/03	S.F.	90	124

(a) Instâncias do Banco de Dados Alpha-Air

BA_Passenger

pid	name
1	John
2	Renee

BA_Fleet

aid	type	capacity
B-1	Boeing 747	340
B-2	Boeing 737	130
B-3	Boeing 737	107

BA_Flight

fno	date	to	sold	aid
BA1023	01/07/03	LAX	67	B-3
BA1078	01/15/03	JFK	118	B-2
BA1109	01/15/03	ATH	164	B-1

BA_Reserve

pid	fno
1	BA1023
2	BA1078
2	BA1109

(b) Instâncias do Banco de Dados Beta-Air

Hyperion – Mapeamentos entre Esquemas

- Compartilhamento de dados depende do estabelecimento de *acquaintances*
- *Acquaintance*
  - ✓ É um relacionamento estabelecido aos pares entre os pontos
  - ✓ Produz um conjunto de tabelas e expressões de mapeamentos
  - ✓ Permite que os dados sejam compartilhados e coordenados pelos pontos

Hyperion – Mapeamentos entre Esquemas

- Expressões de Mapeamento
  - ✓ Impõem restrições no compartilhamento de dados
- Exemplo

```
AA Passenger(p, n) ⊇ BA Passenger(p, n)
```

"Qualquer consulta pelos passageiros de A deve ser executada em B"

Hyperion – Mapeamentos entre Esquemas

- Tabelas de Mapeamento
  - ✓ Definem correspondências entre valores armazenados em diferentes pontos
  - ✓ Metadados ao nível de dados
- Exemplos

dest	to
N.Y.	JFK
N.Y.	LGA
L.A.	LAX
L.A.	ONT
Athens	ATH

(a)

fno_{AA}	fno_{BA}
AA210	BA1023
AA341	BA1078
AA341	BA1080

(b)

date_{AA}	date_{BA}
$\mathcal{X}$	$\mathcal{X}$

(c)

Hyperion – Processamento de Consultas

- Consulta Q a Alpha-Air (datas dos vôos para LA)

Q:
```
SELECT    date
FROM      AA_Flight
WHERE     dest = "L.A."
```

- Q → Q' (uso de tabelas e expressões de mapeamento)

Q':
```
SELECT    date
FROM      BA_Flight
WHERE     to = "LAX" OR to = "ONT"
```

- Consulta a Beta-Air referindo-se a datas de vôos para LAX recupera os vôos para LA de Alpha-Air

Hyperion – Coordenação de Dados

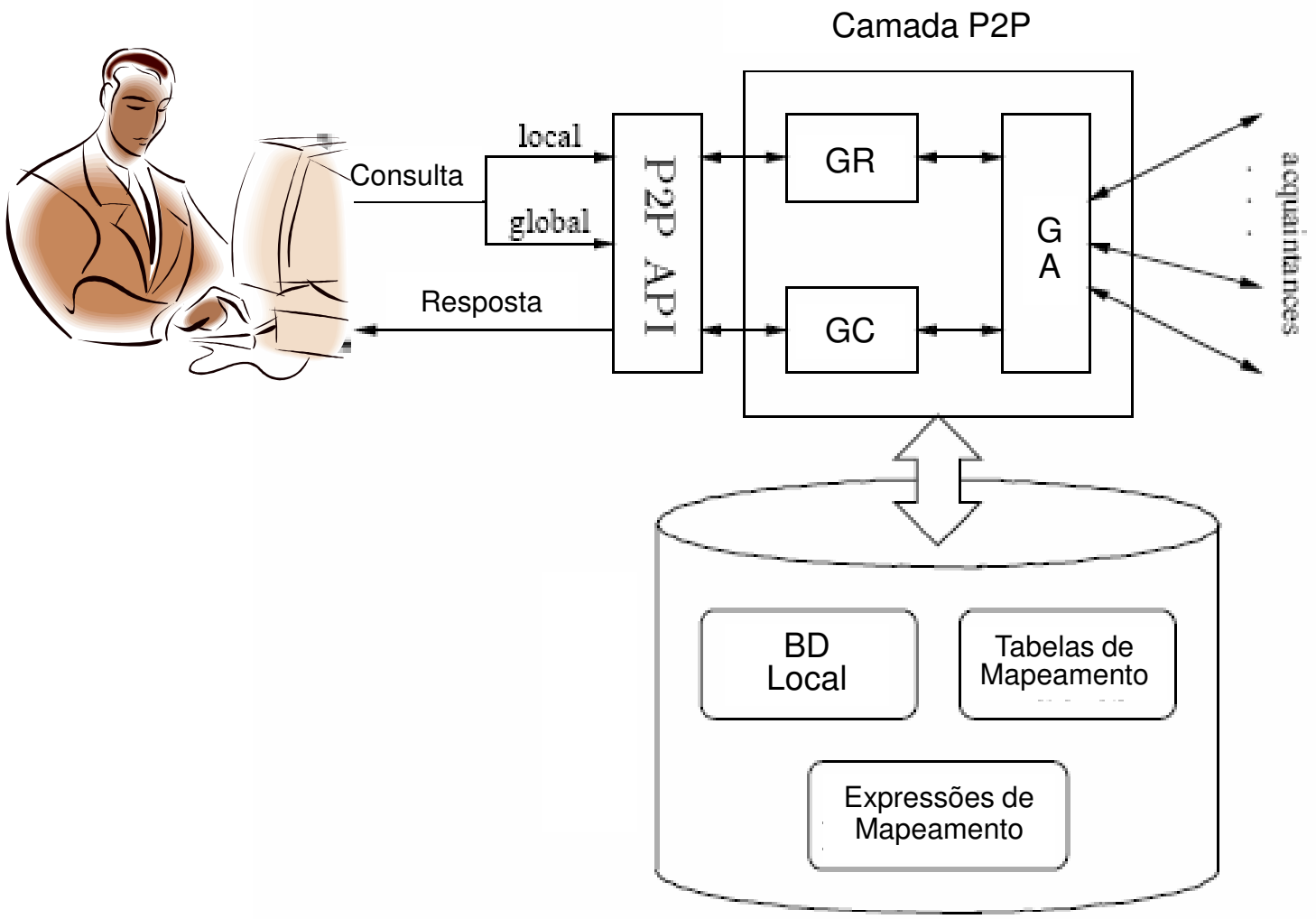
- Expressões de mapeamento permitem a criação de regras de coordenação
- Expressão de Mapeamento

```
AA Passenger(p, n) ⊇ BA Passenger(p, n)
```

- Regra de Coordenação (ECA)

```
CREATE TRIGGER passengerInsertion
AFTER INSERT ON BA_Passenger
       REFERENCING NEW AS NewPass
FOR EACH ROW
BEGIN
    INSERT INTO AA_Passenger VALUES NewPass IN Alpha-
Air DB;
END;
```

Hyperion – Arquitetura



Hyperion – Arquitetura

- Interface (P2P API)
  - ✓ Usuários podem especificar consultas locais e/ou globais
- Camada P2P
  - ✓ Camada de comunicação
- Banco de Dados Local
  - ✓ Armazena os dados locais e as tabelas e expressões de mapeamento
- Gerenciador de *Acquaintances* (GA)
  - ✓ Estabelece (modo semi-automático) e mantém as *acquaintances*

Hyperion – Arquitetura

- Gerenciador de Consultas (GC)
  - ✓ Responsável por reescrever consultas
  - ✓ Utiliza os serviços oferecidos pelo AM
- Gerenciador de Regras (GR)
  - ✓ Responsável pelo gerenciamento das regras de coordenação entre os pontos

Referências Bibliográficas [Hyperion]

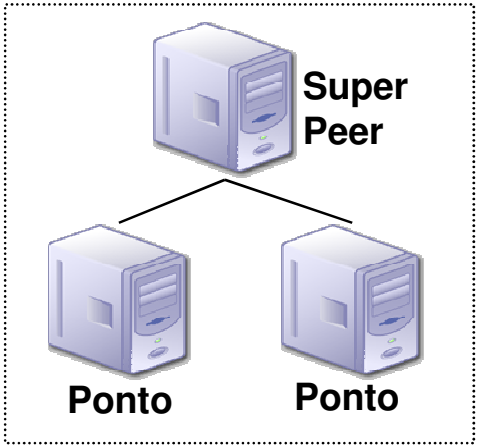
- Bernstein, P., Giunchiglia, F., Kementsietsidis, A., Mylopoulos, J., Serafini, L., and Zaihrayeu, I. (2002) “Data Management for Peer-to-Peer Computing: A Vision”. In Proc. of the 5th International Workshop on the Web and Databases (WebDB)
- Arenas, M., Kantere, V., Kementsietsidis, A., Kiringa, I., Miller, R. J., and Mylopoulos, J. (2003) “The Hyperion Project: From Data Integration to Data Coordination”. In SIGMOD Record, Special Issue on Peer-to-Peer Data Management, 32(3):53-58.
- Kementsietsidis, A., Arenas, M., and Miller, R. J. (2003) “Managing Data Mappings in the Hyperion Project”. In Proc. of the International Conference on Data Engineering (ICDE’03)
- Kementsietsidis, A., Arenas, M., and Miller, R. J. (2003) “Mapping Data in Peer-to-peer Systems: Semantics and Algorithmic Issues”. In ACM SIGMOD.
- Kantere, V., Kiringa, I., Mylopoulos, J., Kementsietsidis, A., and Arenas, M. (2003) “Coordinating Peer Databases using ECA Rules”. In International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P’03)
- Kementsietsidis, A., and Arenas, M. (2004) “Data Sharing through Query Translation in Autonomous Sources”. In International Conference on Very Large Databases (VLDB’04)

SPEED (Semantic PEEr Data Management System)

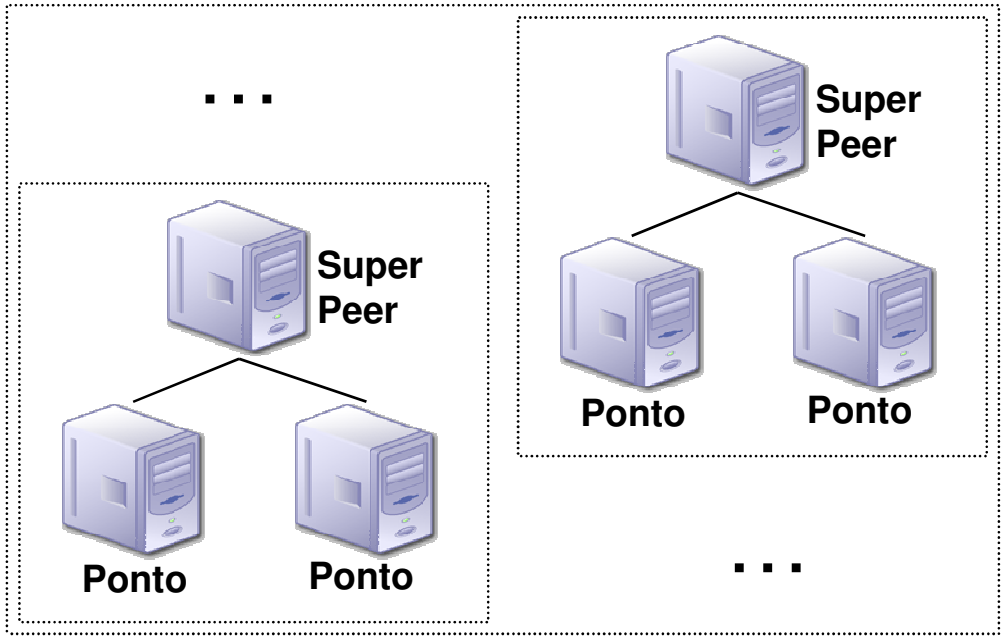
- **Instituição**
  - ✓ *UFPE*
- **Topologia de Rede**
  - ✓ *Super-Peer*
- **Modelo de Dados**
  - ✓ *Relacional, XML*
- **Mapeamentos entre Esquemas**
  - ✓ *Estabelecidos aos pares*
- **Processamento de Consultas**
  - ✓ *Índice semântico*

SPEED

- Proposta
  - ✓ Compartilhamento de dados entre fontes heterogêneas, utilizando uma abordagem baseada em ontologias
- Agrupamento de pontos

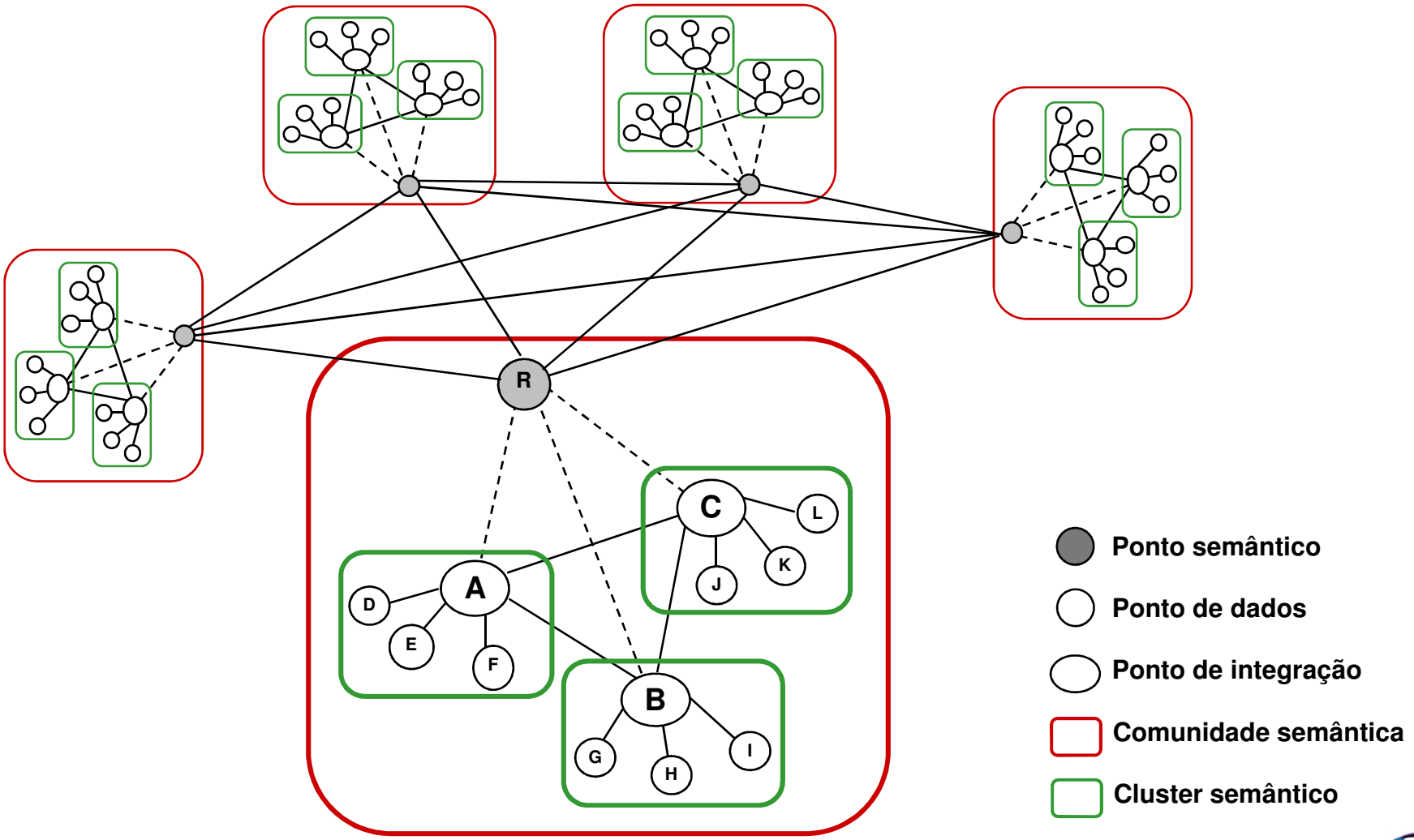


(a) Cluster Semântico



(b) Comunidade Semântica

SPEED


R
A
B
C
D
E
F
G
H
I
J
K
L
Ponto semântico
Ponto de dados
Ponto de integração
Comunidade semântica
Cluster semântico

SPEED – Ponto Semântico

- Ponto de entrada na comunidade
- Comunica-se com pontos semânticos e pontos de integração
- Disponibiliza uma ontologia de domínio
  - ✓ Alocação de pontos de dados e de integração
  - ✓ Enriquecimento semântico de esquemas

SPEED – Ponto de Integração

- Único por cluster semântico
- Responsabilidades
  - ✓ Processamento de consultas
  - ✓ Integração de dados
- São pontos de dados com alta disponibilidade e poder de processamento

SPEED – Ponto de Dados

- Sistema de informação que compartilha dados com outros pontos de dados
- Agrupados de acordo com o domínio semântico
- Possibilidade de se tornar um ponto de integração

Comparativo entre PDMS

PDMS	Topologia	Modelo de Dados	Mapeamentos entre Esquemas	Processamento de Consultas
Piazza	Pura Não-estruturada	Relacional, XML, RDF	Estabelecidos aos pares	Consultas reescritas com base em mapeamentos
PeerDB	Pura Não-estruturada	Relacional	Relation-matching; Palavras-chave	Agentes Inteligentes; Forte interatividade com o usuário
Xpeer	Super-Peer	XML	LAV e GAV	Consultas reescritas pelos cluster peers
Hyperion	Pura Não-estruturada	Relacional	Tabelas e expressões de mapeamento	Reescrita com base nas tabelas e expressões de mapeamento
SPEED	Super-Peer	XML	Mapeamentos PI←→PD e PI←→PI	Índice semântico

Desafios

Localização dos Dados

- Balancear:

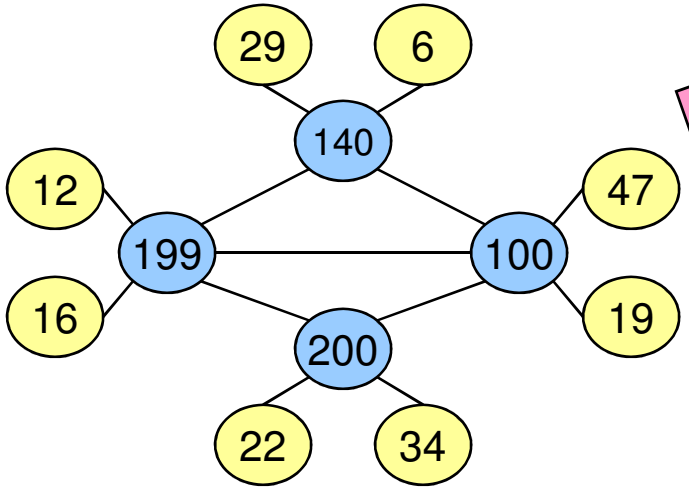
  localização dos dados X tempo de resposta

- Diversificar o tipo de acesso: não apenas baseado em chaves

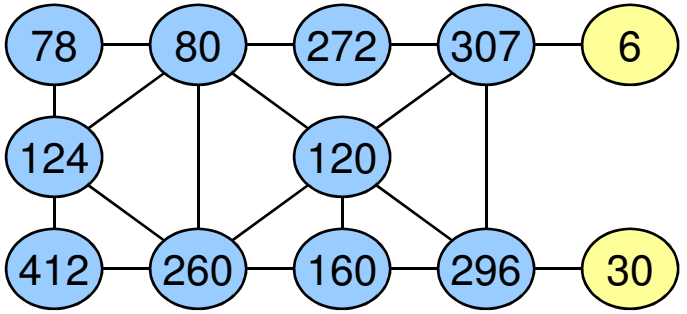
- Estratégia para otimização:
  - ✓ armazenar em locais estratégicos
  - ✓ usar para melhorar o desempenho de consultas

Localização dos Dados

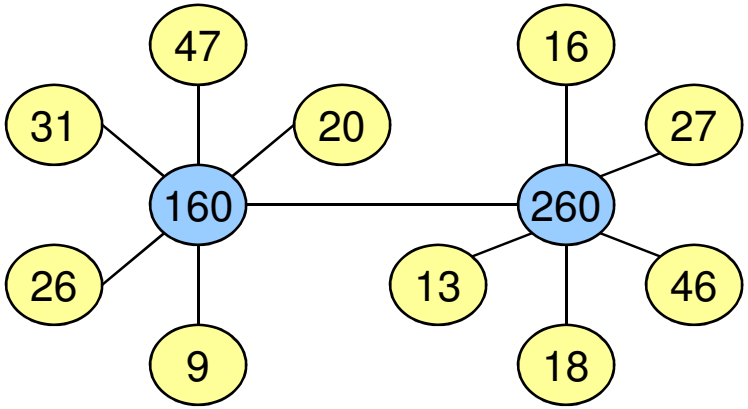
Balanceamento de Clusters



Configuração Ideal



Muitos Super-Peers



Poucos Super-Peers

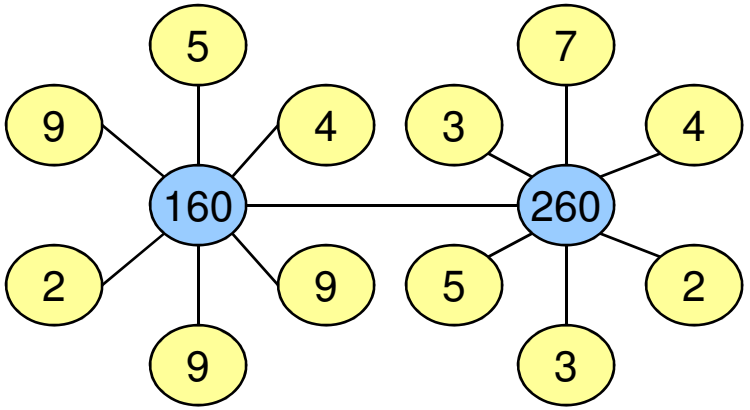
Super-Peer

Peer

Critério: largura de banda >= 50KB/s

Localização dos Dados

- Critérios para Balanceamento
  - ✓ Disponibilidade
  - ✓ Características físicas (CPU, memória, …)
  - ✓ Tempo de resposta
- Critérios podem ser incompatíveis



Disponibilidade

x

Largura de Banda

- Problema
  - ✓ Pontos podem ser promovidos mesmo sem condições físicas ideais

Integração de Dados

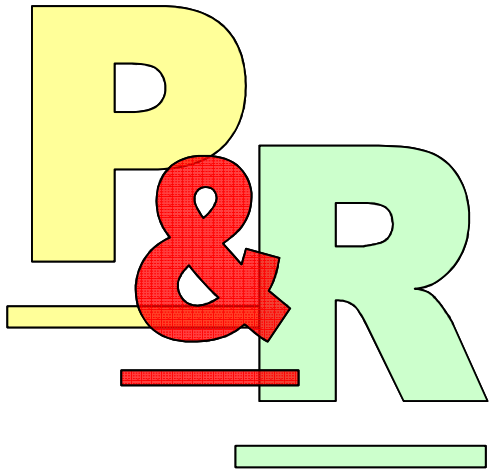
- Mapeamento entre esquemas
  - ✓ Linguagens para especificação de mapeamentos
  - ✓ Facilidade no gerenciamento de mapeamentos: modelo de representação
  - ✓ Mapeamentos semânticos: continuam um desafio
- Eficiência e otimização
  - ✓ Evitar caminhos redundantes (escolha dos melhores)
  - ✓ Propagar atualizações de forma eficiente
  - ✓ Composição de mapeamentos sem perda: se A → B e B → C então A → C

Processamento de Consultas

- Linguagem de consulta apropriada
- Mapeamentos estruturais e semânticos adequados
- Planos de consultas: consultas mutantes
  - ✓ Direcionar para a localização correta, ou
  - ✓ Fornecer os dados diretamente (se sabe como obtê-los)
- Índices de roteamento X DHT
- Regras de tradução de dados e dependências semânticas
- Avaliação da qualidade dos resultados das consultas

Consistência dos Dados

- Oferecer um grau de consistência razoável
- Políticas de replicação e *caching* (*Freshness*)
  - ✓ Manter a continuidade dos dados (dinamismo dos pontos)
  - ✓ Conservar réplicas consistentes
- Tolerância a falhas
- Segurança



acs@cin.ufpe.br – Ana Carolina Salgado

cesp@cin.ufpe.br – Carlos Eduardo Pires

bernafarias@lia.ufc.br – Bernadette Lóscio

www.cin.ufpe.br/~cesp/tutorialSBBD

Referências Bibliográficas [PDMS]

- Sung, L. G. A., Ahmed, N., Blanco, R., Li, H, Soliman, M. A., and Hadaller, D. (2005) “A Survey of Data Management in Peer-to-Peer Systems”. School of Computer Science, University of Waterloo.
- Valduriez, P., Pacitti, E. (2004) “Data Management in Large Scale P2P Systems”, Proc of VECPAR
- Halevy, H.Y., Ives, Z.G. (2005) “Schema Mediation for Large-Scale Semantic Data Sharing”, VLDB Journal, V.14, N.1

Referências Bibliográficas [Integração de Dados]

- WIEDERHOLD, G. (1992) “Mediators in the Architecture of Future Information Systems”. IEEE Computer, 25(3):38-49.
- Lenzerini, M. (2004) “Principles of Peer-to-Peer Data Integration”, citeseer.ist.psu.edu/lenzerini04principles.html
- Ruzzi, M. (2004) “Data Integration: state of the art, new issues and research plan, Dipartimento di Informatica e Sistemistica.

Referências Bibliográficas [P2P]

- Walkerdine, J., Melville, L. and Sommerville, I. (2002) "Dependability Properties of P2P Architectures". In Proc. of the 2nd International Conference on Peer to Peer Computing (P2P'02), Linkoping, Sweden, pp.173-174.
- Milojicic, D., Kalogeraki, V., Lukose, R., Nagaraja, K., Pruyne, J., Richard, B., Rollins, S., Xu, Z. (2002) "Peer-to-Peer Computing". Technical Report HPL-2002-57, HP Labs.
- Rocha, J., Domingues, M., Callado, A., Souto, E., Silvestre, G., Kamienski, C., Sadok, D. (2004) "Peer-to-Peer: Computação Colaborativa na Internet". Mini-curso 22º Simpósio Brasileiro de Redes de Computadores. Gramado, RS.

Referências Bibliográficas [Conectividade]

- Li, J. and Vuong, S. (2004) “An Efficient Clustered Architecture for P2P Networks”. In Proc. of the 18th International Conference on Advanced Information Networking and Applications (AINA'04) Vol. 1, p. 278
- Zhuang, Z., Liu, Y., and Xiao, L. (2004) “Dynamic Layer Management in Super-Peer Architectures”. In Proc. of the International Conference on Parallel Processing (ICPP'04) - Volume 00 p. 29-36

Referências Bibliográficas [Topologias]

- Backx, P., Wauters, T., Dhoedt, B., and Demeester, P. (2002) “A Comparison of Peer-to-Peer Architectures”. In Proc. of the Eurescom Summit.
- Fiorano. (2003) “Super-Peer Architectures for Distributed Computing”. White Paper, Fiorano Software, Inc.

Referências Bibliográficas [GridxP2P]

- Talia, D. and Trunfio, P. (2003) “Toward a Synergy Between P2P and Grids”. In IEEE Internet Computing, Vol. 07, No. 4, pp. 96, 94-95.
- Foster, I. (2002) “The Grid: A New Infrastructure for 21st Century Science”. Physics Today, 55 (2). 42-47.
- Foster, I. and Iamnitchi, A. (2003) “On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing”. In Proc. of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03), Berkeley, USA.

Referências Bibliográficas [SON]

- Garcia-Molina, H., Crespo, A. (2002) “Semantic Overlay Networks for P2P Systems”. Technical Report, Stanford University.
- Vazirgiannis, M., Nørvåg, K., and Doulkeridis, C. (2006) “Peer-to-Peer Clustering for Semantic Overlay Network Generation”. In Proc. of the 6th International Workshop on Pattern Recognition in Information Systems (PRIS-2006, co-located with ICEIS 2006), Paphos, Cyprus.

SON: Funcionamento