Universidade Federal de Pernambuco – UFPE
Centro de Informática – CIn
Pós-graduação em Ciência da Computação

Balanceamento de *Clusters* em Sistemas Gerenciadores de Dados em Redes P2P baseados em Ontologias

Por:
Edemberg Rocha da Silva

Exame de Qualificação e Proposta de Tese

Profa. Dra. Ana Carolina Salgado
Orientadora

Recife, dezembro de 2011

Resumo

Os constantes avanços tecnológicos ocorridos na área de banco de dados propiciaram o surgimento de sistemas que promovem a integração de dados distribuídos em fontes diversas, proporcionando aos seus usuários total transparência em seu uso. Nos últimos anos, a evolução dos bancos de dados foi aliada às redes P2P, com o objetivo de facilitar este compartilhamento de dados. Dessa união, surgiram os sistemas gerenciadores de dados em ambientes P2P (*Peer Data Management Systems*- PDMS) com o intuito de prover aos usuários os benefícios de sistemas P2P e as facilidades pertencentes aos SGBD tais como linguagens de consultas, modelos semânticos e facilidade de uso. Os PDMS visam prover, aos usuários, mais facilidades de consulta, ao mesmo tempo em que eles tenham visões simplificadas dos dados que se encontram, na verdade, localizados em fontes distribuídas, heterogêneas e autônomas.

O sistema SPEED (*Semantic PEEr-to-Peer Data Management System*) é um PDMS baseado em ontologias, capaz de realizar a formação de *clusters* com pontos que possuem similaridades semânticas, de acordo com os esquemas exportados pelas fontes. Essa formação permite facilitar o estabelecimento dos mapeamentos semânticos entre os pontos e, conseqüentemente, melhorar o processamento de consulta sobre um grande volume de fontes de dados. É sobre a infraestrutura do sistema SPEED que será desenvolvido este trabalho.

Assim como o SPEED, outros PDMS têm adotado a estratégia de *cluster* para agrupar pontos de acordo com o conteúdo compartilhado. Esse agrupamento, por exemplo, pode auxiliar o processamento de consultas no sentido de que as consultas são direcionadas para um número menor de pontos, porém mais relevantes para a consulta. No entanto, um problema de desbalanceamento de carga poderá ocorrer devido à heterogeneidade dos pontos (como larguras de banda, processadores, capacidade de armazenamentos diferentes, entre outros) e do total de suas requisições.

Dentro desse escopo, o objetivo deste trabalho é propor um processo de balanceamento de carga para o SPEED, onde este sistema possa detectar e resolver, sem intervenção humana, estados de sobrecarga em seus *clusters*. Esse processo poderá desencadear operações que possam alterar as estruturas dos *clusters* (dividindo-os ou unindo-os). Mas, essas alterações deverão respeitar os critérios de similaridade semântica que norteiam a conectividade dos pontos aos seus respectivos *clusters*.

Abstract

The increasing development of database technologies has led to the creation of data integration systems which provide users with a transparent way of use. In this sense, databases have been combined with P2P networks in order to improve data sharing. Peer Data Management Systems (PDMS) have emerged as a result of putting together such technologies aiming to provide users with the benefits of P2P systems and facilities owned by DBMS such as query languages, semantic models and ease of use. PDMS aim to provide users with query facilities, while they are able to work with simplified data views that abstract data located in the overall distributed, heterogeneous and autonomous existing sources.

The SPEED system (Semantic Peer-to-Peer Data Management System) is an ontology-based PDMS. It is able to form clusters with peers that have some semantic similarity, according to their domain interest and exported schemas. This capability enables the establishment of semantic mappings among the peers and therefore improve query processing on a large volume of data sources. SPEED is the underlying infrastructure over which this work will be developed.

Like SPEED, other PDMS have adopted peer clustering strategies according to the shared common interest or content. Such kind of clustering strategy, for example, can help query processing in such a way that queries are sent to a smaller number of peers. Meanwhile, queries may be sent to more relevant peers that can better answer a given query. However, a problem of load imbalance can occur due to the peers' heterogeneity (such as bandwidth, processing capacity, storage capacity, among others) and the total of query requests.

In this light, the purpose of this work is to propose a process of load balancing for the SPEED system. In such proposal, the system will be able to detect and resolve overload states and conflicts in existing clusters without human intervention. This process will be able to trigger operations that can change clusters formation (e.g., splitting or joining clusters). Nevertheless, these changes must comply with some chosen criteria regarding the existing semantic similarity among peers connectivity and their respective clusters.

Sumário

Lista de Figuras

Lista de Quadros e Tabelas

Capítulo 1

Introdução

Neste capítulo, introduzimos o contexto onde está inserido este trabalho de qualificação e proposta de tese. Apresentamos a motivação para essa tese, definimos o problema, seus objetivos, o escopo e a estrutura organizacional deste trabalho.

1.1. Motivação

Como um meio de prover uma infra-estrutura para um compartilhamento de recurso na *Web*, surgem os Sistemas *Peer-to-Peer* (P2P). Esses sistemas partem do princípio que um controle centralizado inexiste e que cada *peer* (ponto) atua como cliente e/ou servidor. Esses sistemas são caracterizados por possuírem um controle descentralizado, robustez, escalabilidade, estabilidade, entre outras. Diante dessas características, vem crescendo o interesse de pesquisadores sobre esses sistemas. A natureza dos sistemas P2P oferece uma área vasta para pesquisas, não só no tocante ao compartilhamento de conteúdo, mas também em outros aspectos que incluem infra-estrutura, busca, colaboração, roteamento, balanceamento de carga, segurança, tolerância a falhas, dentre outros.

Em alguns ambientes P2P, cada ponto participante representa uma fonte de dados a ser compartilhada. Por convenção deste trabalho, chamaremos esse tipo de participante de ponto de dados. Mas, essas fontes de dados, além de estarem distribuídas nesses ambientes, elas são, em sua maioria, heterogêneas. Os dados podem estar em páginas HTML, XML, bancos de dados, planilhas, dentre outros. Diante deste ambiente heterogêneo, um sistema que possa integrar dados nesse ambiente P2P se faz necessário. Essa necessidade já era prevista em (HALEVY *et al.* 2006). Nesse trabalho, os autores apontam para um tipo de sistema de integração de dados, para esse tipo de ambiente, chamado de PDMS (*Peer Data Management System*).

Alguns PDMS têm adotado a formação de *clusters*, para melhor organizar seus pontos de dados e obter melhor respostas a seus serviços. Porém, nem sempre a adoção de *clusters* implica na melhoria dos serviços oferecidos pelo sistema. A formação de *cluster* pode provocar

uma situação de desbalanceamento de carga, pois poderá existir situações onde um *cluster* poderá estar mais sobrecarregado do que outro. Neste caso, faz-se necessário redistribuir os pontos de dados entre os *clusters* existentes, obedecendo ao interesse dos pontos de dados em questão, na medida do possível. Além disso, podem ocorrer situações em que o interesse semântico de inúmeros pontos coincida, fazendo com que estes pontos concentrem-se em um único *cluster*. Portanto, é necessária a formação de um novo *cluster* semanticamente equivalente para a redistribuição desses pontos.

Sistemas P2P e PDMS que trabalham com *clusters* apresentam dois níveis de conexões: intra-*cluster* e inter-*cluster*. Nesses sistemas o desbalanceamento pode ocorrer nesses dois níveis, portanto soluções de balanceamento de carga intra e inter-*cluster* têm sido pesquisadas.

1.2. Definição do problema

No SPEED, PDMS que servirá de base para este trabalho, a conectividade dos pontos de dados aos *clusters* é baseada em relações semânticas, calculadas através de medidas de similaridades entre a ontologia local de cada ponto e a ontologia do *cluster*. Portanto, em determinado instante, uma grande quantidade de pontos de dados poderá fazer parte de um mesmo *cluster*, devido a suas proximidades semânticas. Embora essa grande quantidade seja permitida, têm-se algumas implicações com relação à eficiência do sistema. Essa eficiência pode ser vista tanto do ponto de vista do usuário quanto do sistema. Do ponto de vista do usuário, este deseja que sua consulta seja executada em um tempo hábil e com os melhores resultados possíveis. Já no ponto de vista de eficiência do sistema, este deverá garantir em sua arquitetura uma forma que seus dados estejam disponíveis e organizados de modo a satisfazer às necessidades de seus usuários, da forma mais eficiente possível.

Nesse PDMS, existem super-pontos que gerenciam cada *cluster* e que são responsáveis pelo gerenciamento de consultas (reformulação, integração dos resultados da consulta, dentre outros). Esses super-pontos são chamados de pontos de integração. O ponto de integração poderá ficar sobrecarregado por lidar com uma quantidade excessiva de pontos de dados, principalmente no aspecto do processamento de consultas para cada participante do *cluster*. Neste sentido, um mecanismo de balanceamento de carga deverá ser realizado de forma a não comprometer a eficiência do sistema. Como no SPEED cada cluster tem uma ontologia que representa os conceitos semânticos que ele detém, uma solução de balanceamento de carga poderá alterar a ontologia do *cluster* (pela migração de pontos, divisão do *cluster*, dentre

outras soluções). Isso implicaria em uma situação onde pontos de dados estariam presentes em um *cluster* no qual eles não possuíssem mais similaridades. A essa situação chamamos de desbalanceamento semântico. Uma vez que o SPEED adotou o mecanismo de "clusterização" para garantir uma melhor organização das fontes de dados, de acordo com seus interesses semânticos, uma situação de desbalanceamento semântico torna-se indesejável por tornar essa garantia questionável.

Portanto, um mecanismo de balanceamento de carga dos *clusters*, no qual não torne os *clusters* semanticamente desbalanceados, faz-se necessário para que a eficiência do sistema não seja comprometida.

1.3. Objetivo

O objetivo deste trabalho é desenvolver um processo de balanceamento de carga dos *clusters* semânticos existentes nos PDMS, cujas suas arquiteturas assemelhem-se a do SPEED. O processo deverá resultar em *cluster* com cargas e semânticas balanceadas.

Objetivos Específicos:

Os objetivos específicos desse trabalho são:

- Identificar ações que implicam em um estado de desbalanceamento de carga e semântico.
- Definir indicadores de desbalanceamento de carga e semântico.
- Identificar as operações sobre ontologias a serem realizadas com as ações das operações de divisão e junção de *clusters* e migração de pontos.
- Identificar situações onde operações de balanceamento semântico possam implicar em *clusters* com desbalanceamento de carga.
- Identificar as conseqüências das operações de balanceamento de carga sobre o balanceamento semântico dos *clusters*.
- Propor um algoritmo para balanceamento de carga e semântico nos níveis intra-*cluster* e inter-*cluster*.

1.4. Escopo

Este trabalho tem como escopo a busca de soluções para problemas com manutenção automática dos *clusters* que possuem conectividade semântica. Abordaremos questões referentes ao balanceamento de carga e semântico para PDMS com formação de *clusters* semânticos.

1.5. Estrutura do Documento

Este trabalho está organizado em cinco capítulos, os quais contemplam a qualificação e a proposta de tese, sendo o último capítulo dedicado à referida proposta.

No Capítulo 2 apresentamos um estado da arte sobre sistemas P2P, destacando suas principais características e topologias. Abordamos o problema de integração de dados e apresentamos abordagens para a implementação de sistemas de integração de dados. Por fim, apresentamos o conceito de ontologias e seus benefícios.

No Capítulo 3 é apresentada uma visão geral sobre PDMS, além de analisarmos exemplos reais de sistemas existentes na literatura.

No Capítulo 4 abordamos aspectos referentes às atividades de balanceamento de carga tanto em sistemas P2P quanto em PDMS. Neste trabalho, focaremos nos sistemas que realizam a "clusterização" dos seus pontos de dados. As soluções existentes na literatura para resolver o problema de balanceamento de carga são apresentadas neste capítulo, considerando os diferentes tipos de conexões estabelecidas entre os *clusters*.

Apresentamos, no Capítulo 5, o PDMS chamado SPEED (***S****emantic* ***Pee****r-to-Peer* ***D****ata Management System*). Enfatizamos a arquitetura desse sistema, considerando, principalmente, o seu mecanismo de formação de *clusters* semânticos. Em seguida, discutimos alguns aspectos sobre a necessidade de um mecanismo que realize o balanceamento de carga de seus *clusters*, considerando os aspectos semânticos destes *clusters*. O SPEED, discutido nesse capítulo, servirá como estudo de caso para a proposta de tese deste trabalho.

A proposta de tese que norteia este trabalho encontra-se descrita no Capítulo 6. Discutimos neste capítulo os problemas que existem ao realizar-se um balanceamento de carga nos *clusters* do SPEED. Logo, os problemas são discutidos e possíveis soluções são propostas para resolver esses problemas. Mostramos algumas contribuições esperadas neste trabalho, assim a metodologia a ser empregada e um cronograma para a execução das atividades desta tese.

Capítulo 2

Conceitos Fundamentais

Como um meio de prover uma infra-estrutura para um compartilhamento de recursos na *Web*, surgem os Sistemas *Peer-to-Peer* (P2P). Esses sistemas partem do princípio que um controle centralizado inexiste e que em cada ponto, os softwares uma vez executados detêm características funcionais equivalentes e os pontos atuam como cliente e/ou servidor. A motivação inicial desses tipos de sistemas foi a possibilidade de compartilhar arquivos. Facilmente um usuário podia fazer a busca por documento hipermídia, através do uso de palavras-chave, sem a necessidade de existir descrições semânticas para que esses documentos fossem encontrados. Porém, as fontes de dados não se limitam apenas a sistemas de arquivos, a pluralidade das diversas formas de persistência de dados é uma realidade hoje. Atualmente, podemos encontrar dados na *Web* em páginas HTML, documentos XML, em banco de dados, entre outros.

Nesse meio heterogêneo, é interessante para o usuário submeter uma consulta sobre uma determinada informação e essa consulta ser difundida sobre as diversas fontes de dados disponíveis na *Web*. No entanto, um sistema de integração de dados se faz necessário para identificar as fontes que possam responder e integrar os dados que atendam a essa consulta. Esse sistema integrador também deverá saber lidar com a heterogeneidade das fontes, sabendo-se que elas fazem parte de um mesmo domínio.

A proposta deste capítulo é fundamentar os sistemas P2P e mostrar o problema da integração de dados que também pode ser estendido a esses tipos de sistemas. Em seguida, conceituamos e discutimos como o uso de ontologias pode auxiliar no processo de integração de dados dentro desse contexto.

2.1. Sistemas *Peer-to-Peer* (P2P)

NOWELL *et al.* (2002) definem sistemas P2P da seguinte forma:

> *"Sistemas Peer-to-peer (P2P) são sistemas distribuídos sem algum controle centralizado ou organização hierárquica,onde o software que é executado em cada ponto (peer) é equivalente em funcionalidade."*

No foco deste trabalho, os sistemas P2P são vistos também como uma rede de *peers* (que neste trabalho chamaremos de pontos) interconectados e que cooperam entre si, trocando informações e serviços sem a intervenção de um servidor. Vários são os sistemas P2P implementados para os mais diversos fins, como podemos citar: computação distribuída, troca de mensagens, trabalho colaborativo e compartilhamento de dados. Em especial, sistemas que permitem a troca de arquivos tornaram as aplicações P2P bastante populares e difundidas entre usuários. São exemplos desses sistemas o Kazaa [KAZAA 2010], Napster [NAPSTER 2010], GNutella [GNUTELLA22 2010], E-mule [E-MULE 2010] dentre outros.

ANDROUTSELLIS-THEOTOKIS *et al.* (2004) apontam as características básicas de um sistema P2P como sendo:

- Pontos se conectam diretamente com outros pontos;
- Pontos são responsáveis pelos seus próprios dados;
- Pontos podem entrar e sair da rede a qualquer momento;
- Pontos podem atuar tanto como clientes quanto como servidores;
- Pontos são autônomos com relação ao controle e estruturação da rede, ou seja, não existe autoridade central.

Se comparados à arquitetura cliente-servidor, onde o número de servidores é fixo, sistemas P2P são mais flexíveis e extensíveis, visto que quando novos pontos entram na rede, a capacidade total do sistema aumenta, enquanto que na arquitetura cliente-servidor, a adição de novos clientes reduz o desempenho do mesmo [ZHAO 2006]. Devido a essas vantagens, os sistemas P2P têm robustez em falhas, extensivo compartilhamento de recursos, auto-organização, balanceamento de carga, persistência de dados, entre outras características.

2.1.1. Arquiteturas de Sistemas P2P

Sistemas P2P têm suas arquiteturas diferenciadas de acordo com alguns fatores. Nesta seção abordaremos as arquiteturas existentes para sistemas P2P, considerando esses fatores.

a. Quanto à forma de compartilhamento dos índices dos objetos

De acordo com [SUNG 2005] são diversas as categorias de arquiteturas de sistemas P2P. Considerando a forma como os índices dos objetos compartilhados são armazenados, temos as seguintes classificações.

I. Centralizados

Nestes sistemas existe um índice central, conforme a Figura 2-1, e com informações que ficam sendo constantemente atualizadas. Ao entrar na rede, o usuário se conecta a um ou mais servidores e estes atuam como um índice, para busca de dados disponíveis. Uma vez conectado, o usuário informa ao servidor todos os dados que ele está disponibilizando na rede. Neste momento, o servidor atualiza seus índices para disponibilizar esses dados para outros possíveis usuários.

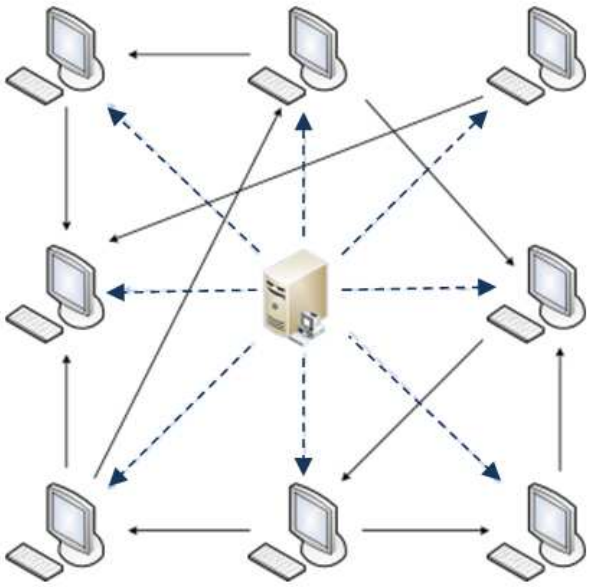


Figura 2-1 - Sistema P2P Centralizado

Quando um usuário realiza a busca por um dado elemento, ele primeiro conecta-se a um servidor e submete sua busca a ele. Após realizar a busca, o servidor retorna ao usuário os IP dos pontos que possuem os elementos procurados pelo usuário. Tanto o sistema de compartilhamento de arquivos do NAPSTER (2010) quanto do SOULSEEK (2010) fazem uso desse tipo de arquitetura.

II. Descentralizados

Nesta arquitetura inexiste o papel de um servidor que contém um índice central. Sua arquitetura pode ser visualizada conforme ilustra a Figura 2-2.

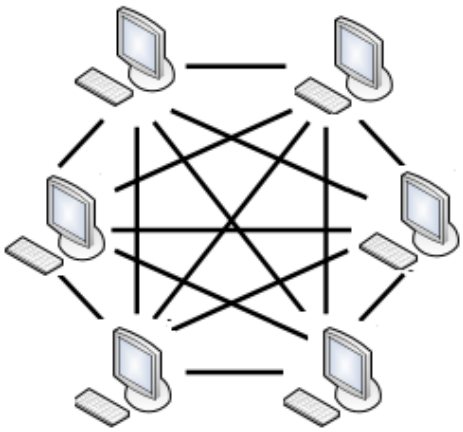


Figura 2-2 - Sistema P2P Descentralizado

Sistemas P2P descentralizados subdividem-se em não estruturados e estruturados. Um não estruturado não possui um controle preciso sobre a topologia, localização e busca de documentos. O KAZAA (2010) e o GNUTELLA2 (2010) são exemplos de sistemas P2P descentralizados e não estruturados. Em contrapartida, um sistema P2P descentralizado e estruturado possui uma significativa estruturação entre os pontos. Nessa, os documentos são postos em locais e, posteriormente, torna-se fácil sua localização e existe um controle da topologia da rede. Esse tipo de arquitetura utiliza busca baseada em DHT (*distributed hash table*) [STOICA 2001]. Uma DHT fornece um alicerce para a construção de sistemas distribuídos descentralizados, provendo um serviço de *lookup* similar a uma tabela hash (chave e valor), sem estabelecer uma hierarquia fixa [GHODSI 2006]. Cabem aos pontos a responsabilidade de manter o mapeamento de chaves para valores. Os sistemas BitTorrent [SUNG 2005], Chord [STOICA 2001], Pastry [DRUSCHEL 2001] e CAN [RATNASAMY 2001] fazem uso de uma arquitetura descentralizada e estruturada.

III. Modelo Hierárquico

O modelo hierárquico faz uso de pontos com características especiais, os super-pontos, para a execução de tarefas como a manutenção de índices, como ocorre no Kazaa [KAZAA 2010]. Este modelo pode ser visto como um intermediário entre os dois modelos anteriores [REZENDE 2009].

b. Quanto ao tipo de busca

Quanto ao mecanismo de busca em sistemas P2P, [TOWSLEY 2003] divide-os em três categorias:

- Serviço de localização centralizada (*Centralized Service Location* - CSL): neste mecanismo, existe um ponto central e a ele são submetidas consultas provenientes dos pontos existentes na rede. É através do ponto central onde as trocas de informações são realizadas entre os pontos. Podemos citar como exemplo de sistema P2P que realiza uma busca centralizada, o NAPSTER (2010).
- Serviço de localização por inundação (*Flooding-based Service Location* - FSL): neste mecanismo, não existe uma estruturação na rede, tão pouco um ponto central como no mecanismo anterior. Ao entrar na rede, um ponto entra e conecta-se aos demais pontos existentes na rede e as buscas são realizadas através de mecanismos de inundação[1]. O GNUTELLA2 (2010) é um exemplo de sistema P2P que realiza busca por inundação.
- Serviço de localização baseado em DHT (DHT *Service Location*): rede onde os pontos têm autonomia e usam um mecanismo de busca semelhante ao de uma tabela *hash*, porém mantêm partes dessa tabela em diversos pontos que compõem a rede [REZENDE 2009]. Os sistemas P2P Chord [STOICA 2001], CAN [RATNASAMY 2001], Tapestry [ZHAO 2001] e Pastry [DRUSCHEL 2001] realizam buscas por DHT.

Além dos três mecanismos citados anteriormente, [COSTA 2009] classifica mais dois mecanismos:

- Rede Semântica Overlay (*Semantic Overlay Network* - SON): existe uma rede virtual onde encontram-se agrupados pontos de acordo com ligações semânticas. Pontos unem-se formando uma sub-rede com valor semântico associado, podendo cada sub-rede se sobrepor. A Figura 2-3 ilustra uma divisão semântica baseada em categorias de filmes que cada ponto possui. E, no caso de um ponto possuir filmes de mais de uma categoria, ele entra em diversas sub-redes semânticas das quais ele faz parte da categoria. O ponto X possui filmes tanto da categoria “ação” quanto “suspense”.

[1] O termo inundação refere-se ao número de mensagens que são enviadas à rede até que o ponto procurado seja encontrado [REZENDE 2009].

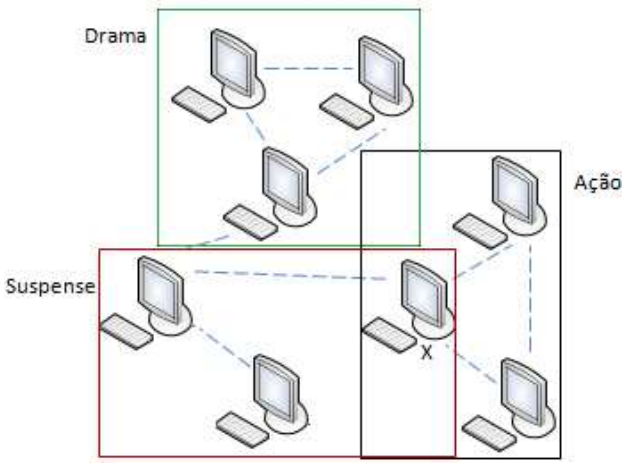


Figura 2-3 - Busca baseada em SON

As consultas realizadas são enviadas somente aos grupos semânticos relacionados, ignorando aqueles que não possuem qualquer relação à semântica estabelecida na consulta [COSTA 2009].

- Colônia de formigas (*Ant Colony Optimization System*- ACO): surgiu com o intuito de melhorar o desempenho de busca por inundação [COSTA 2009]. Algumas pesquisas [MICHLMAYR 2006, CIGLARIC *et al.* 2006] têm indicado o uso do algoritmo de colônia de formigas para a otimização dos caminhos que devem receber uma consulta. O algoritmo ACO é baseado no comportamento de formigas e de suas colônias. Alguns experimentos mostram que as formigas possuem um mecanismo de comunicação indireta que permite traçar uma melhor rota para chegar até o seu alimento, deixando marcas no caminho através do feromônio liberado por elas. Utilizando várias trilhas, as formigas detectam o melhor caminho pela maior intensidade de feromônio detectado por elas [COSTA 2009].

  O problema do caixeiro viajante [DORIGO *et al.* 1997] e a otimização de algoritmos para alinhamento múltiplo de seqüências em bioinformática [ZAFALON 2009] são exemplos de problemas que podem ser resolvidos através de colônia de formigas.

  Devido à absorção de mudanças dinâmicas em um grafo, o ACO permite ser utilizado no roteamento em redes de computadores dinâmicas, como no caso das redes P2P, fato este que pôde ser observado em [CIGLARIC *et al.* 2006]. No caso de redes que fazem uso de técnicas de inundação para o roteamento, o ACO pode predizer caminhos que possuem a melhor probabilidade de encontrar melhores resultados [COSTA 2009]. O AntNet [CARO *et al.* 1998] utilizou colônia de formigas para a geração de tabelas de roteamento em redes P2P. Através do uso de palavras-chave definidas em cada ponto, foram criados vários tipos de

feromônio, que resultam em diferentes caminhos para cada palavra-chave fornecida na busca desejada [COSTA 2009].

c. Outras classificações

No tocante a características relacionadas à confiabilidade, escalabilidade, desempenho, dentre outros, FIORANO (2003) compara e avalia diferentes topologias, objetivando determinar a arquitetura mais adequada para problemas que necessitam de computação distribuída. No trabalho [FIORANO 2003] são apresentadas as seguintes categorias para sistemas P2P:

- Arquitetura Pura

Modelo descentralizado onde não existe um ponto central de controle. Os pontos são auto-suficientes, atuam tanto como cliente quanto como servidor e estão conectados, podendo comunicar-se uns aos outros. Um exemplo de funcionamento pode ser visualizado através da Figura 2-4, onde o ponto X se interliga aos outros pontos os quais possuem outras conexões. Por sua vez, estes se conectam a outros pontos com os quais mantêm conexão e repetem a solicitação inicial. Tal procedimento é realizado repetidamente até que o recurso procurado seja encontrado, no caso, em "Y" [REZENDE 2009]. Quando o recurso é encontrado, uma conexão é criada entre o ponto onde o recurso foi encontrado e o ponto que fez a requisição, executando-se assim a transferência do recurso solicitado sem a existência de nenhum intermediário [LAFFRANCHI 2004].

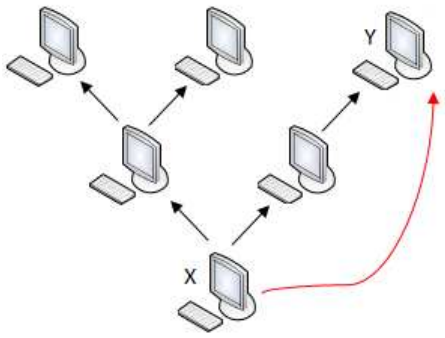


Figura 2-4 - Arquitetura pura de um sistema P2P

Essa arquitetura possui como característica a escalabilidade, uma vez que qualquer ponto pode entrar e trocar informações com outros pontos na rede [FIORANO 2003]. Considerando que a queda em um ponto não afeta o restante do sistema, podemos considerar

também que existe tolerância a falhas na arquitetura pura. Um dos principais sistemas que faz uso desta arquitetura é o sistema GNUTELLA2 [GNUTELLA2 2010].

- Arquitetura Híbrida

O modelo descentralizado é raramente utilizado devido a sua complexidade [REZENDE 2009]. Em uma arquitetura híbrida, os dados trafegam entre os pontos como na arquitetura pura, porém a informação de controle é tratada por um servidor central, que realiza o monitoramento para os pontos e garante a coerência das informações. A Figura 2-5 ilustra uma situação onde o ponto "X" entra em contato com o servidor para localizar um determinado recurso. Em seguida, o servidor coleta as informações necessárias para a conexão entre "X" e o ponto que possui o recurso desejado, no caso o ponto "Y". Uma vez que "X" e "Y" estabelecem uma conexão, a transação entre os pontos será executada.

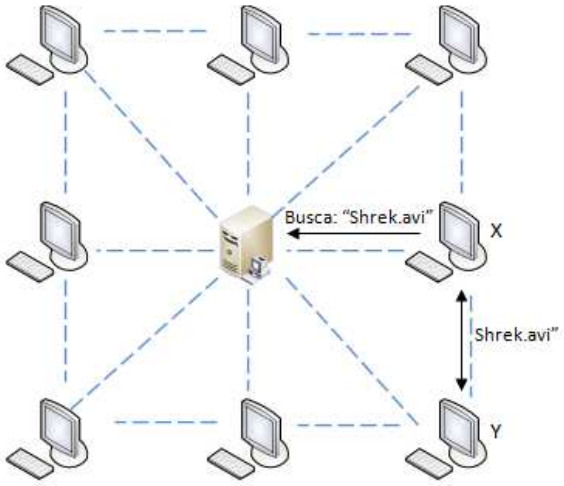


Figura 2-5 - Arquitetura híbrida

Sistemas híbridos têm sido utilizados principalmente em aplicações críticas, geralmente limitadas a um pequeno conjunto de problemas [FIORANO 2003]. O Groove [GROOVE 2010] é um sistema de gerenciamento de projetos colaborativos, que faz uso da arquitetura híbrida.

- Arquitetura Super-ponto

Nesta arquitetura existe um ponto global que gerencia outros pontos que estão conectados a ele. Esse ponto global é chamado de super-ponto. A idéia é de ter sub-redes P2P, cada uma com um super-ponto, e para que haja uma comunicação entre um ponto de uma

sub-rede com outro ponto de outra sub-rede, os super-pontos de ambas devem estar conectados. A Figura 2-6 ilustra uma arquitetura de super-pontos, onde estão conectados três super-pontos, A, B e C.

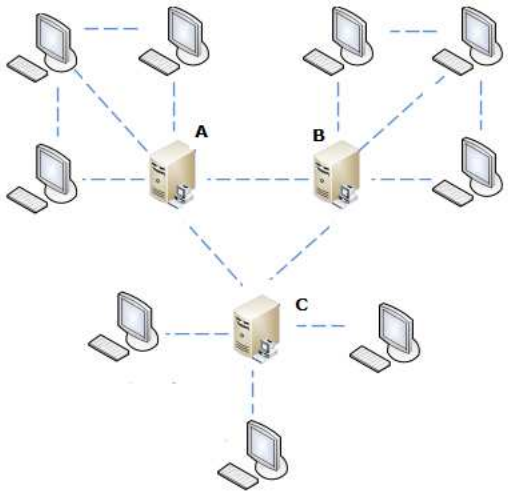


Figura 2-6 - Arquitetura de super-ponto

Nos sistemas que implementam uma arquitetura de super-pontos, existe uma agilidade quanto à busca da informação desejada, se comparada às outras arquiteturas. Tal agilidade deve-se ao fato que os sistemas são divididos em agrupamentos (*cluster*s) onde cada um destes possui informações indexadas sobre seus pontos. De acordo com FIORANO (2003), uma busca que leva O(n) em uma rede pura ou híbrida, levará O(n/m) (sendo m o número médio de pontos conectados a um super-ponto) em uma arquitetura de super-pontos. O sistema Kazaa [KAZAA 2010] é um exemplo que faz uso de uma arquitetura como esta.

Caso ocorra alguma falha em alguns dos super-pontos, os clientes poderão sentir essa falha. E para resolver esse problema, redundância de super-ponto (super-ponto *redundancy*) pode ser utilizada. Nesta solução pontos reservas são definidos para assumiram o papel de um super-ponto, caso ocorra algum tipo de falha. Logo, uma arquitetura de super-ponto redundante que combina vantagens tanto de sistemas centralizados quanto de sistemas descentralizados é considerada a mais apropriada para sistemas que necessitam de computação distribuída [FIORANO 2003].

2.1.2. Roteamento

Um dos principais objetivos de um sistema P2P é prover um mecanismo de busca de dados através de consultas utilizando-se palavras-chave [SUNG 2005]. A fim de identificar os pontos que podem responder às consultas, um mecanismo de roteamento se faz necessário e

sua implementação pode variar de acordo com a arquitetura utilizada, como visto na seção anterior. Algoritmos de buscas e roteamento procuram otimizar o encaminhamento de um ponto a um outro, assim como as estratégias de roteamento dependerão da característica dos sistemas de serem estruturados ou não.

As estratégias de roteamento são baseadas nos seguintes modelos: centralizado, inundação e DHT.

- Modelo Centralizado

A estratégia do modelo centralizado é manter um ponto central onde ficam armazenados nele o conteúdo que cada ponto na rede tem a oferecer. Logo, quando uma consulta é submetida por um ponto ao ponto central, este último escolhe o ponto que for mais adequado a atender à solicitação da consulta e, em seguida, a troca de dados é realizada entre os pontos.

No modelo centralizado, se um dado existir na rede, ele será encontrado e, como o mecanismo de busca por palavras-chave é fácil de implementar, isso traz vantagens ao modelo centralizado. Porém, esse modelo torna-se susceptível a falhas devido à concentração do conteúdo dos pontos em um único ponto.

- Modelo de Inundação

Neste modelo é realizada uma busca por inundação controlada por um tempo (*time to live* - TTL). Ao solicitar uma busca por um determinado dado, um ponto manda uma mensagem a todos os pontos conectados a ele. Se um dos pontos vizinhos possuir o dado desejado, o IP deste é retornado ao solicitante. Caso contrário, os pontos mandam mensagens aos seus pontos vizinhos e o procedimento de busca vai se repetindo até que o dado seja encontrado ou o TTL seja alcançado.

Devido à necessidade de um grande processamento e largura de banda, a escalabilidade do modelo de inundação fica comprometida. Além disso, pode haver casos onde o dado existe, porém não seja encontrado devido a sua distância, ao ponto solicitante da busca, ser grande e o tempo do TTL ser logo alcançado.

- Modelo DHT

Nesse modelo, um ID randômico é associado a cada ponto que conhece um determinado número de pontos como poder ser visto na Figura 2-7. Ao ser publicado um documento no sistema P2P, um valor de ID é gerado através de uma função *hash* tomando como entrada o conteúdo dos documentos e o seu nome. Em seguida, o ponto repassa o

documento ao ponto cujo ID é o mais próximo do ID do documento. Esse processo é repetido até que o ID do ponto atual seja o mais próximo do ID do documento [LOEST 2007]. Uma vez encontrado, o documento será transferido ao ponto solicitante, enquanto que cada ponto que participou do roteamento ficará com uma cópia local do documento. Com isso, podemos afirmar que cada operação de roteamento também garante que uma cópia do documento seja mantida [LOEST 2007, LUA *et al.* 2004].

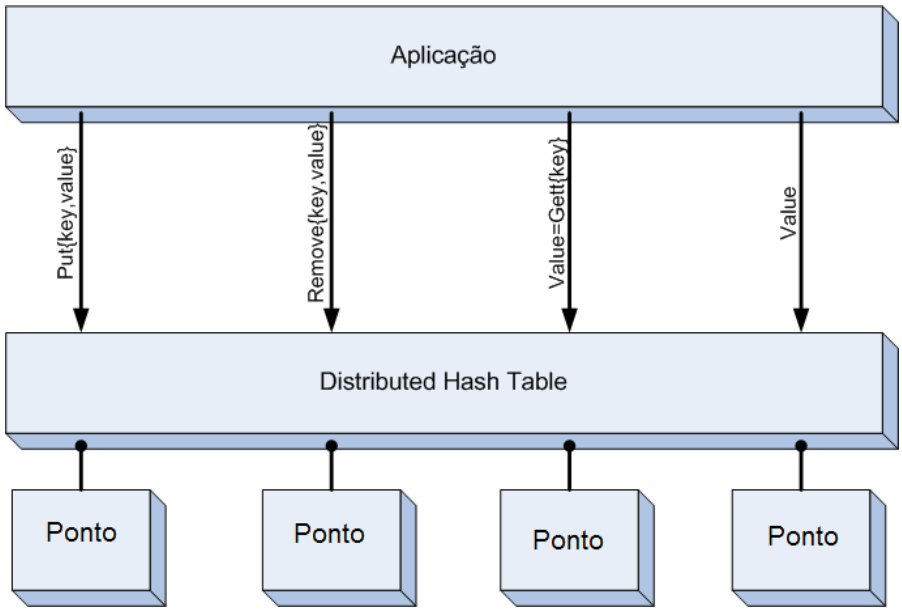


Figura 2-7 - Sistema DHT [LUA *et al.* 2004]

Embora o modelo DHT seja considerado eficiente para comunidades grandes e globais, ele apresenta um problema relacionado ao ID do documento. Este precisa ser conhecido antes que uma requisição do documento seja realizada. Assim, é mais difícil implementar uma pesquisa nesse modelo que no modelo de inundação. Além disso, pode ocorrer a formação de "ilhas", onde a comunidade se divide em subcomunidades que não possuem nenhuma ligação entre si [KAMIENSKI *et al.* 2005, LOEST 2007].

2.2. Integração de Dados

Com a *Web*, um grande volume de informações encontra-se disposto de forma distribuída e sob diferentes formas de armazenamento (banco de dados, sistema de arquivos,

XML, entre outros), caracterizando um ambiente cujas fontes são heterogêneas. Além disso, cada uma dessas fontes possui autonomia sobre seus dados e os dispõe apenas para leitura. Logo, com essa heterogeneidade, a atividade de realizar uma consulta a um domínio específico não representa uma tarefa fácil. Assim, para resolver esse problema, um novo tipo de sistema, chamado de sistema de integração de dados, surgiu buscando a integração entre fontes heterogêneas, distribuídas e autônomas, através de uma visão global e única.

De acordo com SOUZA (2007), Sistemas de Integração de Dados têm como principal objetivo liberar o usuário de ter que, manualmente, localizar fontes, interagir com elas isoladamente e depois combinar os resultados obtidos. Um aspecto fundamental em sistemas desta categoria é que eles devem ser capazes de unir todos os esquemas das fontes associadas, de modo a criar uma visão única que será o elo de comunicação com o usuário. Esta visão única é chamada de esquema de mediação ou esquema global.

Em [HALEVY *et al.* 2006], os autores descrevem uma retrospectiva sobre as pesquisas em integração de dados do período de 1996 a 2006. Nesse trabalho, os autores apontavam direções de pesquisas para soluções de integração de dados em sistemas de gerenciadores de dados em P2P (PDMS – *Peer Data Management System*). Abordaremos, com mais detalhes, sobre esses sistemas no Capítulo 3.

2.2.1. Abordagens para sistemas de integração de dados

Para o projeto de um sistema que realize a integração de dados, duas abordagens podem ser utilizadas: a abordagem materializada [KIMBALL *et al.* 2002] e a abordagem virtual [WIEDERHOLD 1992].

2.2.1.1. Abordagem materializada

Na abordagem materializada, as informações mais relevantes são previamente extraídas das fontes de dados distribuídas, integradas e armazenadas em um repositório comum. Logo, as consultas submetidas a esses sistemas, terão como fonte esse repositório comum, podendo implicar em uma inconsistência entre esse repositório e as fontes de dados. Em contra partida, o tempo de resposta da consulta será minimizado.

Os *data warehouses* são exemplos de sistemas de integração que usam a abordagem materializada. A arquitetura de um sistema de informação baseado em *data warehouse* é ilustrada na Figura 2-8.

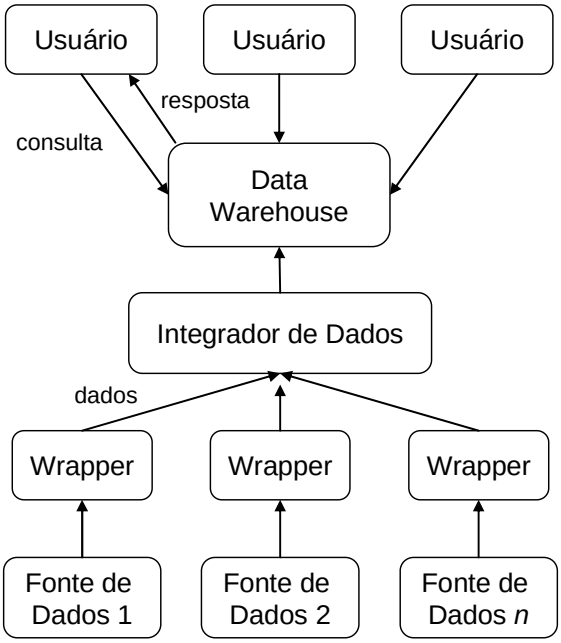


Figura 2-8 - Arquitetura Simplificada de um Sistema de Informação baseado em *Data Warehouse* [PIRES 2007]

Para cada fonte de dados existe a necessidade de um *wrapper* para traduzir a informação do formato da fonte de dados para o formato e modelo utilizado pelo sistema de *data warehousing*. Além disso, o *wrapper* faz a ligação entre a fonte de dados e o integrador de dados, este último responsável por instalar as informações no *warehouse*, através de procedimentos como filtragem, sumarização e integração das informações provenientes das fontes distribuídas.

2.2.1.2. Abordagem virtual

Ao contrário da abordagem materializada, a abordagem virtual não armazena as informações em um repositório único para posteriores consultas. Cada consulta submetida passará apenas pelas fontes de dados distribuídas capazes de responder a consulta, e as informações retornadas estarão sempre atualizadas. Entretanto, se uma das fontes de dados não estiver disponível, algumas informações não estarão disponíveis.

A Figura 2-9 ilustra o mediador como uma camada que, de acordo com WIEDERHOLD (2002), visa explorar o conhecimento codificado sobre determinados conjuntos ou sub-conjunto de dados, para gerar informações destinadas a uma camada de mais alto nível composta de aplicações.

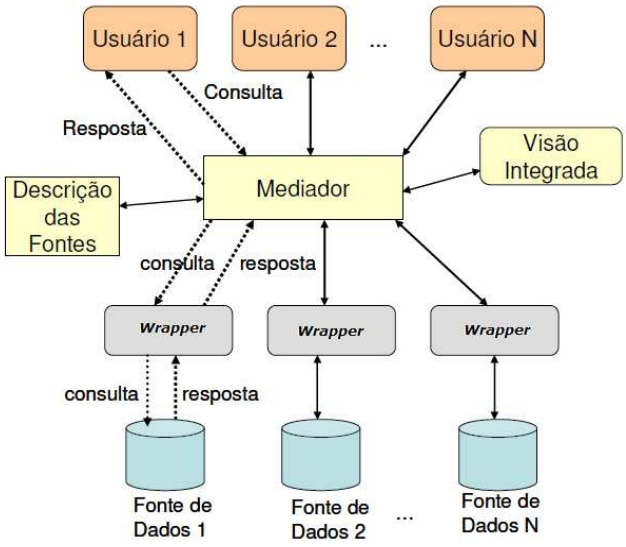


Figura 2-9 - Arquitetura de Mediadores [SOUZA 2007]

Na arquitetura baseada em mediadores [WIEDERHOLD 2002], destacam-se os dois componentes: *wrapper* e o mediador. O *wrapper* reescreve as consultas submetidas à aplicação para que cada fonte de dados seja capaz de respondê-la. Além disso, os dados retornados pelas diversas fontes de dados são convertidos, pelo *wrapper*, para o modelo de dados do sistema integração. Os mediadores oferecem uma visão integrada sobre os dados distribuídos nas diversas fontes de dados e disponibilizam um esquema para essa visão, bem como descrições das fontes de dados [SOUZA 2007]. Logo, uma consulta recebida por um mediador é decomposta em várias subconsultas (destinadas a cada fonte de dados) que serão traduzidas pelos *wrappers*.

2.2.2. Estratégias de mapeamento entre esquemas

Uma consulta submetida por um usuário é formulada de acordo com o esquema de mediação. Em seguida, de acordo com o esquema das fontes de dados, a consulta submetida será reformulada. De acordo com SOUZA (2009), existem quatro estratégias para especificar o mapeamento entre esquemas: GAV (*Global-as-view*), LAV (*Local-as-view*), GLAV (*Global-local-as-view*) e BAV (*Both-as-view*).

A estratégia GAV o sistema descreve a entidade mediadora como visões sobre os esquemas das fontes. Já na LAV, as fontes são descritas como visões sobre o mediador. O mapeamento GLAV é especificado por uma relação de equivalência entre as conjunções de consultas sobre os esquemas da fonte e o do destino. No BAV, os esquemas são mapeados

para cada outro utilizando uma seqüência bidirecional de transformação de esquemas as quais são chamadas de caminhos de transformações [McBRIEN *et al.* 2006]. Desses caminhos é possível extrair uma definição do esquema global como uma visão sobre os esquemas locais (GAV), e é possível extrair, também, definições dos esquemas locais como visões sobre o esquema global (LAV) [SOUZA 2007]. O Quadro 2-1 destaca alguns sistemas de integração que fazem uso das abordagens de mapeamento descritas nesta seção.

Quadro 2-1 - Abordagem de mapeamentos e sistemas

Abordagem	Sistema(s)
LAV	APPA [VALDURIEZ *et al.* 2004]
GAV	SEWASIE [FILLOTTRANI 2005]
GLAV	Piazza [HALEVY *et al.* 2004], Humboldt Discoverer [HERSCHEL *et al.* 2005]
BAV	iXPeer [BELLAHSÈNE *et al*. 2006a]

2.3. Ontologias

Segundo GRUBER (1993) "Ontologia é uma especificação formal e explícita de uma conceituação compartilhada". O termo formal aplicado nessa definição significa que a ontologia deve ser passível de processamento automático. Explícita porque os elementos e suas restrições estão claramente definidos. Conceituação porque representa um modelo abstrato de algum fenômeno que identifica os conceitos relevantes para o mesmo. E compartilhada, reflete a noção de que uma ontologia captura conhecimento consensual, aceito por um grupo de pessoas [BREITMAN 2005].

Como ontologias têm sido utilizadas por várias comunidades – Inteligência Artificial, Representação do Conhecimento, Processamento de Linguagem Natural, *Web* Semântica, Engenharia de Software, Banco de Dados, entre outras – é razoável que exista uma grande variação desse artefato. USCHOLD *et al.* (1999) propõem uma definição mais abrangente:

> "*Uma ontologia pode assumir vários formatos, mas necessariamente deve incluir um vocabulário de termos e alguma especificação de seu significado. Esta deve abranger definições e uma indicação de como os conceitos estão inter-relacionados, o que resulta na estruturação do domínio e nas restrições de possíveis interpretações de seus termos*."

O consórcio W3C (2011) define uma ontologia como a *"definição dos termos utilizados na descrição e na representação de uma área de conhecimento"*. Esse consórcio coloca que ontologias devem prover descrições para os seguintes tipos de conceitos:

- Classes (ou "coisas") nos vários domínios de interesse;
- Relacionamentos entre essas classes;
- Propriedades (ou atributos) que essas classes devem possuir.

Independente da definição escolhida é necessário entender que ontologias têm sido utilizadas para descrever artefatos com variados graus de estruturação e diferentes propósitos. A variação vai de simples taxonomias até representações para metadados, tais como Dublin Core, chegando a modelos escritos em lógica.

2.3.1. Classificação das ontologias

Em relação ao conteúdo e à natureza e abrangência dos conceitos definidos, ontologias podem ser classificadas nos seguintes tipos [GUARINO 1998]:

- Ontologias de representação: definem as conceituações que fundamentam os formalismos de representação de conhecimento, definindo as primitivas de representação.
- Ontologias genéricas: trazem definições gerais a respeito de conceitos abstratos, tais como, tempo, espaço, matéria, seres, coisas, entre outros, independentes de um problema ou domínio particular, mas necessárias para a compreensão de aspectos do mundo.
- Ontologias centrais: definem os ramos de estudos de uma área. Um exemplo é ontologia central de direito. Ontologias centrais servem de base para a construção de ontologias de ramos mais específicos (por exemplo, direito de família).
- Ontologias de domínio: expressam conceituações de domínios particulares de uma área genérica (direito, medicina, entre outros) de conhecimento.
- Ontologias de aplicação: descrevem conceitos dependentes do domínio e das tarefas que lidam com um problema específico desse domínio como, por exemplo, identificar doenças do coração.
- Ontologias de tarefas: descrevem o vocabulário ligado a uma tarefa ou atividade específica.

2.3.2. Usos e Aplicações de ontologias

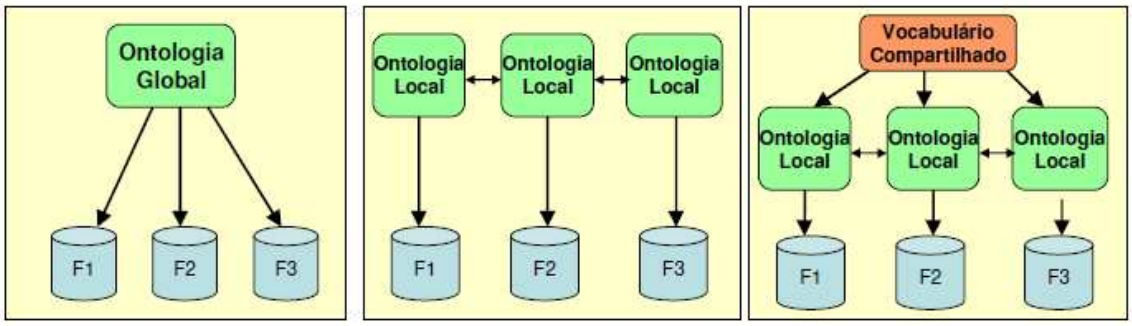
Ontologias podem ser utilizadas nos seguintes aspectos [BREITMAN 2005]:

- Comunicação: provendo uma estrutura que reduz a ambigüidade que uma organização possua a respeito de conceitos e termos. Também promovem a comunicação e o compartilhamento de conhecimento entre pessoas das mais diversas organizações.
- Interoperabilidade: fornecem ambientes integrados onde diferentes ferramentas de software possam ser utilizadas em conjunto, como nas arquiteturas multi-agentes. Nesses casos, ontologias funcionam como uma linguagem para auxiliar a interoperação entre tais ferramentas.
- Engenharia de Sistemas: apoiando o projeto e o desenvolvimento de sistemas. Identificando requisitos e o entendimento das relações entre seus componentes.
- Web Semântica: viabilizando o fornecimento de descrições que complementam o conteúdo dos documentos disponíveis na *Web*, de forma que máquinas possam lê-las.

2.3.3. Ontologias e Integração de Dados

Ontologias podem ser utilizadas pelos sistemas de integração, com intuito de descrever a semântica dos esquemas das diferentes fontes de dados. Ontologias podem ajudar a lidar com problemas de integração semântica e heterogeneidade dos dados, e assim auxiliar na integração dos dados de fontes distribuídas [PIRES *et al.* 2011].

A Figura 2-10 apresenta o uso de ontologias em diferentes arquiteturas de integração de dados. Na Figura 2-10 (a) ilustra uma situação na qual todas as fontes de dados estão relacionadas a uma única ontologia global, a qual possui o conjunto de termos que são compartilhados pelas fontes. Segundo SOUZA (2007), a abordagem global única é adequada à construção de sistema de integração de dados que utilizam a abordagem GAV. Assim como no caso do uso em sistema P2P, onde existe a presença de um super-ponto, a ontologia global residiria nesse super-ponto.



(a) (b) (c)

Figura 2-10-Arquiteturas de Integração que utilizam ontologias [SOUZA 2007]

Na arquitetura ilustrada pela Figura 2-10 (b), cada fonte de dados possui seu esquema representado por uma ontologia local, não há uma ontologia global, porém existe um mecanismo que provê a inter-relação entre essas ontologias. Esse modelo arquitetural é mais adequado para o uso da abordagem LAV e, no caso de ambientes P2P, nos sistemas de topologia pura.

Por fim, a arquitetura destacada pela Figura 2-10 (c) representa uma arquitetura onde cada fonte de dados possui seus esquemas descritos através de suas próprias ontologias locais, porém cada uma dessas ontologias é construída a partir de um vocabulário comum. Essa arquitetura é mais adequada a abordagem LAV e, no uso de sistemas P2P, em topologias com super-ponto.

Alguns processos que envolvem ontologias recebem atenção pela comunidade de Banco de Dados e de *Web* Semântica. Esses processos visam representar da melhor forma os esquemas das fontes de dados, a fim de uma melhor integração e manipulação dessas fontes. Alguns desses processos são: *matching, merging* e *summarizing* de ontologias.

- *Matching* de ontologias

Matching de ontologias é o processo de encontrar relacionamentos ou correspondências entre elementos de duas ontologias distintas (pertencentes a um mesmo domínio ou um similar) [PIRES 2009]. A saída desse processo é um alinhamento de ontologias (um conjunto de correspondências indicando quais elementos, de duas ontologias, correspondem a cada um desses elementos).

As correspondências são expressas como relacionamentos (disjunção, equivalência entre outras) ou como valores de similaridades entre zero (baixa similaridade) e um (alta similaridade). As correspondências são utilizadas para calcular a medida de similaridade global entre duas ontologias. Essa medida indica o grau de similaridade (proximidade dos conceitos) existente entre duas ontologias. Para os sistemas de integração, esse processo pode ser utilizado para identificar, semanticamente, fontes que possuem similaridades semânticas.

Os alinhamentos são usados em várias tarefas entre elas na tradução de dados em sistemas de integração e no processo de *merging* de ontologias discutido na próxima seção.

- *Merging* de ontologias

O processo de *merging* de ontologias consiste em obter uma nova ontologia a partir de outras duas. As entidades combinadas nas ontologias fontes estão relacionadas como prescrito pelo alinhamento gerado pelo processo de *matching*. A saída de um processo de *merging* é

uma ontologia que reúne os conceitos definidos por duas ontologias, ou seja, a ontologia gerada contém o conhecimento das ontologias fontes.

Em um sistema de integração de dados, onde todas as fontes fazem parte do mesmo domínio, é possível ter uma visão geral dos esquemas dessa fonte. Essa visão geral pode ser obtida através do *merging* das ontologias dessas fontes. De acordo com PIRES (2009), a noção de *merging* de ontologias está intimamente relacionada com o esquema de integração em banco de dados.

- *Summarizing* de ontologia

Uma sumarização (*summarizing*) de uma ontologia provê uma visão sucinta de uma ontologia inteira, tornando possível explorar apenas os elementos mais relevantes dessa ontologia. Segundo PIRES *et al*. (2010), em se tratando de sistemas de integração de dados, um sumário de um esquema de uma fonte de dados poderá ser útil para compreendê-la e melhorar alguma tarefa de integração automatizada como, por exemplo, a comparação entre dois esquemas. Se os esquemas estão representados através de ontologias, a tarefa de comparar duas ontologias sumarizadas, ao invés de duas ontologias inteiras, reduz o custo do processo como um todo. Em [PIRES *et al.* 2011] estão descritas algumas pesquisas referentes a *summarizing* (que neste trabalho chamaremos de sumarização) em fontes de dados.

2.4. Considerações finais

Este capítulo conceituou e classificou sistemas P2P. Exemplos de sistemas foram citados, de acordo com as classificações apresentadas. Em seguida, o problema da integração de dados foi abordado, assim como algumas abordagens existentes para a implementação de sistemas que permitem a integração de dados. Além disso, métodos de mapeamentos de esquemas necessários em ambientes integradores foram apresentados, assim como exemplos de sistemas que fazem uso desses métodos. No capítulo que sucede este, abordaremos o problema de integração de dados, onde as fontes estão em um ambiente P2P.

Ainda neste capítulo, apresentamos conceitos de ontologias, sua classificação e o seu uso em alguns tipos de aplicações. Por fim, abordamos a fundamental importância do uso de ontologias, em sistemas de integração de dados, para a descrição dos aspectos semânticos a fim de enriquecer as representações das fontes de dados, buscando uma melhor integração entre essas fontes.

Capítulo 3

Sistemas de Gerenciamento de Dados em P2P

Os constantes avanços tecnológicos ocorridos na área de banco de dados propiciaram o surgimento de sistemas de integração de dados, proporcionando aos seus usuários total transparência em seu uso. Neste sentido, a evolução dos bancos de dados foi aliada às redes P2P, para o compartilhamento de dados. Dessa união surgiram os sistemas gerenciadores de dados em ambientes P2P (*Peer Data Management Systems*- PDMS) com o intuito de prover aos usuários os benefícios de sistemas P2P e as facilidades pertencentes aos SGBD como linguagens de consultas, modelos semânticos e facilidade de uso. Os PDMS visam prover, aos usuários, mais facilidades de consulta, ao mesmo tempo em que eles tenham visões simplificadas dos dados que se encontram, na verdade, localizados em fontes distribuídas, heterogêneas e autônomas.

Um PDMS é composto por um conjunto de pontos (*peers*), cada um com um esquema associado que representa seu domínio de interesse. Geralmente mapeamentos são providos entre pontos ou um pequeno número de pontos que compartilham interesses comuns. Logo, quando um usuário formula uma consulta, o sistema pode obter dados relevantes de qualquer ponto conectado através destes mapeamentos.

Uma vez que os PDMS possuem como alicerce um sistema P2P, todas as características deste último são herdadas pelos PDMS. Como exemplo de tal característica podemos citar a autonomia que cada ponto possui para entrar e sair da rede. No tocante ao ponto de vista de gerenciar dados, um PDMS deverá tratar problemas como [SUNG *et al.* 2005]:

- Localização de dados: pontos capazes de identificar e localizar dados armazenados em outros pontos.
- Processamento de consulta: dada uma consulta realizada em um ponto, o sistema deverá ser capaz de descobrir os pontos que podem contribuir com informações relevantes ao que se é esperado pela consulta, e processá-la de forma eficiente para os usuários.
- Integração de dados: uma vez que uma consulta foi submetida e os pontos que atendam à consulta foram localizados, os dados acessados precisam ser integrados e retornados aos usuários, mesmo que as fontes apresentem diferentes esquemas e representações.

- Consistência de dados: em caso de replicação e uso de *cache*, a consistência dos dados deverá ser preservada.

Em um PDMS cada ponto está associado a um esquema que representa o seu domínio de interesse, e as relações semânticas entre os pontos são organizadas entre pares ou conjuntos pequenos de pontos. Percorrendo caminhos obtidos através de mapeamentos, as consultas submetidas em um ponto podem obter dados relevantes e complementares de qualquer outro ponto na rede [HAVELVY *et al.* 2004].

Segundo XU (2011), PDMS é a melhor escolha para a integração de dados em ambientes heterogêneos pelas três seguintes razões:

- Um PDMS não depende de um esquema de mediação, permitindo uma maior flexibilidade para lidar com vários domínios de conhecimento.
- O mecanismo de mapeamento "*pay-as-you-go*" é fácil de ser implementado em um PDMS, pois cada fonte de dados pode decidir quando e o que mapear. Segundo SARMA *et al.* (2008), "*pay-as-you-go*" é um mecanismo de mapeamento que reduz o custo de operações de mapeamentos necessários aos sistemas de integração. Esse custo é reduzido porque nesse mecanismo os mapeamentos são feitos, gradualmente, quando eles se fizerem necessários ou quando existirem recursos para fazê-los.
- Em uma situação em que não se pode assumir que a rede e a infraestrutura de comunicação são estáveis, é desejável que as fontes de dados possam, rapidamente, organizarem-se de forma *ad-hoc*, em uma rede P2P. Portanto, PDMS encaixam-se nesse cenário.

3.1. Requisitos para PDMS

De acordo com [VALDURIEZ *et al.* 2004] alguns dos requisitos necessários ao se projetar um PDMS são:

- Autonomia: um ponto dever ser capaz de entrar e sair do sistema a qualquer momento e sem qualquer restrição. Um ponto também deve controlar seus dados armazenados e saber quais outros pontos confiáveis também podem armazenar os seus dados.
- Expressividade da consulta: um PDMS deve permitir fazer o uso de uma linguagem de consulta expressiva, com um nível adequado de detalhes. Em um ambiente onde os dados estão estruturados, uma linguagem como SQL se faz necessária.

- Eficiência: o uso eficiente de recursos de sistemas P2P (poder computacional, escalabilidade, armazenamento, tolerância a falhas, largura de banda dentre outros) deve resultar em um menor custo, possibilitando um maior processamento de consultas em um determinado intervalo de tempo.
- Qualidade do serviço: em um PDMS deve ser considerada a completude dos resultados de uma consulta, assim como o tempo de sua resposta. A disponibilidade e a consistência dos dados também devem ser consideradas.
- Tolerância a falhas: a qualidade do serviço e a sua eficiência devem existir mesmo na ocorrência da falhas. Portanto, uma solução seria a replicação de dados devido à natureza dinâmica existente nos PDMS (pontos podem entrar e sair a qualquer momento).
- Segurança: devido à natureza aberta dos sistemas P2P, isso põe em risco um PDMS uma vez que pontos não muito confiáveis podem entrar no sistema. Considerando esse aspecto, questões sobre o controle de acesso deverão ser consideradas.

3.2. Classificação dos PDMS

Além dos requisitos citados anteriormente, VALDURIEZ *et al.* (2004) classificam os PDMS em:

- Não estruturados: cada ponto pode se comunicar diretamente com seus vizinhos.
- Estruturados: são baseados em DHT.
- Super-ponto: alguns pontos podem realizar a tarefa mais complexa como indexação, processamento de consultas e gerenciamento de metadados.

O Quadro 3-1 ressalta a análise dos diversos tipos de PDMS de acordo com os requisitos definidos em [VALDURIEZ *et al.* 2004]:

Quadro 3-1 - Requisitos versus tipos de PDMS [PIRES 2007]

Requisito\Tipo PDMS	Não Estruturado	Estruturado	Super ponto
Autonomia	Alta	Baixa	Média
Expressividade da consulta	Alta	Baixa	Alta
Eficiência	Baixa	Alta	Alta
Qualidade do serviço	Baixa	Alta	Alta
Tolerância a falhas	Alta	Alta	Baixa
Segurança	Baixa	Baixa	Alta

3.3. Principais PDMS Existentes

Diversas pesquisas vêm sendo desenvolvidas com o objetivo de propor soluções atender aos problemas referentes ao gerenciamento de dados em ambientes P2P. Na literatura, vários PDMS têm sido desenvolvidos para tratar alguns desses problemas. Esta seção oferece uma breve descrição de alguns dos principais PDMS, destacando suas principais características. Ao final da seção, apresentamos um quadro comparativo entre os PDMS discutidos.

3.3.1. PeerDB

Desenvolvido pela Universidade de Singapura, o PeerDB é um PDMS implementado sobre a plataforma BestPeer [NG *et al.* 2002], baseado na topologia pura não-estruturada, onde cada ponto possui dados que são gerenciados por um SGBD relacional, proporcionando buscas por conteúdo. Os dados gerenciados são compartilhados sem a presença de um esquema global e acessado através da linguagem SQL.

A Figura 3-1 ilustra a arquitetura do PeerDB que basicamente é composta por três camadas:

- Camada P2P: camada responsável por ofertar serviços de descoberta de recursos na rede e por compartilhar dados.
- Camada de agentes inteligentes: sistema multi-agentes que viabiliza uma infra-estrutura para que agentes móveis possam operar nos diversos pontos da rede. E, em cada ponto existe um agente responsável por gerenciar as consultas dos usuários, esse agente é chamado de DBAgent. Além disso, o DBAgent monitora estatísticas sobre pontos vizinhos e gerencia políticas de configuração da rede.
- Camada de gerenciamento de dados: camada responsável pela manipulação dos dados disponíveis no ponto. Nesta existe mecanismos de *cache* para proporcionar o armazenamento temporário dos resultados provenientes de diversos pontos, com o intuito de minimizar o tempo de resposta às consultas subsequentes. Para o gerenciamento de *cache*, o PeerDB faz uso do algorítimo LRU [O'NEIL *et al.* 1993].

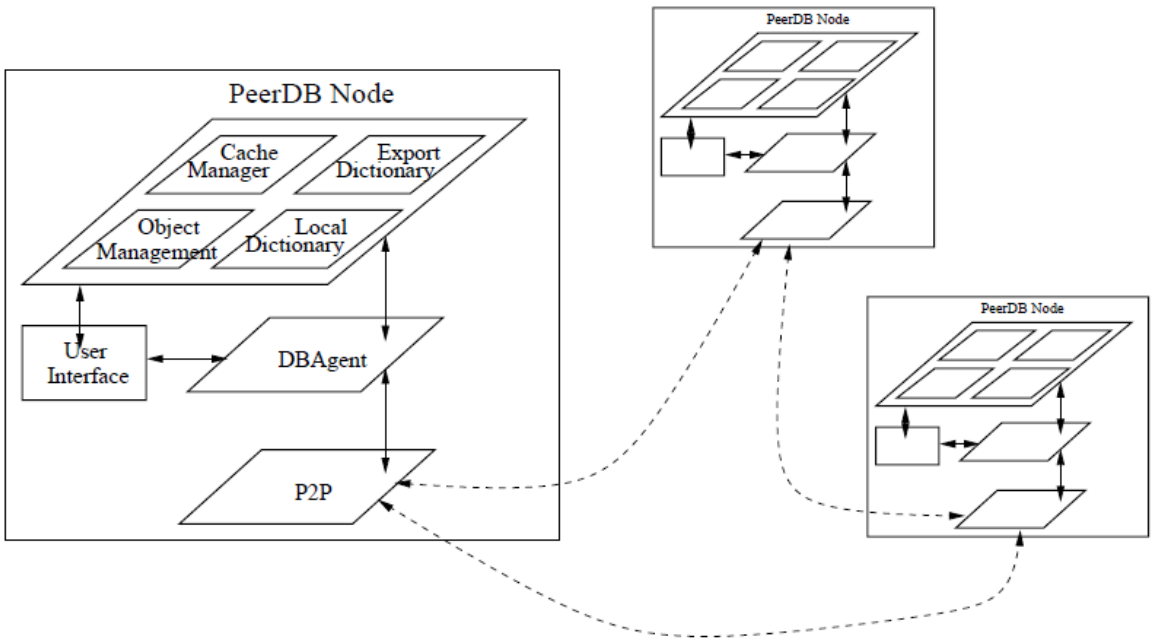


Figura 3-1- Arquitetura do PeerDB [OOI *et al.* 2003]

3.3.2. Piazza

Desenvolvido pela Universidade de Washington e da Pensilvânia, o Piazza foi concebido com o intuito de ser um PDMS escalável em um ambiente heterogêneo e distribuído. O sistema assume que usuários participantes interessados em compartilhar seus dados, devem estar dispostos a definir mapeamentos entre seus esquemas, conforme ilustra a Figura 3-2. Cada ponto compartilha seus dados na forma de relações armazenadas, de forma que outros pontos possam acessá-las. Além disso, o ponto mantém dois tipos de mapeamentos de esquemas. O primeiro refere-se às relações armazenadas e ao esquema do ponto. O segundo refere-se ao mapeamento entre o esquema do ponto com os esquemas de seus vizinhos [HALEVY *et al.* 2003a].

Em [HALEVY *et al.* 2003b] encontra-se relatado que o Piazza suporta compartilhamento de dados em XML/RDF para suporte a aplicações de *web* semântica. Portanto, o esquema do ponto é descrito usando XML *Schema* ou ontologias na liguagem OWL para dados XML e RDF, respectivamente. Além disso, a linguagem para mapeamento e para consulta é baseada em XQuery.

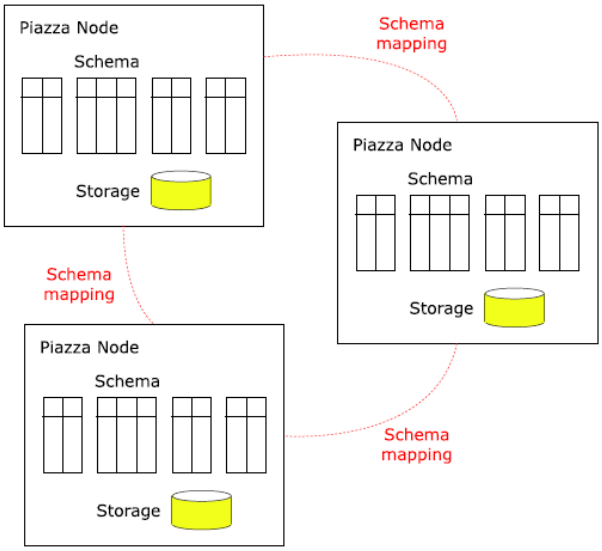


Figura 3-2-Arquitetura do Piazza [VU *et al.* 2010]

O Piazza possui uma arquitetura puramente descentralizada, inexistindo a presença de um esquema global único. O conjunto de mapeamentos definidos por este sistema visa definir a semântica do mesmo. O mapeamento é implementado através de um gráfico arbitrário de esquemas conectados, sendo alguns destes esquemas definidos virtualmente para propósitos de consulta e de mapeamento. E, para a construção dos mapeamentos, o Piazza baseia-se no uso conjunto de heurísticas e algoritmos a fim de resolver problemas semânticos. Estes por sua vez são baseados em técnicas de aprendizagem de máquina e na exploração de experiências anteriores, onde informações sobre mapeamentos válidos já existentes são utilizados para o mapeamento de novos esquemas.

3.3.3. SEWASIE

SEWASIE (*SEmantic Webs and AgentS in Integrated Economies*) é um projeto de pesquisa, financiado pela Comissão Européia, que visa implementar um mecanismo de busca avançado e que permita acesso inteligente a fontes de dados heterogêneas espalhadas na *web*, através de enriquecimento semântico [SEWAISE 2010].

SEWAISE foi projetado utilizando sistemas multi-agentes com base segura, escalável e com foco na arquitetura de um sistema distribuído, para busca semântica com ontologias específicas da comunidade multilíngüe. Ontologias são utilizadas em camadas de inferência e baseadas em padrões W3C [BENEVENTANO *et al.* 2003].

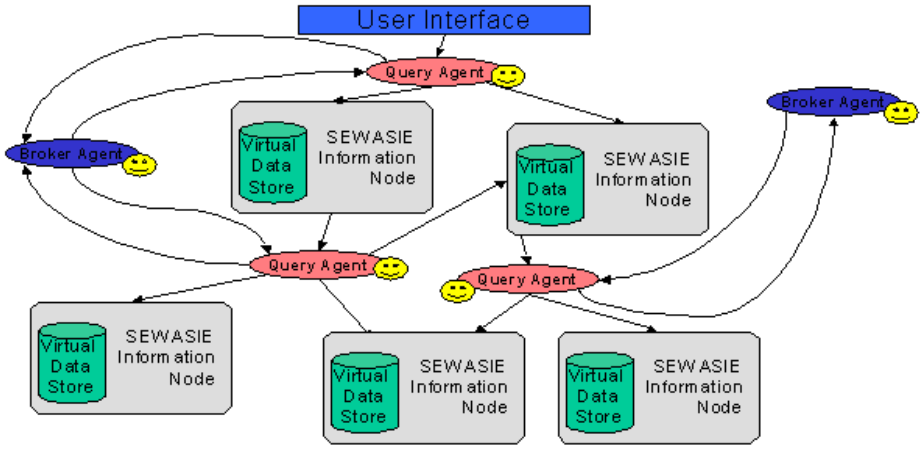


Figura 3-3 - Arquitetura SEWAISE [SEWAISE 2010]

A Figura 3-3 especifica os componentes da arquitetura do SEWAISE que são o ***user interface***, **SINode**, ***broker agent*** e ***query agent***. O *user interface* possui um conjunto de módulos integrados que juntos propiciam ao usuário uma interação com um sistema de busca semântica. Esta interface é personalizada e configurada com o perfil de um usuário específico e com uma referência para as ontologias que são comumente usadas por esse usuário. O *SINode (SEWAISE Information Node)* une os módulos os quais trabalham para definir e gerenciar uma ontologia global integrada apresentada na rede. Além disso, o SINode realiza o processamento de enriquecimento semântico de forma semi-automática, para criar a ontologia SINode. Por sua vez, esta ontologia do SINode também está integrada com outros componente similares aos do mapeamento de ontologias usados pelos *broker agents*.

Broker agents são os pontos responsáveis por gerenciar uma visão do conhecimento manipulado pela rede, assim como a informação específica de cada SINode. Um *broker agent* trabalha como um intermediário detendo o controle direto sobre o número de SINodes, e fornecendo meios para publicar um manifesto no âmbito da rede, no tocante às informações mantidas localmente em um perfil semântico. Os *query agent* são os portadores das consultas a partir da interface do usuário até os SINodes. Eles interagem com um ou vários *broker agents* no intuito de resolver uma consulta e integrar as respostas ao usuário.

3.3.4. PEPSINT

Desenvolvido pela Universidade de Illinois, em Chicago, o PEPSINT (*Peer-to-peer Semantic INTegration*) foi proposto para realizar integração semântica em fontes heterogêneas, as quais estão em XML e RDF [CRUZ *et al.* 2004]. Este PDMS foi construído em uma arquitetura híbrida, na qual a ontologia RDF global (construída usando GAV) no super-

ponto comporta-se não apenas como um ponto de controle central sobre os pontos, mas também como um mediador de consulta de um ponto para outro.

A proposta do PEPSINT é de preservar a estrutura do domínio existente em RDF e em XML, implementando a integração semântica em nível de esquema (por meio de um processo de correspondência de esquemas) e de instância (através do processo de resposta a uma consulta).

Por ser baseado na arquitetura híbrida, o PEPSINT contém os dois tipos de pontos: super-ponto contendo a ontologia global em RDF e os pontos contendo esquemas e fontes de dados locais. Conforme ilustra a Figura 3-4, a arquitetura do PEPSINT é composta por quatro principais componentes: **XML *to* RDF *wrapper*** usado para transformar o esquema XML para um esquema RDF local, e este mapeado para a ontologia global. ***Local* XML *and* RDF *schemas*** estão presentes em pontos que possuem dados e metadados, e possuem o propósito de integração semântica. ***Global* RDF *ontology*** localizada no super-ponto com o propósito de atuar como um esquema mediador virtual integrado dos esquemas RDFs locais. Trata-se de uma simples ontologia que não possui altos níveis de axiomas. ***Mapping table*** armazena os mapeamentos entre os esquemas locais e a ontologia global.

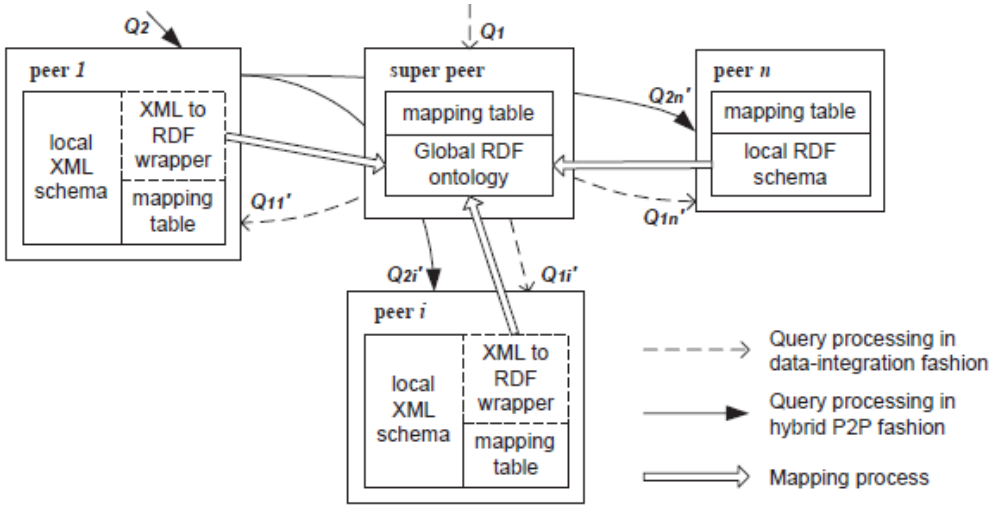


Figura 3-4-Arquitetura do PEPSINT [CRUZ *et al.* 2004]

3.3.5. Outros PDMS

Desenvolvido pelo Instituto Militar de Engenharia (IME-RJ), o Rosa-P2P [BRITO *et al.* 2005] é um PDMS concebido para compartilhar dados, cujo conteúdo é acadêmico. Trata-se da evolução do sistema ROSA (*Repository of Objects with Semantic Access*). O ROSA é um sistema

centralizado que armazena objetos de aprendizagem com conteúdos instrucionais. No entanto, o ROSA-P2P levou o ROSA a um ambiente P2P. O ROSA-P2P é baseado na arquitetura de super-ponto e faz uso de uma estrutura semântica complexa para resolver conflitos semânticos. A otimização do processamento de consulta é feito através do uso de agregações e agrupamentos de pontos.

O Humboldt Discoverer provê um índice semântico para uma arquitetura PDMS estendida (nas dimensões semântica, *Web* e qualidade) [HERSCHEL *et al.* 2005]. Esse índice é utilizado para localizar fontes de informações relevantes que não são alcançáveis por meio de caminhos de mapeamentos convencionais. O Humboldt Discoverer implementa uma infra-estrutura híbrida combinando DHT com uma rede não estruturada, cujos esquemas dos pontos são indexados por tecnologias da *Web* semântica. A dimensão semântica provê uma forma de mapeamento entre esquemas de pontos, de modo a possibilitar o compartilhamento de seus dados. A dimensão de qualidade abordada no Humboldt Discoverer influencia no processamento de consultas, pois a qualidade do resultado da consulta é diretamente dependente dos caminhos de mapeamentos providos por dois pontos. E na dimensão *Web*, métricas para localizar e realizar ranking de pontos são definidas para responder uma consulta.

Baseado em uma estrutura de acesso descentralizada, o PDMS GridVine [MAUROX *et al.* 2007] foi projetado seguindo o principio de independência de dados, separando a camada lógica, onde dados e esquemas e seus mapeamentos semânticos são gerenciados, da camada física que consiste de uma rede P2P *overlay*, a qual suporta um eficiente mecanismo de roteamento de mensagens. Essa independência permite processar operações lógicas na sobreposição semântica usando diferentes estratégias de execução. O GridVine detém um conjunto de algoritmos que organizam automaticamente a rede de mapeamento de esquemas. Além disso, segundo ABERER *et al.*(2004), esse PDMS oferece uma solução completa para implementação de SON com suporte à interoperabilidade e escalabilidade do sistema.

O Hyperion [ARENAS *et al.* 2003] foi proposto para ser um PDMS para bancos de dados relacionais (um em cada ponto). O Hyperion possui um mecanismo semelhante de *mapping table*, adotado no PEPSINT, para armazenar as conexões entre os esquemas locais nos pontos. Além disso, possui um gerenciador de consulta que faz uso do *mapping table* para reescrever uma consulta postada em termos de um esquema local. O processo de reescrita produz uma consulta que é executada sobre pontos próximos.

O Orchestra [GREEN *et al.* 2007] é um sistema de compartilhamento de dados colaborativo, construído sobre os conceitos desenvolvidos pelo Piazza. O Orchestra foi projetado para atender às necessidades dos pesquisadores de bioinformática e biomédica.

Nesse sistema, os seus pontos possuem seus próprios esquemas de dados e manipulam seus bancos de dados locais. Porém, ao contrário dos demais PDMS existentes, o Orchestra permite que atualizações sejam realizadas em suas bases de acordo com uma política de confiança [KARVOUNARAKIS *et al.* 2010]. Essas atualizações passam por um processo de importação seletiva, de atualizações, denominado reconciliação, que visa evitar conflitos com os dados existentes.

3.3.6. Comparativo dos PDMS

O Quadro 3-2 resume e compara algumas características gerais dos PDMS descritos nesta seção. O significado de cada coluna encontra-se descrito a seguir.

- PDMS: nome dado ao PDMS.
- Representação dos dados: formato de representação de dados no PDMS.
- Modelo de Arquitetura: arquitetura adotada no PDMS. Alguns PDMS aceitam a combinação de arquiteturas.
- Linguagem de consulta: linguagem de consulta adotada pelo PDMS.
- *Middleware* para ambiente distribuído: identifica o *middleware* utilizado como base para a implementação da rede P2P do PDMS.
- Outras características: aponta uma ou mais características encontradas somente em um determinado PDMS.

Quadro 3-2 - Quadro comparativo entre PDMS

PDMS	Representação dos dados	Modelo de Arquitetura	Linguagem de Consulta	*Middleware* para ambiente distribuído	Outras características
Piazza	XML, RDF e DAML+OIL	Pura	XQuery	JXTA [HALEVY *et al.* 2004]	Estrutura de índice para auxiliar na identificação e otimização da busca.
SEWASIE	Estruturado, semi-estruturado e não estruturado	Pura	Nativa do SEWASIE	JADE [FILLOTTRANI 2005]	Possui interface de consulta inteligente, já que possui um raciocinador *online*;Integração com ferramentas OLAP; Sistemas multi-agentes.
PeerDB	Estruturado	Pura	SQL	JADE	Possui agentes que monitoram estatísticas sobre pontos vizinhos e que gerenciam políticas de configuração na rede; Sistemas multi-agentes
Rosa-P2P	XML e RDF	Super-ponto	RosaSQL	Sockets [BRITO 2005]	Agrupamentos baseados em assunto e localização.
Hyperion	Relacional	Pura	SQL	JXTA (protótipo) [GIANOLLI et al. 2005]	Tabelas de mapeamentos e de expressões são usadas para armazenar conexões entre esquemas locais nos pontos.
PEPSINT	XML e RDF	Híbrida	XQL e RDQL	Não identificado	Tabelas de mapeamento para armazenar correspondência entre esquemas.
Orchestra	Estruturado e Semi-estruturado	Pura	ProQL	Não identificado	Dados provenientes de políticas de confiança e atualizações de dados de forma incremental e eficiente. Projetados para as necessidades de bioinformática e biomédica.
GridVine	Estruturado e Semi-estruturado	Híbrida	RDQL	Sockets	Possui um mecanismo dinâmico para a auto-organização da rede (auto-reorganização dos mapeamentos).
Humboldt Discoverer	Estruturado	Híbrida	SPARQL/ RDF	Não identificado	Possui índice que provê um lookup de pontos com conteúdos semelhantes e prioriza relevância em vez de eficiência.

3.4. PDMS com comunidades semânticas

Conforme podemos constatar na seção anterior, vários PDMS foram propostos na literatura. Em sua maioria, esses sistemas propõem soluções para problemas associados ao processamento de consulta e mapeamentos de esquemas. Com fins de obter melhores resultados em suas consultas, alguns PDMS propõem abordagens baseadas em semânticas, no intuito de agrupar seus pontos de forma que eles fiquem mais próximos semanticamente, ou seja, que fiquem relacionados de acordo com o conhecimento semântico comum entre eles. Nesta seção, abordaremos alguns desses PDMS e seus mecanismos para agrupar, semanticamente, seus pontos.

3.4.1. ESTEEM

O ESTEEM (*Emergent Semantics and cooperaTion in multi-knowledgE EnviroMents*) visa proporcionar uma colaboração semântica para compartilhamento de informações e de serviços através do uso de ontologias, em uma rede P2P não estruturada. Esse sistema usa técnicas que permitem que os ponto*s* se auto-organizem com base nos seus respectivos conteúdos e interesses, com o objetivo de prover estruturas homogêneas de ponto*s* e assim as comunidades semânticas. Essas técnicas são baseadas em ontologias e em correspondências entre ontologias para forçar a negociação do consenso durante o processo de formação da comunidade semântica dos ponto*s* [MONTANELLI *et al.* 2011].

Uma comunidade no ESTEEM surge de forma autônoma como um grupo de ponto*s* construído em torno de um interesse declarado baseado em um "manifesto", na forma de uma ontologia. O manifesto é formado de acordo com as preferências do ponto fundador da comunidade e é extraído da ontologia desse ponto. No manifesto está descrito o interesse semântico da comunidade e serve para que os ponto*s* tomem conhecimento dela, cabendo a eles o arbítrio de quererem participar ou não da comunidade.

Cada ponto, ao receber um manifesto, realiza a correspondência existente entre sua ontologia local e a do manifesto. A Figura 3-5 representa parte da arquitetura do ESTEEM. Nela podemos observar o manifesto que descreve uma comunidade semântica e os ponto*s* que fazem parte dessa comunidade.

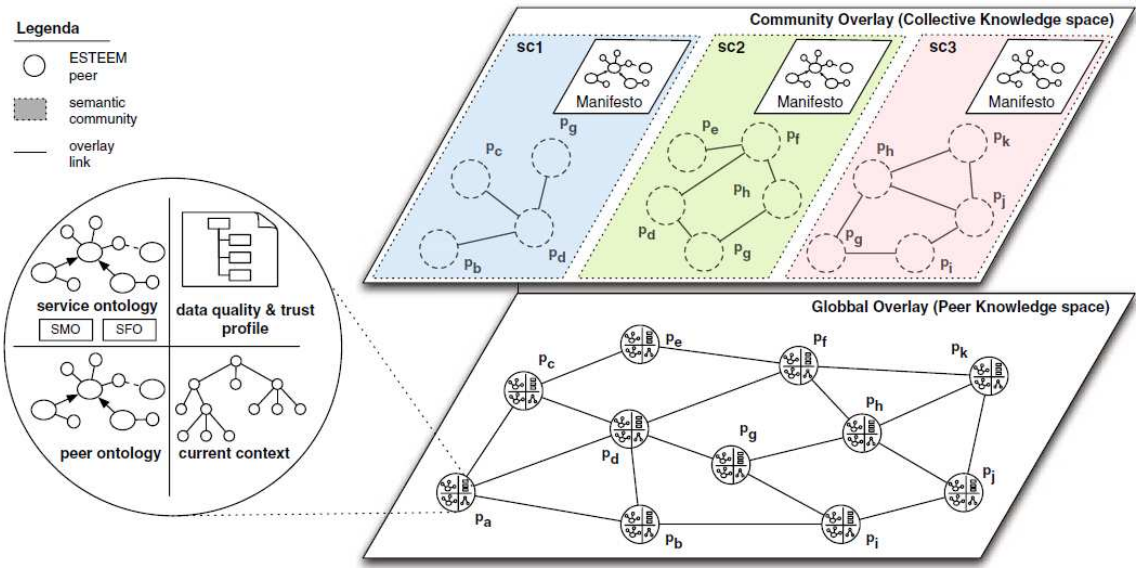


Figura 3-5 - O conhecimento do ESTEEM [MONTANELLI *et al.* 2011]

A ontologia de serviço (*service ontology*), destacada na Figura 3-5, prover uma rica descrição dos serviços dos pontos que estão disponíveis para compartilhamento. Esses serviços estão descritos através da linguagem WSDL [WSDL 2011]. De acordo com BIANCHINI *et al.* (2009), uma SMO (*Service Message Ontology*) é utilizada para adicionar semântica aos parâmetros de entrada e saída dos serviços. Já a SFO (*Service Functionality Ontology*) é utilizada para adicionar semânticas às funcionalidades.

O "atual contexto" (*context current*) trata-se de uma sub-árvore de *Context Dimension Tree* (CDT) [MONTANELLI *et al.* 2011, MONTANELLI *et al.* 2007b]. Ela descreve o perfil, seus interesses, situação e coordenadas espaço-temporal no momento da interação com a rede do ESTEEM. A CDT procura expressar várias perspectivas que determinam quais dados são importantes para determinadas situações. O fundador de uma comunidade semântica poderá associar a CDT a um contexto desejável para a comunidade. A Figura 3-6 ilustra um exemplo de uma CDT que representa o contexto de "saúde", considerado pelo ESTEEM. A área cinzenta identifica a área de interesse de um médico sobre drogas úteis para patologias presentes na região em que ele se encontra [MONTANELLI *et al.* 2011].

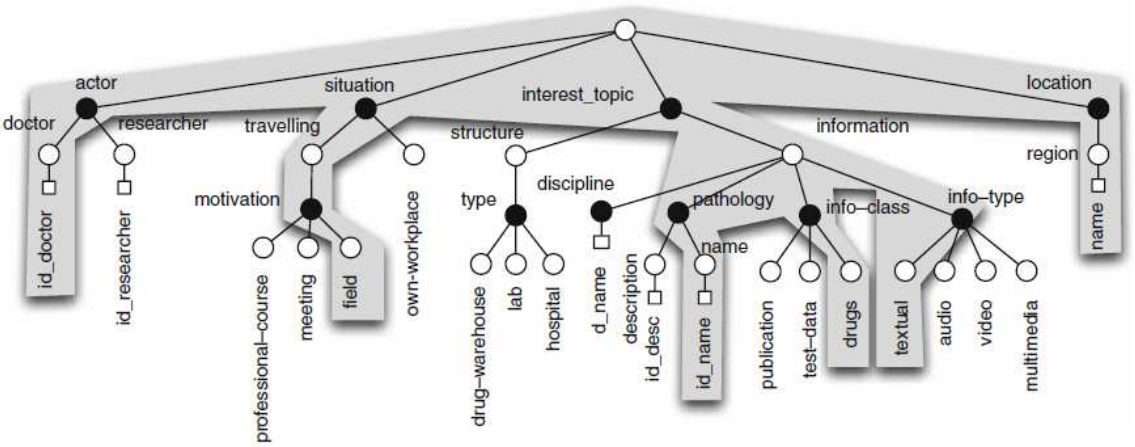


Figura 3-6 - Exemplo de uma CDT [MONTANELLI *et al.* 2011]

A qualidade do dado (*quality data*) e o perfil de confiança (*trust profile*) envolvem a computação de métricas de qualidade dos dados que são disponibilizados pelos pontos. Essas métricas referem-se aos aspectos de completude, consistência, precisão e consistência interna [MONTANELLI *et al.* 2011].

- Formação da comunidade semântica

No ESTEEM, uma comunidade semântica surge de forma autônoma, e de acordo com o interesse de um ponto, chamado de ponto fundador. O processo inicia-se através de mensagens, enviadas através da rede, para os demais pontos, convidando-os/anunciando-os na formação de uma comunidade semântica. Nessas mensagens encontram-se trechos dos manifestos, chamados *startup manifesto*, definidos ponto fundador, baseando-se em sua ontologia local. O *startup manifesto* [MONTANELLI *et al.* 2007a] é definido como a ontologia da comunidade e ela provê descrições semânticas da perspectiva do interesse da comunidade. Ao receber a mensagem, cada ponto na rede (membro candidato) avalia seu nível de interesse em participar da comunidade que está surgindo. A avaliação é feita através do resultado das correspondências analisadas entre o *startup* manifesto e a ontologia local do membro candidato. Se existir correspondência entre os conceitos, o membro candidato poderá enviar uma mensagem de resposta ao ponto fundador, informando o seu interesse em participar dessa comunidade, junto com uma parte de sua ontologia local.

Ao receber as mensagens de interesses dos membros candidatos, o ponto fundador analisa, também, as ontologias de cada candidato, avaliando as correspondências semânticas existentes entre a porção da ontologia do candidato com o seu conhecimento semântico. Se os

conceitos analisados através das correspondências tiverem um resultado satisfatório, mensagens de aprovação serão enviadas do fundador aos membros candidatos. Em seguida, os candidatos aprovados constituirão a comunidade semântica que será formada por interesses comuns e estarão organizados em uma estrutura *overlay* [MONTANELLI *et al.* 2007a].

3.4.2. OntoZilla

Buscando uma melhor sinergia entre P2P e ontologias, o OntoZilla [JOUNGA *et al.* 2009] foi concebido com o intuito de melhorar o mecanismo de busca, integração e interoperabilidade em ambientes P2P. Um ponto poderá ter interesse em algum serviço particular ou prover algum conhecimento sobre um domínio específico. Uma vez detectados pontos com interesses em comum, esses pontos têm relacionamentos estabelecidos através de ontologias, formando *clusters*. Esses relacionamentos são *links* semânticos estabelecidos entre os pontos, e as consultas submetidas ao sistema poderão ser roteadas através desses *links* para encontrar os pontos que possuam a informação ou os serviços adequados e desejados. Considere o seguinte exemplo, um ponto poderá ter conhecimento sobre "bancos de dados" e outro ponto com conhecimento sobre "banco de dados geográficos". Esses dois pontos poderão estar relacionados de forma hierárquica. Com relação a serviços, um ponto poderá fornecer um serviço sobre "reserva de hotel" e outro ponto fornecer "agente de viagem", logo esses dois pontos também podem estar relacionados.

Os relacionamentos e as auto-descrições das informações e serviços ofertados por cada ponto são descritos através de ontologias. Devido à natureza dinâmica do ambiente P2P, onde os pontos poderão entrar e sair do sistema a qualquer momento, os pontos pertencentes ao OntoZilla trocam informações, periodicamente, com o intuito de ajustarem os *links* semânticos de acordo com a alteração na rede.

- Formação de *clusters*

No OntoZilla, os interesses particulares de um ponto estão agrupados em grupos de interesses especiais - SIG (*Special Interest Group*). Portanto, pontos com interesses similares estarão agrupados no mesmo SIG. Dentro de cada SIG, os pontos empregam algum sistema que hierarquicamente classifica os pontos dentro de classes de interesses. Cada classe representa um *cluster* de pontos que compartilham os mesmos conceitos.

Um novo ponto *Pt* deseja entrar no sistema, logo precisa ser determinado qual o SIG que *Pt* pertencerá. Logo, aleatoriamente *Pt* escolherá um ponto qualquer Y, na rede. Através

de Y, *Pt* realizará um *broadcast* com seu nome, endereço IP e seu interesse descrito em uma ontologia. Ao encontrar os pontos que interessam a *Pt*, mensagens de respostas provenientes dos pontos que fazem parte do grupo de interesse de *Pt* (SG_{Pt}), são enviadas a esse ponto. Em seguida, *Pt* escolhe um ponto Z pertencentes a SG_{Pt} e Z ajudará *Pt* a entrar no *cluster*. Através dos *links* de Z, *Pt* encontrará seu *cluster* dentro do SG_{Pt} sem a necessidade de um *flooding* na rede. Caso *Pt* não encontre um SIG, ele mesmo formará um SIG e um *cluster* que mais corresponde aos seus interesses e conceitos.

A Figura 3-7 ilustra um exemplo de uma SIG sobre "Computadores", cuja classificação dos seus *clusters* é baseada no sistema de classificação da ACM, o CSS (*Computing Classification System*) [ACM 2011]. Os pontos pertencentes ao mesmo *cluster* estão envolvidos por linhas tracejadas. Cada *cluster* recebe um nome mais significativo de classe. A Figura 3-7 ilustra as classes (*clusters*) P, X, E, D, e C chamadas de "Computer System Organization(1)","Computer networks (2)", "Distributed systems (3)", "Network management(4)", "Network architecture & design (3)", respectivamente.

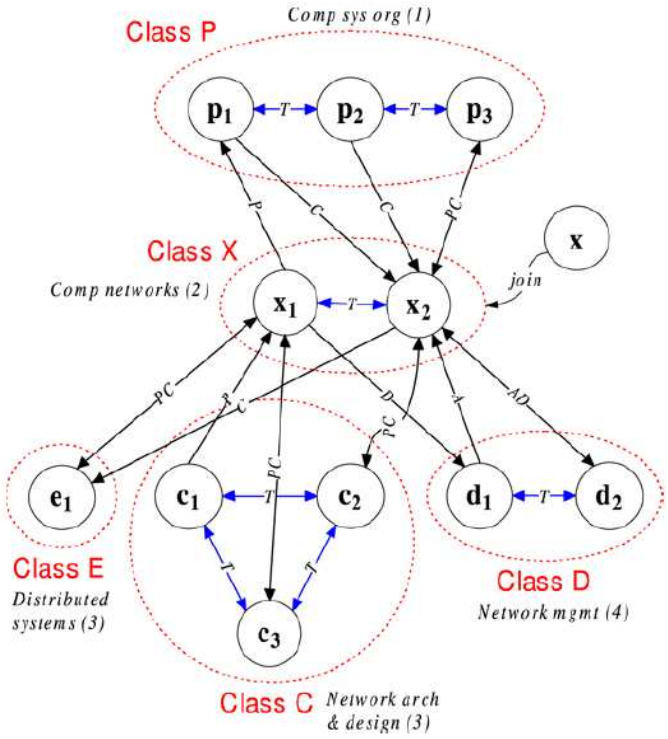


Figura 3-7 - *Clusters* classificados de acordo com a ACM CCS [JOUNG *et al.* 2009]

Dentro de um mesmo SIG, os pontos poderão estar relacionados, semanticamente, de acordo com os seguintes tipos de *links*, conforme ilustra a Figura 3-7:

- *Twin link*: relação entre pontos pertencentes ao mesmo *cluster* é chamada de *twin link*(T).

- *Parent e Child links*: *links* utilizados para estabelecer a relação hierárquica entre os pontos. A relação semântica que um ponto é filho de outro, em uma hierarquia, é chamada de *child link e denotada pela letra C.* Entretanto, a relação inversa a C, chamada de *parent link,* é denotada pela letra P. Porém, uma relação bidirecional poderá ser estabelecida entre um ponto pai e seu filho, denotada pela relação PC.
- *Ancestor* e *Descendant links:* os *clusters* em um SIG poderão ser criados ou destruídos a qualquer momento. Portanto, poderá existir momentos em que pontos que antes tinham relações hierárquicas, com outros pontos de *clusters* que deixaram de existir, precisam estabelecer relações com outros pontos de seus ancestrais (A) ou descendentes (D) mais próximos. Porém, uma relação bidirecional poderá ser estabelecida entre um ponto ancestral e seu descendente, denotada pela relação AD.

3.4.3. SUNRISE

Com a proposta de possuir uma completa infra-estrutura, a qual estende cada ponto com funcionalidades para capturar uma aproximação semântica proveniente de um esquema heterogêneo, e explorá-lo para uma organização de uma rede semântica e roteamento de consulta, foi proposto o SUNRISE (*System for Unified Network Routing, Indexing and Semantic Exploration*) [MANDREOLI *et al.* 2008]. Este PDMS foi desenvolvido pela Universidade de Modena e Reggio Emilia, na Itália.

O SUNRISE dá suporte à criação e manutenção da organização de uma rede flexível que gera *clusters* dos pontos que estão semanticamente relacionados (de forma semelhante ao iXPEER, porém em *Semantic Overlay Networks*-SON). Além disso, o sistema fornece um conjunto de técnicas que podem ser usadas para a efetiva e eficiente exploração da rede como pode ser observado em [MANDREOLI *et al.* 2007].

A Figura 3-8 ilustra duas SON com dados relacionados a “cinema” e cada ponto está representado por um tópico principal de interesse derivado do seu esquema. Podemos observar que alguns pontos são monotemáticos (só lidam com um assunto), como é o caso dos pontos B e F. Outros pontos estão preocupados com ambos os temas, como o caso do ponto C.

Podemos observar que um ponto pode estar na base de uma ou mais SON, e para um PDMS esta operação é um desafio devido à falta de um entendimento comum entre os dicionários dos pontos locais. Para isso, o SUNRISE assiste a entrada de novos pontos em duas fases: realiza a escolha da SON mais próxima; escolhida a SON, dentro dela uma seleção refinada dos melhores vizinhos entre os pontos mais relacionados semanticamente é realizada [LODI *et al.* 2008a].

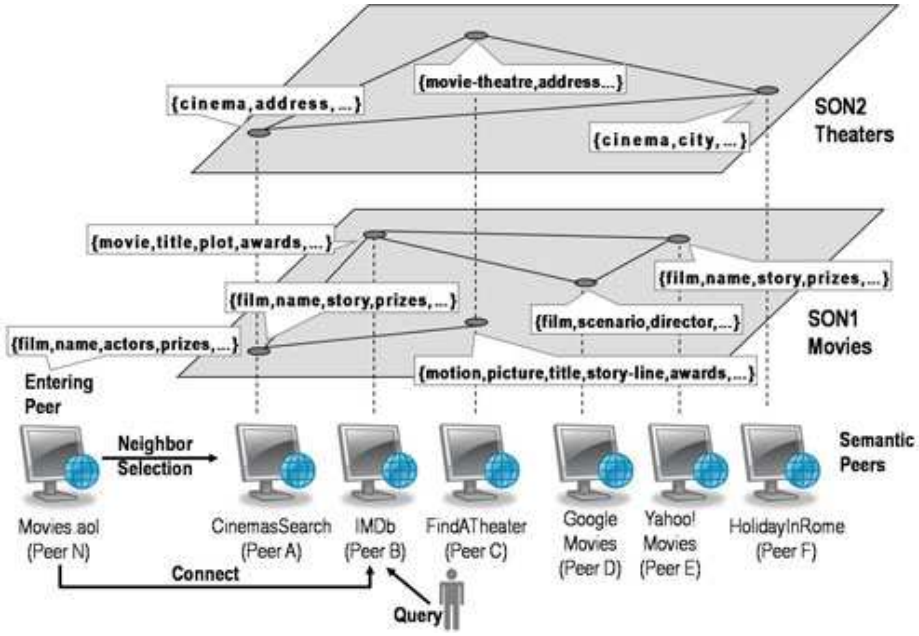


Figura 3-8 - Exemplo da organização da rede no SUNRISE [MANDREOLI *et al.* 2008]

As descrições resumidas sobre as SON disponíveis na rede estão descritas em uma estrutura chamada *Access Point Structure* (APS), a fim de ajudar os novos pontos à decidirem que SON participar ou se for o caso, formar novas SON.

- Formação de *clusters*

Quando um novo ponto deseja fazer parte do sistema, ele explicita seus esquemas através de semântica. Baseado nos esquemas fornecidos pelo ponto, o SUNRISE o auxilia a encontrar os seus vizinhos semânticos partindo da análise da APS, conforme ilustra a Figura 3-9. Em seguida, sistema seleciona a(s) SON que possui(em) mais proximidade semântica (conceitos mais próximos) com o novo ponto. Dependendo dos conceitos, descritos através de XML *Schema*, verificados no novo ponto, este poderá se conectar a mais de uma SON.

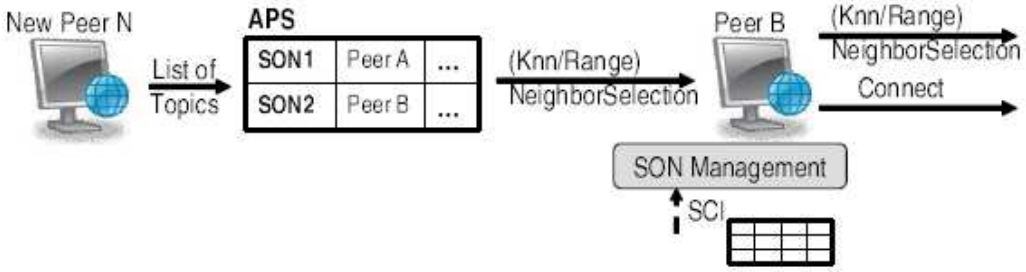


Figura 3-9 - Ação desempenhada para encontrar os vizinhos semânticos de um ponto [MANDREOLI *et al.* 2008]

De acordo com a Figura 3-9, o sistema identificou que a SON2 possui proximidade semântica com o novo ponto. Logo, este navegará através dos *links* que partem do ponto "PeerB". Para encontrar, nesses *links*, os pontos que possuem semântica mais próxima ao do novo ponto, as políticas de *range-based* e k-NN são utilizadas [LODI *et al.* 2008a]. Com o *range-based* um ponto é conectado a outros de acordo com uma faixa de similaridade semântica. Em seguida, nesta faixa estabelecida, os *k* pontos que estiverem semanticamente mais próximos serão escolhidos (política k-NN) para que o novo ponto seja conectado.

3.4.4. GrouPeer

O GrouPeer é um PDMS que faz uso de uma rede P2P não estruturada. Sabendo que pontos ricos em informações podem ficar invisíveis por consultas emitidas pela rede, devido ao repetitivo processo de reformulação de consulta que é realizado, o GrouPeer propõe uma estratégia que procura evitar que esses pontos fiquem invisíveis a essas consultas.

Segundo KANTERE *et al*. (2009), os ponto*s* têm mais chances de receber e responder a uma consulta, se eles receberem a consulta original. O GroupPeer propõe um procedimento no qual evita as sucessivas reescritas que uma consulta poderá sofrer ao passar pelos ponto*s*. O procedimento visa fazer com que um ponto possa aprender, gradualmente, com outro ponto os esquemas e interesses similares que eles têm. Esse aprendizado é realizado através de consultas submetidas, entre os pares de ponto*s*, e em seguida, na análise dos seus resultados. Com esses resultados, um mapeamento entre os pares de ponto*s* é formado. Esses mapeamentos encapsulam o interesse comum dos ponto*s*, desde que eles façam referência a partes vitais do esquema os quais expressam e respondem às consultas [KANTERE *et al.* 2008, KANTERE *et al.* 2009]. Se os pontos decidirem tornarem-se conhecidos, esses mapeamentos estão prontos para fazerem uso de linguagem especifica para sua comunicação e diminuir a carga de trabalho do administrador para a criação de mapeamentos para um novo entendimento.

- Formação de *clusters*

O método utilizado pelo GrouPeer, para a descoberta de pontos remotos e relevantes, é propagar, ao longo do caminho percorrido por uma consulta, tanto a consulta original submetida, mas como também as sucessivas versões que vão sendo reescritas. Neste caso, os pontos receberão um conjunto de consultas e decidirão qual delas eles irão responder. Para isso, cada ponto possui um mecanismo de reescrita para reescrever as consultas expressas em

seus esquemas baseados nos respectivos conhecimentos. Os pontos possuem, também, uma ferramenta usada para compreender e traduzir as consultas ou parte delas expressadas nos esquemas para os quais os mapeamentos não estão disponíveis.

Sabendo-se que alguns conceitos poderão ser perdidos em uma consulta, devido às várias reformulações que ela poderá sofrer, o GrouPeer mantém os conceitos eliminados e tenta correspondê-los com os próximos esquemas. A Figura 3-10 ilustra uma situação onde uma consulta original Q_{orig} foi submetida e que ela, ao longo do seu percurso, é várias vezes reescrita, devido aos mapeamentos existentes em cada ponto. Por essa figura, verificamos que o ponto P2 recebe a consulta original Q_{orig} e a consulta reescrita $Q_{sr\_M1,2}$, resultante do mapeamento $M_{1,2}$, entre P1 e P2.

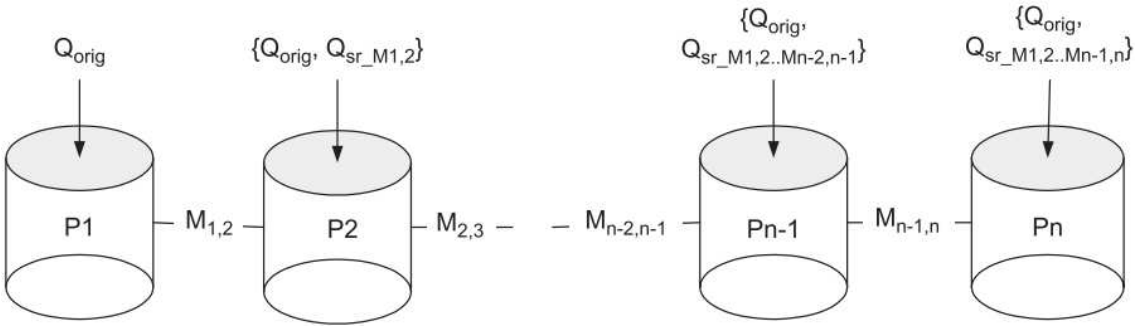


Figura 3-10 - Propagação da consulta postada pelo ponto P1 ao longo dos *n* pontos conhecidos [KANTERE *et al.* 2009]

Ao chegarem as duas consultas, a original (Q_{orig}) e a previamente reescrita ($Q_{sr\_previous}$), a um ponto, cada uma delas sofrerá novas reescritas de acordo com as correspondências desse ponto. A Q_{orig} será rescrita para Q_{ar} e a $Q_{sr\_previous}$ para Q_{sra}, conforme representa a Figura 3-11. Em seguida, as duas consultas geradas Q_{ar} e Q_{sra} são comparadas com Q_{orig}, através do uso de uma função de similaridade descrita em [Kantere *et al.* 2009]. Depois dessa comparação, a consulta que tiver o maior grau de similaridade com a consulta original, será a consulta a ser respondida pelo ponto. Ao receber a resposta da consulta, o ponto que submeteu a consulta original irá avaliar a satisfatibilidade da resposta recebida e envia um *feedback* sobre a versão da consulta respondida, ao ponto que a respondeu. Com esse *feedback*, o ponto que respondeu poderá avaliar as suas boas e más ações de reescritas que ele gerou. Baseado nisso, o ponto que submeteu a consulta original pode decidir que ele tem interesse comum com o ponto remoto e convidá-lo para fazer parte de um *cluster* que compartilha os mesmos interesses. Nesse novo *cluster*, os mapeamentos serão criados baseados na comunicação que existiu entre esses pontos.

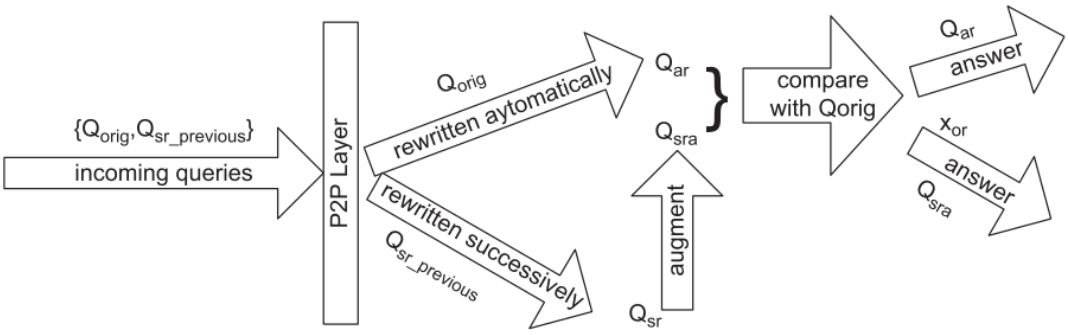


Figura 3-11 – Representação de um processamento de consulta em um ponto [KANTERE *et al.* 2009]

3.4.5. SPEED

Desenvolvido pela Universidade Federal de Pernambuco, o SPEED (*Semantic Peer-to-peer Data Management System*) é um PDMS que possui uma estrutura mista: DHT, super-ponto e não estruturada [PIRES 2009]. O SPEED realiza a "*cluster*ização" de ponto*s* que possuem similaridades semânticas, com o intuito de facilitar o mapeamento semântico entre ponto*s* e, conseqüentemente, facilitando o processamento de consulta sobre o grande volume de pontos de dados. Em seguida, os *clusters* formados, que pertencerem ao mesmo domínio, farão parte de uma comunidade semântica.

A base do desenvolvimento deste trabalho será o SPEED, devido ao fato desse sistema propiciar um melhor entendimento do problema de balanceamento de carga o qual desejamos resolver. Portanto, abordaremos com mais detalhes esse sistema, assim como a formação dos seus *clusters* e comunidades semânticas, no Capítulo 5.

3.4.6. Comparativo entre os PDMS com comunidades semânticas

O Quadro 3-3 resume e compara algumas características gerais dos PDMS descritos nesta seção. O significado de cada coluna encontra-se descrito a seguir.

- PDMS: nome dado ao PDMS.
- Modelo de Arquitetura: arquitetura adotada no PDMS
- Formação da comunidade semântica: estruturas ou mecanismos adotados, exclusivamente pelo PDMS, para a formação da comunidade semântica.

- Recurso compartilhado: o recurso compartilhado pelos pontos. Alguns desses PDMS compartilham tanto serviços quanto dados.
- Outras características: aponta uma ou mais características encontradas somente em um determinado PDMS.

Quadro 3-3 - Quadro comparativo entre PDMS com comunidades semânticas

PDMS	Modelo de Arquitetura	Formação da comunidade semântica	Recurso compartilhado	Outras características
ESTEEM	Pura	Os pontos com autonomia para a formação da comunidade; Manifestos através de ontologias.	Serviços e dados	Métricas de qualidade; Utilização de informações contextuais.
OntoZilla	Super-ponto	Grupos de interesses especiais (SIG) pré-definidos (representados através de ontologias)	Serviços e dados	Agrupamento hierárquico de *clusters*;
Sunrise	Pura	*Access Point Structure* (APS), descritos através de ontologias	Dados	Políticas baseadas em range e k-NN
GrouPeer	Pura	Os pontos possuem autonomia para a formação da comunidade; Agrupamento de acordo com a consulta.	Dados	Mecanismo para diminuir a degradação semântica de um consulta

3.5. Considerações Finais

Neste capítulo foram apresentados conceitos referentes à PDMS, enfatizando a realidade e importância de seu uso nos dias atuais. Foram descritas aqui as características arquiteturais de alguns PDMS. Exemplos de implementações desses sistemas foram descritos, considerando os seus aspectos estruturais e características específicas. Um quadro resumido comparou alguns dos PDMS pesquisados. Além disso, foi feito um levantamento sobre PDMS

que realizam agrupamentos (*clusters*) de pontos, de acordo com os esquemas de cada um desses pontos.

O entendimento da importância e a forma na qual alguns PDMS realiza esses agrupamentos é de fundamental importância para este trabalho. No próximo capítulo, abordaremos o problema de balanceamento de carga enfrentado pelos sistemas P2P e PDMS baseados em *clusters* e, mostraremos as soluções existentes na literatura para esse problema.

Capítulo 4

Balanceamento de Carga em Sistemas P2P e em PDMS

Em sua natureza, os sistemas P2P são caracterizados por possuírem um controle descentralizado, robustez, escalabilidade, estabilidade entre outras. Diante dessas características, vem crescendo o interesse de pesquisadores sobre esses sistemas. A natureza dos sistemas P2P oferece uma área vasta para pesquisas, não só no tocante ao compartilhamento de conteúdo, mas também em outros domínios que incluem infra-estrutura, busca, colaboração, roteamento, balanceamento de carga, segurança, tolerância a falhas dentre outros.

Apesar da grande quantidade de usuários que utilizam sistemas P2P, a capacidade desses sistemas proverem serviços com qualidade é questionada [QIAO 2009], ou seja, serviços que correspondam às expectativas dos seus usuários. Um balanceamento de carga pode prover serviços com pequena taxa de falha e com um melhor desempenho, com isso a qualidade do serviço poderá ser melhorada. Além disso, sistemas baseados em *clusters* têm sido adotados para os serviços que são tolerantes a falha. Porém, apesar de um sistema de *cluster* ser adotado para melhorar a confiabilidade e robustez de um sistema P2P, o problema do desbalanceamento ainda persiste por causa da heterogeneidade dos pontos (como larguras de banda, processadores, capacidade de armazenamentos diferentes, entre outros) e do total de suas requisições.

Podemos observar que além de P2P, outros ambientes como *grid* e redes de sensores sem fio também vêm se preocupando com aspectos de balanceamento de carga, onde *clusters* são formados, conforme mostram os trabalhos [SURI *et al.* 2010, YAGOUBI *et al*. 2010, ISHMANOV *et al.* 2009, HUANG *et al.* 2008].

Alguns dos sistemas que trabalham com *clusters* apresentam dois níveis de conexões: intra-*cluster* e inter-*cluster*. Nesses sistemas o desbalanceamento pode ocorrer nesses dois níveis, portanto soluções de balanceamento de carga intra e inter-*cluster* têm sido pesquisadas.

Este capítulo apresenta o problema do não balanceamento de carga e algumas estratégias que foram propostas pela comunidade científica. Porém, o foco deste trabalho está

nos sistemas P2P e PDMS baseados em *clusters* e que possuem os dois níveis de conexões citados no parágrafo anterior.

4.1. Balanceamento de Carga em Sistemas P2P

Balanceamento de carga é uma das propriedades dos sistemas P2P. Segundo FIORANO (2003), o balanceamento de carga visa lidar com a distribuição de solicitações entre servidores distintos, a fim de melhorar o desempenho da rede, aliviando aqueles pontos que possuem sobrecarga (*hot peers*). Uma vez que os *hot peers* tornem-se um gargalo, eles tendem a aumentar o tempo de resposta ao usuário e, significativamente, degradam o desempenho do sistema [AYYASAMY et al. 2010].

De acordo com FIORANO (2003), muitas das estratégias de balanceamento de carga são para as topologias híbrida e de super-ponto. Na primeira, vários servidores podem ser utilizados para melhor distribuir o processamento de consultas. De acordo com PIRES (2007), em se tratando da topologia de super-ponto, os aspectos a serem considerados para manter um bom balanceamento de carga são:

- Quantidade de super-pontos

Um grande número de super-pontos tendem a provocar um aumento na quantidade de mensagens trocadas na rede, durante o roteamento de consulta, de forma que a topologia venha a ter um comportamento similar ao de uma topologia pura não-estruturada. Por outro lado, um número muito reduzido de super-pontos pode fazer com que eles fiquem sobrecarregados por lidar com uma quantidade excessiva de pontos. Portanto, a quantidade de super-pontos deve ser adequada [ZHUANG *et al.* 2004].

- Métricas para a escolha do super-ponto

Métricas bem definidas precisam ser estabelecidas para que a escolha de um super-ponto possa ser realizada. Essas métricas deverão considerar o espaço para armazenamento, disponibilidade e largura de banda presentes nos possíveis super-pontos [JOHNSTONE *et al.* 2005, MONTRESSOR 2004].

Quando um sistema P2P é projetado baseando-se em uma arquitetura, onde pontos tendem a assumir alguns papéis, a tolerância a falhas torna-se mais susceptível a acontecer

[PIRES 2007]. Considerando uma arquitetura de super-pontos, caso um super-ponto falhe, este não poderá comprometer os demais pontos. Neste caso, estratégias de tolerância a falhas devem estar pré-definidas nos pontos, para que o sistema possa imediatamente eleger um novo super-ponto. Algumas pesquisas apontam estratégias para tolerância a falhas para sistemas que possuem super-pontos. Em [BRITO 2005, JOHNSTONE *et al.* 2005] são propostos que a eleição do super-ponto seja semelhante às métricas definidas no parágrafo anterior. Já JOUNGA *et al.* (2009) consideram, também, o número mínimo de *links* que o ponto necessita estar conectado.

Ainda se tratando de tolerância a falhas, LI *et al.* (2004) sugerem a utilização de servidores de *backup*. Neste caso, os metadados armazenados no super-ponto são periodicamente replicados em um super-ponto de *backup*. Caso o super-ponto venha a falhar, o super-ponto de *backup* assume o papel do super-ponto. ROUSE *et al.* (2006) propõem a presença de vários super-pontos em um mesmo *cluster,* visando melhorias no balanceamento de carga e na tolerância a falhas.

Outras propriedades dos sistemas P2P, além do balanceamento de carga e tolerância a falha, podem ser encontradas em [AHMED *et al.* 2011]. Porém, neste trabalho nos focaremos apenas no balancemanto de carga.

4.1.1. Balanceamento de Clusters

Balanceamento de carga é uma das questões abordadas em redes P2P *overlays* contemporâneas que incluem DHT [GODFREY *et al.* 2004, KARGER *et al.* 2004, PITOURA *et al.* 2006], super-pontos [MONTRESOR 2004] e *clusters* [AYYASAMY *et al.* 2010, JOUNGA *et al.* 2009, GAROFALAKIS *et al.* 2009, MONDAL 2006].

Em se tratando de sistemas P2P baseados em *cluster*, dois tipos de balanceamentos de carga fazem-se necessários: balanceamento *intra-cluster* e o balanceamento *inter-cluster* [AYYASAMY *et al.* 2010]. A seguir, um levantamento sobre soluções para balanceamento de carga intra e inter-*cluster* serão apresentados e no final desta seção, um quadro resumirá os aspectos mais relevantes de cada solução apresentada.

4.1.1.1. Balanceamento de Carga Intra-*Cluster*

No caso de intra-*cluster*, o balanceamento de carga é realizado dentro de um *cluster* particular. De acordo com TRIANTAFFILOU *et al.* (2003), um balanceamento intra-*cluster*

significa que, para cada *cluster*, todos os seus pontos devem receber em média (ou aproximadamente) o mesmo número de requisições, do total de requisições que chegam ao *cluster*. A seguir, apresentaremos as soluções propostas por diversos autores para a realização do balanceamento de carga intra-*cluster*.

- Popularidade dos pontos

Em [MICHAIL 2002] cada *cluster* replica seus documentos entre pontos, no intuito de garantir o balanceamento de carga. Cada ponto de um *cluster* é autorizado a criar réplicas para os documentos que ele detém. Cabe ao ponto decidir onde e quando replicar o documento, baseado no conhecimento sobre a carga de tarefas dos membros do *cluster*. Esse conhecimento é obtido através do mecanismo do relato de popularidade de cada ponto. A popularidade de cada ponto é o percentual de sua carga de trabalho em relação à carga de trabalho do sistema. Além disso, para saber se houve um desbalanceamento de carga, periodicamente os pontos verificam o índice de equilíbrio dos *clusters* (*fairness* - métrica utilizada para avaliar o estado de desbalanceamento do sistema) que é um valor compreendido entre 0 e 1. A medida que o índice de equilíbrio aproxima-se do valor 1, o *cluster* necessita ser balanceado. Mas detalhes deste índice pode ser encontrado em [MICHAIL 2002], [TRIANTAFILLOU 2003, JAIN *et al.* 1984].

A replicação de documentos e redirecionamento de requisições são as principais ações do balanceamento de carga intra-*cluster* apresentado por MICHAIL (2002). Todas as requisições de consultas que chegam a um *cluster* através de um ponto *i* são redirecionadas ou não para outros pontos de acordo com a capacidade de processamento dos mesmos. Todos os pontos em um *cluster* podem processar consultas, devido ao repositório compartilhado de metadados que é mantido entre os pontos. A probabilidade que uma consulta seja respondida por um ponto *i* é dada pela fórmula apresentada na Figura 4-1 - Fórmula para o cálculo da probabilidade de um ponto responder uma consulta [MICHAIL 2002]Figura 4-1.

$$p(i) = \frac{pc_i}{\sum_{j=1}^{N} pc_j}$$

Figura 4-1 - Fórmula para o cálculo da probabilidade de um ponto responder uma consulta [MICHAIL 2002]

Onde pc_i é a capacidade de processamento do ponto *i* e *N* é o número de pontos no *clusters*. Se $\{P_1, ..., P_n\}$ é o conjunto de pontos que contêm uma cópia dos documentos requisitados em

uma consulta, então cada documento é recuperado de um ponto *i*, com a probabilidade dada pela fórmula da Figura 4-1.

- Seleção aleatória de pontos

Quando uma consulta chega a qualquer ponto de um *cluster*, ela será respondida localmente a partir do ponto consultado. Porém, caso a consulta não possa ser respondida localmente, um ponto será selecionado aleatoriamente, considerando alguns metadados, para respondê-la. A seleção aleatória de pontos pode garantir que os pontos do *cluster* obtenham um compartilhamento igual de sua carga de trabalho sobre seu *cluster*. Assim, se a consulta for encaminhada para um ponto, escolhido aleatoriamente, então a carga será balanceada entre os pontos do *cluster*. Logo, o balanceamento de carga intra-*cluster* será alcançado [TRIANTAFFILOU *et al.* 2003].

- Tomada de decisão centralizada

A decisão de quando disparar o mecanismo de balanceamento intra-*cluster*, detectar pontos sobrecarregados e a quantidade de dados a serem replicados ou migrados são tarefas críticas para o desempenho do sistema. Para resolver essas tarefas, MONDAL *et al.* (2003) adotaram uma abordagem chamada "tomada de decisão centralizada" para realizar o balanceamento intra-*cluster*. Nessa abordagem, cada ponto envia, periodicamente, mensagens contendo estatísticas de suas tarefas executadas para o *líder* do *cluster* (um super-ponto). O líder do *cluster* é um ponto responsável por coordenar as atividades dos demais pontos do *cluster*, além de gerenciar as informações sobre o conjunto de categorias de seu *cluster*, assim como as dos seus *clusters* vizinhos.

Em [MONDAL *et al.* 2003], um líder de *cluster* C_i recebe periódicamente informações sobre as cargas L_i e dispobibilidade de espaço em disco D_i dos pontos. O líder cria uma lista ordenada pelas cargas L_i dos pontos, onde o primeiro elemento da lista é o ponto mais sobrecarregado. Considerando a lista com *n* elementos, e dentre os últimos *n/2* pontos da lista cujos respectivos valores de D_i forem menores que um limiar estabelecido pelo sistema, esses pontos serão removidos da lista. Logo, o balanceamento de carga se dará pela migração ou remoção dos dados mais procurados do primeiro ponto da lista para o último lugar , do segundo para o penúltimo e assim sucessívamente. E os dados só serão movidos (migrados ou replicados) se a diferença de carga entre os pontos excederem o limiar pré-estabelecido no sistema.

MONDAL *et al.* (2003) acreditam que se a detecção da necessidade de um balanceamento de carga fosse realizada cada vez que um ponto entrasse/saísse do sistema, isso iria resultar em condições indesejáveis para se trabalhar (pontos se preocupando mais com o balanceamento de carga do que com seu trabalho útil). Por isso, a verificação é realizada em determinados intervalos de tempo.

- Seleção de pontos

GAROFALAKIS *et al.* (2009) analisam dois pontos que para eles são as principais razões para um desbalanceamento em uma distribuição de carga. O primeiro está relacionado com a preferência do usuário para *downloading* sobre pontos com alta largura de banda. A segunda razão está relacionada com pontos que atuam como *free-rides,* ou de uma forma geral pontos que compartilham um pequeno número de documentos ou documentos com baixa popularidade.

Para tratar a primeiro problema citado anteriormente, um super-ponto é utilizado para direcionar a consulta de um usuário para um ponto selecionado pelo próprio sistema. A seleção desse ponto é baseada nas informações sobre a largura de banda de conexão dos pontos ou pela carga recebida até então por eles, lidando com estratégias de respostas que podem ser encontradas com mais detalhes em [GAROFALAKIS *et al.* 2009].

Com relação ao segundo problema, a estratégia de replicação é adotada. Neste caso, os documentos mais populares são replicados para um *cache* local de cada ponto. O algoritmo utilizado por GAROFALAKIS *et al.* (2009) toma decisões baseadas nas seguintes questões: o que replicar? Quantos documentos serão replicados? Quais pontos receberão uma réplica? Quantas réplicas por ponto? De onde as réplicas são transferidas?

Além das questões anteriores, métricas são definidas para auxiliar o sistema a identificar situações de desbalanceamento e no tocante à qualidade do serviço. As métricas definidas em [GAROFALAKIS *et al.* 2009] são:

- Índice de Equilíbrio *(Fairness Index*-FI): métrica que mostra a distribuição de carga entre pontos em um *cluster*. Esse índice é baseado no que foi definido em [JAIN *et al.* 1984] que considera os recursos alocados pelo montande de usuários. Quanto mais o índice se aproximar do valor inteiro um, mais o balanceamento torna-se necessário.
- Qualidade de disponibilidade *(Quality of Availability)*: fração das requisições aceitas para a transferência de dados sobre o total de pedidos de transferência de dados.

- *Throughput*: número total de documentos transferidos, sem considerar os documentos que tiveram suas transferências iniciadas pelo processo de replicação.
- *Tamanho dos dados replicados (Size of Replicated Data)*: além do custo da transferência dos dados que foram replicados, ainda existe o custo para armazená-los. O tamanho total de dados replicados pode ser comparado ao tamanho total de documentos disponíveis para determinar o peso desse custo do ponto de vista do sistema. Segundo GAROFALAKIS *et al.* (2009), a avaliação média dos documentos replicados por pontos é importante também pois ela mostra a distribuição por cada ponto.
- Acertos do *cache* local *(Local Cache Hits)*: métrica definida pela razão entre o número de documentos requisitados e que tiveram suas resquisições antendidas pelos documentos que já foram transferidos para o *cache* local durante a replicação.

o Listagem de pontos segundo sua carga

De forma similar a encontrada em [MONDAL *et al.* 2003], AYYSAMY *et al.* (2010) também levantam questões de quando detectar e a quantidade de dados a serem replicados. A necessidade de um balanceamento é detectada pelo líder do *cluster*, assim como em [MODAL *et al.* 2003]. A cada intervalo de tempo os pontos enviam informações sobre seus espaços em disco e estatísticas sobre os seus trabalhos de carga aos líderes dos seus *clusters*. Baseado nas cargas desses pontos, o líder do ponto cria uma lista cujo primeiro elemento é o ponto que possui a maior carga. Considerando N o tamanho dessa lista, serão removidos os últimos N/2 elementos da lista, cujo espaço de armazenamento seja inferior ao limiar estabelecido pelo sistema. O balanceamento é obtido através de uma replicação de dados na lista da mesma forma que ocorre em [MODAL *et al.* 2003].

4.1.2. Balanceamento de Carga Inter-*Cluster*

No inter-*cluster*, o balanceamento de carga é realizado entre os *clusters* presentes no sistema. Uma definição mais formal do problema do balanceamento de carga inter-*cluster* pode ser encontrada em [TRIANTAFFILOU *et al.* 2003].

A seguir, apresentaremos as soluções propostas por diversos autores para a realização do balanceamento de carga inter-*cluster*.

o Migração de pontos

Mecanimos de monitoramento e de relatos de popularidades, como vistos anteriormente quando apresentamos o balanceamento de carga intra-*cluster* de MICHAIL(2002), provêem aos pontos do sistema uma forma de alerta sobre o balanceamento de carga do sistema. Cada ponto verifica, periodicamente, se uma ação de balanceamento de carga inter-*cluster* deverá ser desempenhada. Se a necessidade de um balanceamento existir, pontos poderão ser movidos para outros *clusters* através de uma operação chamada "migração de ponto".

Uma alternativa para balancear a carga é o reparticionamento de pontos dentro de novos *clusters*. Contudo, com essa alternativa é difícil garantir que o novo particionamento terá proximidade com a partição original [MICHAIL 2002]. No caso da nova partição derivar de uma velha, o custo de grande transferência de dados torna-se inaceitável. Uma estratégia alternativa para modificar a distribuição de carga é migrar os pontos, distribuindo-as entre os *clusters*.

A migração do ponto necessita apenas dos metadados dos pontos para serem transmitidos e pode ser considerada uma operação de baixo custo, em relação ao montante de dados que é movido durante uma operação normal do sistema. Cabe ao líder de cada *cluster* calcular a carga que deverá ser movida para outros *clusters*. O tamanho atual do sistema, isto é o número de *clusters*, é verificado antes da escolha do algoritmo de distribuição dos pontos.

Finalmente, cada líder de *cluster* segue o resultado do algoritmo de distribuição e tenta selecionar os pontos para migrá-los para outros *clusters*. Os pontos selecionados são informados por uma mensagem de "migração" para que seja executado o procedimento de migração.

- Atribuição de categorias de documentos populares

TRIANTAFFILOU *et al.* (2003) afirmam que para garantir o alto desempenho do sistema abordado em seu trabalho, existe a necessidade de evitar gargalos e balancear a carga em todos os pontos do sistema. É considerada como carga, o número de requisições que chegam a um ponto pertencente ao sistema. O balanceamento pode ser parcialmente obtido pela associação das categorias de documentos aos *clusters* de pontos, de modo a garantir uma justa distribuição de categorias de documentos populares (documentos mais requisitados) para os *clusters*. Baseado nisso, TRIANTAFFILOU *et al.* (2003) realizam o balanceamento de carga inter-*cluster*.

O objetivo do balanceamento de carga formalizado em [TRIANTAFFILOU *et al.* 2003] é criar *k clusters* de pontos que terão igual popularidade e, consequentemente, igual carga. A

popularidade de um documento é dada pela probabilidade que um usuário deseja recuperá-lo. Porém, nem todos os *cluster* são do mesmo tamanho. Logo, a popularidade deverá ser normalizada com relação ao número de pontos nos *clusters* de modo que a carga média enfrentada por cada ponto seja igual.

Para o banceamento de carga inter-*cluster* um algoritmo chamado MaxFair é executado. O algorítimo procura equilibrar a carga do *cluster* realocando algumas de suas categorias semânticas para outros *clusters*, dependendo do índice de equilíbrio destes. Para que uma nova categoria seja atribuída a um dos *clusters*, todas as possíveis atribuições serão testadas, considerando o índice de equilíbrio de cada *cluster*. Logo, a *cluster* que tiver o melhor resultado do índice de equilibrio é que receberá a nova categoria.

- Colaboaração entre líderes de *clusters*

O balanceamento de carga inter-*cluster* é realizado através de um trabalho colaborativo entre os líderes dos *clusters* [MONDAL *et al.* 2003]. A movimentação de dados de um *cluster* para um outro pode incorrer em alta sobrecarga de comunicação para justificar o movimento.

Periodicamente, os líderes de *clusters* trocam informações apenas com os líderes vizinhos. Se um líder α percebe que sua carga excede a carga média do conjunto β de seus líderes vizinhos em mais de 10% da média de carga, esse líder primeiro verifica os dados mais procurados que devem ser movidos. Em seguida, esse líder envia uma mensagem informando o espaço mínimo necessário para armazenar cada dado mais acessado nele, para cada líder vizinho em β, a fim de aliviar uma parte de sua carga entre eles. Os líderes β verificam em cada pontos de seus *clusters* a disponibilidade de espaço em disco. Se algum desses pontos satisfizer a condição de espaço em disco, os líderes informarão ao líder α sobre a disponibilidade de espaço disponível e a média de carga recebida pelos pontos. α classifica os os líderes do β em uma lista $List_1$, onde seu primeiro elemento é o líder menos sobrecarregado [MONDAL *et al.* 2003]. Portanto, α movimentará seus dados para o líder menos sobrecarregado de acordo com a classificação da $List_1$. Depois que os dados movidos por α chegarem ao líder de destino, este líder criará uma segunda lista $List_2$ (com seus pontos ordenados de forma crescente de acordo com suas cargas) e redistribuirá os dados recebidos entre os elementos desta lista.

Podemos observar, pelo texto contido em [AYYASAMY *et al.* 2010], que seus autores adotaram para o balanceamento inter-*cluster*, o mesmo mecanismo utilizado no trabalho de [MONDAL *et al.* 2003]. Mecanismo esse que já foi descrito anteriormente.

- Monitoramento do número de pontos

Apesar de não mostrar resultados, GAROFALAKIS *et al.* (2009) informam que o balanceamento de carga inter-*cluster* pode ser implementado por um mecanismo de monitoramento do número de pontos em cada *cluster*. Esse mecanismo decidiria pela divisão dos *clusters* ou pela migração dos pontos para outros *clusters*, garantindo o número igual de pontos (ou valor aceitável estabelecido por alguma medida) entre os *clusters*. Parte desse mecanismo pode ser feito pelo processo de migração de pontos para que a tranferência de dados possa ser feita em outros *clusters*.

- Movimento de pontos

Diferente dos demais trabalhos, QIAO *et al.* (2009) adoratam o movimento de pontos ao invés do movimento de dados para o processo de balanceamento de carga. Segundo eles, sem servidores virtuais, o balanceamento através do movimento de objetos só pode ser realizado entre *clusters* consecutivos em um sistema P2P baseado em *cluster*. Dessa maneira, o balanceamento de carga iria convergir lentamente. Com o balanceamento através do movimento de pontos*,* são evitadas: a sobrecarga de gerenciar a consistência dos dados, considerando o grande volume de pontos em diferentes *clusters* a serem movimentados; e as atualizações das tabelas de roteamento da rede.

Assim como os outros trabalhos, é necessário mensurar quando um balanceamento deverá ocorrer. Pensando nisso, QIAO *et al.* (2009) adotaram a capacidade disponível como índice indicador de balanceamento de carga inter-*cluster*. Esse índice indicador é o que os autores chamam de "índice de carga" (*load index*). Esse índice foi proposto em outros trabalhos [ZHU *et al.* 1998, BHASKARAM *et al.* 2003]. A fórmula apresentada na Figura 4-2 denota a equação que calcula a capacidade disponível de um ponto.

$$\text{AvailableCapacity} = \text{MaximumCapacity} * (1 - \text{utlization})$$

Figura 4-2 - Fórmula para calcular a capacidade disponível de um ponto [QIAO *et al.* 2009]

Utilizando capacidade disponível como o índice de carga, cada *cluster* interativamente executa um procedimento de balanceamento difuso que indentifica o estado de seu próprio

cluster com relação a seus vizinhos. QIAO *et al.* (2009) consideram três esquemas: *directory-initiated*, *sender-initiated* e *receiver-initiated*. O *sender-initiated* é o ponto que irá transferir sua carga, e o *receiver-initiated* é o ponto que receberá a carga de um *sender*. Quando um *cluster* executa o procedimento de balanceamento, o *directory-initiated* localiza os que enviam (*senders*) e os que recebem (*receivers*) entre seus *clusters* vizinhos.

A execução do procedimento de balanceamento é disparada através de evento de *timeout* ou por uma mudança do estado em um *cluster* (passou a ser um *sender* ou *receiver*). O *cluster* determina o *status* de sua carga, assim como a de seus vizinhos, através de troca de mensagens entre eles. Nessas mensagens, informações sobre os índices de carga dos *clusters* vizinhos são fornecidas. Um parâmetro chamado *bound* é utilizado para determinar se um *cluster* está sobrecarregado ou não. A carga média de todos os *clusters* vizinhos é calculada. O limite mínino e máximo de carga são calculados pela fórmula = *average* – *load index* * (1 +/- *bound*). Onde o *bound* é dado em porcentagem da carga média (*average*). Logo, um *cluster* é um candidato a *receiver*(*sender*) de uma carga se seu índice de carga for maior(menor) do que o limite inferior obtido pela fórmula anterior. Todo esse cálculo visa auxiliar na tomada de decisão de qual *cluster* deverá assumir o papel de *receiver* ou de *sender*, e enviar um pedido de transferência de carga para o receptor de cada ponto, incluindo parâmetros como o ID do rementente e a quantidade de carga necessária que o ponto necessita para atingir a carga média.

Após o *sender cluster* receber do sistema a instrução para movimentação de ponto, ele enviará um dos seus membros para o *receiver cluster*. Um fato a ser a considerado é que com a movimentação do ponto para o *receiver cluster*, este não poderá mudar seu estado de não sobrecarregado para sobrecarregado.

4.1.3. Comparativo das abordagens

O quadro a seguir sumariza as estratégias adotadas pelas pesquisas descritas nesta seção. A Quadro 4-1 considera a estratégia de balanceamento tanto no nível intra quanto inter-*cluster* utilizada pelos autores. Além disso, métricas adotadas para a tarefa de balanceamento também são citadas. O significado de cada coluna encontra-se descrito a seguir.

- Artigos: referência na literatura sobre soluções para o balanceamento de carga de *cluster*, em sistemas P2P.
- Balanceamento inter-*cluster:* solução adotada para o balanceamento inter-*cluster*.
- Balanceamento intra-*cluster:* solução(ões) adotada(s) para o balanceamento intra-*cluster.*

- Outras características*:* contém as características únicas em cada solução abordada para um sistema P2P.

Quadro 4-1 - Comparativo das pesquisas sobre balanceamento de carga em sistemas P2P

Artigos	Balanceamento Inter-*Cluster*	Balanceamento Intra-*Cluster*	Outras características
AYYASAMY *et al.* (2010)	Migração de dados	Replicação de dados	o *Cluster* líder. o Replicação é feita em tempo de execução. o Envio de estatísticas de volume de carga aos *clusters* líderes.
GAROFALAKIS *et al.* (2009)	Migração de pontos	Replicação de dados	o Considera as seguintes características dos pontos: largura de banda, capacidade de processamento, tempo *online*. o Conceito de *leaf*, superNode e *candidate* super-*node*. o Métricas: *fairness index*, *quality of availability*, *throughput*, *size of replicated data* e *local cache hits*.
MONDAL *et al.* (2003)	Migração de dados	Replicação/Migração de dados	o Realiza o cálculo da carga de um ponto. o Os pontos guardam estatísticas de acesso. o Em tempo de execução, decide qual estratégia de balanceamento de carga será utilizada. o *Cluster* líder. o Algoritmo orientado a qualidade. o Abordagem *tomada de decisão centralizada.* o Verifica se está desbalanceado em determinados intervalos de tempo.

MICHAIL (2002)	Migração de pontos (difusão)	Replicação de dados/ Redirecionamento de requisições	o Métrica: índice de equilíbrio. o Categorização de documentos. o Monitoramento de popularidades. o Fases para balanceamento inter-*cluster*: *load evaluation*, *flow vector calculation*, *load selection* e *load migration*. o Gerenciamento de réplicas.
TRIANTAFILLOU *et al.* (2003)	Replicação de dados	Replicação de dados	o Utiliza *fairness index* para balanceamento de carga inter-*cluster*. o Considera a popularidade dos documentos. o Algoritmo "MaxFair" para balanceamento inter-*cluster*. o Realiza a normalização de popularidade.
QIAO et al. (2009)	Migração de pontos (difusão)	(não abordado)	o Balanceamento feito em determinado intervalo de tempo ou por algum evento ocorrido no *cluster*. o Propõem balanceamento difuso. o *Load index* é baseado na capacidade de armazenamento. o Capacidade de disponibilidade de um ponto calculada considerando sua utilização. o Fases do procedimento para balanceamento: LB *triggering*, *load determination*, *decision* e *load transfer*.

Alguns trabalhos aparecem na comunidade científica propondo balanceamento de carga em sistemas P2P não baseados em *clusters*. Alguns desses trabalhos encontram-se em [PITOURA *et al.* 2006, LI *et al.* 2005, GODFREY *et al.* 2004, STOICA *et al.* 2003, NOVÁK 2006, ABERER *et al.* 2003, JAGADISH *et al.* 2005, ASPNES *et al.* 2004, GANESAN *et al.* 2004, KARGER *et al.* 2004, RISSON *et al.* 2006].

4.2. Balanceamento de Carga em PDMS

Assim como os sistemas P2P, os PDMS têm a preocupação com a carga dissipada em seu sistema e que possa comprometer a sua eficiência. Nesta seção, soluções de balanceamento de carga utilizadas por alguns dos PDMS, com comunidades semânticas, serão apresentadas.

4.2.1. Sunrise

No Sunrise, cada ponto possui um *Semantic Routing Index (SRI)* a fim de rotear, efetivamente, uma consulta. Um SRI é uma estrutura que sumariza, para cada conceito definido nos esquemas dos pontos, a aproximação semântica de cada sub-rede a partir de seus vizinhos imediatos [LODI *et al.* 2008a, LODI *et al.* 2008b]. O uso do SRI tem como conseqüência a redução da carga na rede, ou seja, reduzir o número de pontos acessados e o esforço computacional desejado em cada ponto acessado. Para este fim, as informações armazenadas nos SRI são apropriadamente exploradas para podar os pontos não relevantes.

4.2.2. iXPeer

No iXPeer, um ponto pode replicar seu conteúdo para outros pontos pertencentes ao mesmo *cluster*. Essa estratégia de réplica, discutida anteriormente, tem por objetivo balancear a carga no *cluster* e explorar os pontos com maiores recursos computacionais. Neste caso, podemos observar que o iXPeer realiza um balanceamento de carga intra-*cluster* para diminuir a carga sobre os pontos do *cluster*.

No caso da sobrecarga incidir sobre um super-ponto *sp*, *sp* realoca alguns dos pontos para outros novos *cluster*s a fim de realizar um balanceamento de carga inter-*cluster*. Porém, se o problema persistir, *sp* solicita ao sistema novos super-pontos e delega a eles parte de sua

carga. Mesmo assim, caso *sp* permaneça sobrecarregado, ele poderá desconectar alguns dos seus pontos, temporariamente.

Em uma situação contrária, quando a carga de *sp* torna-se baixa, *sp* poderá importar alguns pontos de outros super-pontos sobrecarregados, ou ele poderá decidir realocar seus pontos para outro super-ponto e, então, deixar o sistema. Mas vale salientar que essa importação ou exportação de pontos só faz sentido se existir similaridade semântica compatível entre o super-ponto e os pontos importados ou exportados.

4.2.3. OntoZilla

No OntoZilla, devido a estrutura SIG (*Special Interest Groups*) que esse sistema possui, caso um ponto receba uma consulta a qual nem sua classe (*cluster*) ou subclasse possam responde-la, a consulta é encaminhada para a sua superclasse. Como conseqüência, o nível mais alto de um *cluster* sofrerá uma sobrecarga. Portanto, para reduzir a carga sobre o ponto, dois mecanismos foram adotados:

- *Cache* dos endereços dos pontos: como pontos trocam informações, eles podem conhecer pontos de outros *clusters*, verificando familiaridades entre eles. Logo, um *cache* dos endereços desses pontos (mais familiares) é mantido, nos pontos, a fim de melhorar o desempenho da rede.
- Nós virtuais: se a carga em um ponto, pertencente a um *cluster* C, excede um *threshold* estabelecido pelo sistema, o ponto poderá enviar mensagens solicitando, aos pontos da subclasse, o compartilhamento de carga. Logo, os pontos da subclasse atuarão como nós virtuais, compartilhando a carga. Assim que o valor do *threshold*, no ponto sobrecarregado, for reduzido, os nós virtuais deixarão o *cluster* C.

Observamos que o balanceamento de carga de um *cluster* é auxiliado por pontos de outros *clusters*, o que caracteriza um balanceamento de carga inter-*cluster*.

4.2.4. ESTEEM

Assim como no Sunrise, a preocupação está, também, no desbalanceamento de carga que pode ocorrer na rede do ESTEEM. Logo, admite-se que a carga na rede do ESTEEM é diminuída devido ao agrupamento semântico dos pontos desse sistema.

No ESTEEM, os pontos utilizam a CDT - *Context Dimension Tree* (discutida no capítulo anterior) para reduzir a carga de informações selecionadas que não são relevantes para uma determinada pesquisa.

4.2.5. Comparativo das abordagens

O Quadro 4-2 - PDMS versus Balanceamento de Cargaresume as estratégias de balanceamento de carga adotadas pelos PDMS abordados neste capítulo. O significado de cada coluna encontra-se descrito a seguir:

- Sistema: nome do PDMS
- Balanceamento de carga intra-*cluster*: estratégia de balanceamento de carga intra-*cluster* adotado.
- Balanceamento de carga inter-*cluster*: estratégia de balanceamento de carga inter-*cluster* adotado.
- Outros aspectos de balanceamento de carga: considerações sobre o balanceamento de carga utilizado pelos PDMS.

Quadro 4-2 - PDMS versus Balanceamento de Carga

Sistema	Balanceamento de Carga intra-*cluster*	Balanceamento de Carga inter-*cluster*	Outros aspectos de balanceamento de carga
Sunrise	Não identificado	Não identificado	Uso de SRI para diminuir a sobrecarga de mensagens na rede.
iXPeer	Replicação de dados.	Migração de pontos	Ênfase no balanceamento nos dois níveis de conexões (intra e inter-*cluster*).
OntoZilla	Não identificado	Através do uso de nós virtuais.	C*ache* de endereços para diminuir a sobrecarga de mensagens na rede.
ESTEEN	Não identificado	Não identificado	Uso da CDT para diminuir a carga no ponto. Agrupamento semântico para diminuir a carga sobre a rede.

4.3. Considerações Finais

O tamanho dos sistemas P2P e sua natureza dinâmica tornam o gerenciamento dos dados um desafio. Consultas enviadas pelos usuários necessitam de um mecanismo de balanceamento de carga que venha a diminuir o tempo de resposta para o usuário. Estratégias de formação de *clusters* tendem a diminuir esse tempo de resposta por agruparem os dados de forma categorizada. Porém, à medida que os *clusters* são construídos, estes não estão livres de sobrecarga. Logo, soluções para balanceamento de carga em sistemas P2P se fazem necessárias tanto no nível intra-*cluster* quanto no nível inter-*cluster*.

Neste capítulo foram apresentadas noções de balanceamento de carga em sistemas P2P, assim como a sua importância nesses tipos de sistemas. Foram apresentadas as diversas soluções propostas na comunidade cientifica, para os sistemas P2P baseados em *cluster*. No final, uma tabela resumindo as estratégias adotadas para o balanceamento de carga intra-*cluster* e inter-*cluster* foi exibida. No entanto, outros trabalhos sobre balanceamento de carga em sistemas P2P, sem o uso de *clusters* foram citados. Por fim, apresentamos alguns PDMS com *clusters* e quais as soluções adotadas por diminuir a sobrecarga sobre eles. Neste trabalho, o balanceamento de carga que trataremos abordará apenas o excesso de carga de trabalho sobre os *clusters*.

Entretanto, no caso de *clusters* formados baseados em ontologias, a solução para o balanceamento de carga não é trivial. Muitos outros aspectos necessitam ser considerados ao se balancear esses *clusters*, aspectos esses até então não discutidos nas pesquisas aqui apresentadas. Um desses aspectos é a semântica que mantém a conectividade dos pontos nos *clusters*. No próximo capítulo, mostraremos através do PDMS SPEED, uma visão do problema de balanceamento de carga sobre os aspectos semânticos dos *clusters*. A escolha desse sistema deve-se ao fato dele melhor conceber uma visão sobre o problema de balanceamento o qual desejamos solucionar.

Capítulo 5

SPEED

O SPEED (***S**emantic **PEE**r-to-Peer **D**ata Management System*) é um PDMS, baseado em ontologias, capaz de realizar a formação de *clusters* com pontos que possuem similaridades semânticas. Essa formação permite facilitar o estabelecimento dos mapeamentos semânticos entre os pontos e, conseqüentemente, melhorar o processamento de consulta sobre um grande volume de fonte de dados. O SPEED, em sua arquitetura, utiliza um misto de várias topologias de rede P2P que são a DHT, super-ponto e não estruturada. O objetivo de realizar esse misto é para auxiliar a organização dos pontos na rede, de acordo com seus esquemas exportados [PIRES 2009]. Nas seções seguintes, exploraremos a arquitetura desse PDMS, a comunidade semântica dos pontos, os *clusters* semânticos, os problemas decorrentes na manutenção desses *clusters* e algumas considerações finais sobre esse sistema.

5.1. Arquitetura do SPEED

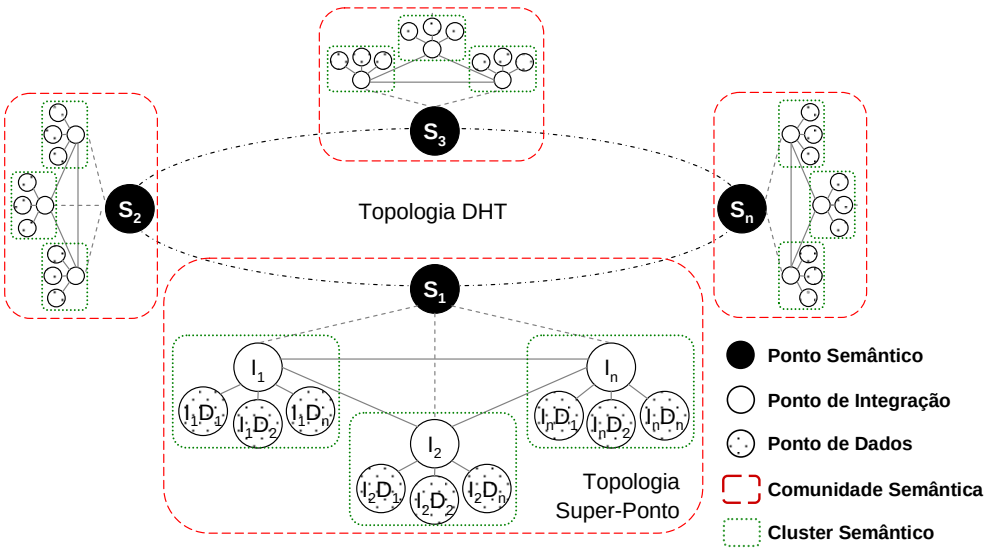


Figura 5-1-Visão geral da arquitetura do SPEED

A Figura 5-1 ilustra a arquitetura do SPEED. Nela podemos observar:

- Ponto de dados (*Data Peer - DP*)

Pontos de dados são pontos que possuem dados a serem compartilhados com outros pontos. Trata-se de um simples computador ou um servidor, que expõe seu conteúdo através da exportação dos seus esquemas, representados através de ontologias. Através desses esquemas é que as fontes de dados, ou parte delas, são acessadas. O ponto de dados corresponde à fonte de dados cujo conteúdo é compartilhado com outros pontos de dados através de mapeamentos semânticos estabelecidos.

- Ponto de integração (*Integration Peer – IP*)

Ponto de integração é o ponto responsável por processamento de consulta, indexação de metadados e integração de dados. Pontos de integração são pontos de dados com alta disponibilidade, largura de banda, capacidade de armazenamento e poder computacional, além disso, eles nomeiam seus respectivos clusters semânticos. O ponto de integração é responsável pela operação de *matching* de ontologias.

- Ponto requisitante (*Requesting Peer*)

Ponto requisitante é um ponto que deseja fazer parte do sistema. Para isto, esse ponto deverá prover a exportação dos seus esquemas, para que estes sejam utilizados para direcionar o ponto à comunidade semântica que ele irá participar.

- Ponto semântico (*Semantic Peer – SP*)

O ponto semântico atua como um ponto de entrada para uma comunidade semântica, além disso, ele é responsável por encaminhar um ponto requisitante a um *cluster* semântico inicial (obtido de um índice semântico). No entanto, o ponto semântico armazena e oferece uma ontologia padrão de um domínio específico (por exemplo, “saúde”, “educação”, “engenharia”, entre outros), e nomeia suas comunidades semânticas correspondentes. No SPEED, existe apenas um único ponto semântico por comunidade semântica.

- *Cluster* semântico

O *cluster* semântico está associado a um interesse comum entre pontos de dados e possui um ponto de integração. O interesse comum é verificado através da análise dos esquemas que foram exportados por cada ponto de dados. Além disso, o ponto de integração

é um dos pontos de dados do *cluster*. Pela definição da arquitetura do SPEED, um ponto de dados está presente em apenas um *cluster* semântico.

- Comunidade semântica

Uma comunidade semântica é construída através da composição de *clusters* semânticos que possuem interesses e semânticas de um domínio de conhecimento em comum como, por exemplo, saúde e educação. Ao entrar no sistema, um ponto de dados é selecionado, por um ponto semântico, para entrar em um *cluster* semântico apropriado, onde o ponto de dados deve ser conectado.

- Índice semântico (*Semantic Index)*

O índice semântico está localizado em um ponto semântico e descreve o conteúdo disponível nos *clusters* de uma comunidade semântica. Cada entrada do índice armazena uma representação resumida dos esquemas compartilhados pelos pontos de um *cluster* particular. É armazenado, também, o endereço de rede do ponto de integração correspondente ao *cluster*. O índice semântico auxilia o ponto requisitante a se conectar ao sistema.

- Vizinho semântico (*Semantic Neighbor)*

Dois *clusters* são vizinhos semânticos se eles pertencem à mesma comunidade semântica, compartilham conteúdos semânticos similares e estão a certa distancia dentro da rede *overlay*.

- Vizinhança semântica (*Semantic Neighborhood)*

Chama-se de vizinhança semântica o conjunto de vizinhos semânticos de um *cluster*.

- Ponto Relevante (*Relevant Peer*)

Um ponto relevante refere-se a um ponto, seja ele de integração ou de dados, que é capaz de responder a uma consulta de forma integral ou parcial.

- Topologia de super-ponto

Com o intuito de fragmentar a rede em porções mais fáceis de gerenciar, o conceito de *cluster* alia-se ao de super-ponto. Com o emprego desta topologia, existe uma maior exploração da heterogeneidade dos pontos participantes.

- Topologia DHT

No SPEED, a topologia DHT é adotada para auxiliar os pontos que possuem interesses comuns a encontrar um ao outro, e construírem comunidades semânticas. Essa topologia é composta pelos pontos semânticos com boa largura de banda e que permanecem na rede por longos períodos de tempo.

5.2. Uso de ontologias no SPEED

Diversos são as formas aos quais as ontologias são aplicadas ao SPEED. Neste sistema, as ontologias são utilizadas para enriquecer seus serviços e proporcionar aos seus usuários, resultados mais completos para suas consultas.

No SPEED, as fontes de dados são heterogêneas, portanto, um modelo de dados comum precisou ser adotado. Logo, todos os pontos que desejam compartilhar dados no SPEED necessitam ter seus esquemas representados através de ontologias (modelo de dados comum utilizado nesse sistema). Além disso, uma ontologia é usada para a representação conceitual de cada *cluster* para prover um melhor entendimento da informação que está sendo compartilhada entre os pontos, dentro do *cluster*.

A Figura 5-2 ilustra diversos tipos de ontologias que são utilizadas nos pontos pertencentes à arquitetura do SPEED. Onde as variáveis *i*, *j* e *k* representam o i-ésimo ponto semântico, j-ésimo ponto de integração e k-ésimo ponto de dados, respectivamente.

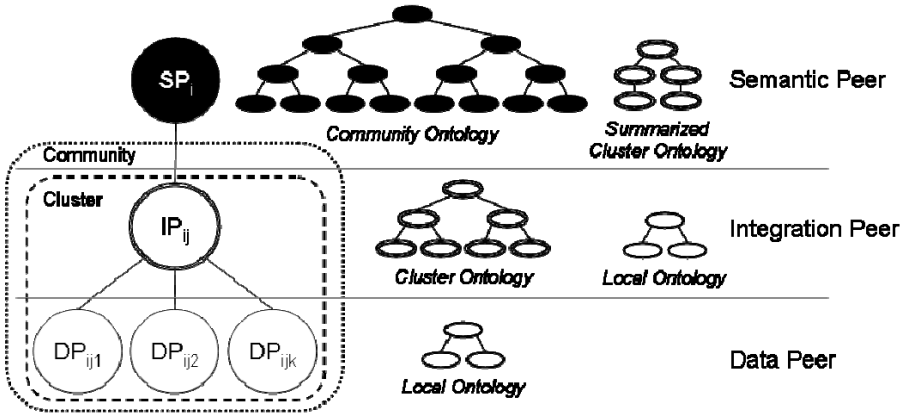


Figura 5-2 - Ontologias utilizadas nos pontos [PIRES 2009]

Como podemos observar pela figura anterior, quatro tipos de ontologias são empregadas na arquitetura do SPEED:

- Ontologia Local

Uma ontologia local é utilizada para descrever os esquemas exportados que descrevem os dados a serem compartilhados por um ponto de dados, de integração ou por um ponto requisitante.

- Ontologia do *cluster*

Uma ontologia do cluster sé utilizada para descrever o conteúdo disponível no *cluster* semântico e fica armazenado no ponto de integração, que o gerencia. Essa ontologia é obtida pela operação de *merging* dos esquemas exportados pelos pontos de dados e de integração que compõem o *cluster*. A Figura 5-3 ilustra uma ontologia de um *cluster* cujo domínio é *educação.*

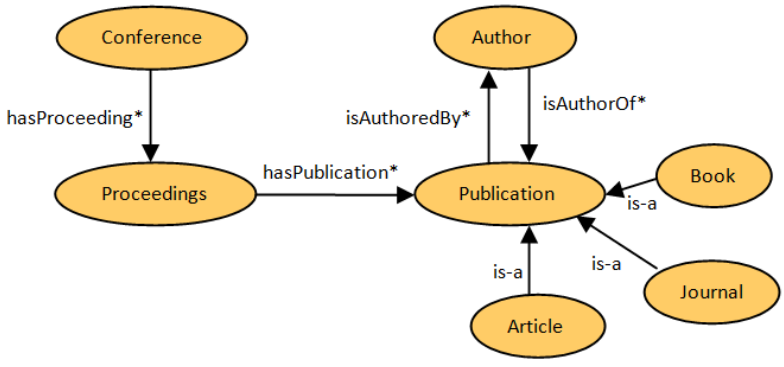


Figura 5-3 - Ontologia do cluster

- Ontologia da comunidade

Armazenada no ponto semântico, a ontologia da comunidade é utilizada para descrever o domínio de conhecimento associado à comunidade semântica.

- Ontologia sumarizada do *cluster*

Uma ontologia sumarizada do *cluster* corresponde a uma versão resumida da ontologia do *cluster*. Trata-se de um subconjunto dos elementos mais relevantes pertencentes à ontologia do cluster. O processo como essa sumarização é realizada encontra-se em [PIRES *et al.* 2011].

Com a finalidade de facilitar um ponto requisitante a se conectar ao sistema, cada ontologia de *cluster* é representada pela sua ontologia sumarizada. Esses sumários são guardados no índice semântico de um ponto semântico. A Figura 5-4 ilustra como ficaria a ontologia do *cluster*, ilustrada pela Figura 5-3, após o processo de sumarização definido por PIRES (2009). Mais detalhes sobre a importância da ontologia sumarizada para o SPEED encontram-se na próxima seção.

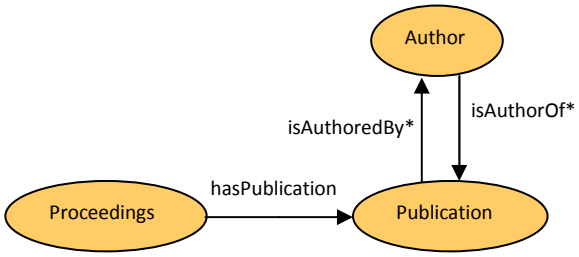


Figura 5-4 - Ontologia sumarizada de um cluster

Cada uma dessas ontologias é manipulada por módulos apropriados existentes em cada tipo de ponto do SPEED. Esses módulos encontram-se detalhados em [PIRES 2009].

5.3. Mapeamento dos Esquemas

Os mapeamentos entre esquemas, no SPEED, são também chamados de mapeamentos semânticos ou mapeamentos de ontologias. De acordo com PIRES (2009), os mapeamentos semânticos descrevem as correspondências que existem entre os elementos de duas ontologias distintas. Como no SPEED existem vários tipos de ontologias, o mapeamento

semântico é subdividido em dois tipos: mapeamento *cluster-to-local* e mapeamento *cluster-to-cluster*.

- Mapeamento *cluster-to-local*

Quando existe um mapeamento semântico entre uma ontologia local de um ponto de dados com a ontologia de um determinado cluster, esse mapeamento é dito mapeamento *cluster-to-local*. Esse mapeamento só é realizado a partir do momento em que um ponto requisitante passa a fazer parte de um *cluster*, como um ponto de dados. Porém, caso o ponto de dados deixe por algum motivo o *cluster*, esses mapeamentos deverão ser removidos da ontologia do *cluster*. Esses mapeamentos encontram-se armazenados no ponto de integração.

- Mapeamento *cluster-to-cluster*

Quando há um mapeamento semântico entre pontos de integração vizinhos, chamamos esse mapeamento *cluster-to-cluster*. Esse mapeamento é fundamental para determinar os vizinhos semânticos dos *clusters*, e só é realizado quando há formação de um novo cluster. Segundo PIRES (2009), mapeamentos *cluster-to-cluster* são definidos entre o novo *cluster* e seus vizinhos correspondentes. Estes mapeamentos sofrem atualizações na medida em que a ontologia do *cluster* é alterada (pela entrada/saída de um ponto de dados ou pela evolução da ontologia local de seus pontos de dados). O mapeamento *cluster-to-cluster* é armazenado em ambos os pontos de integração, semanticamente vizinhos.

5.4. Outras considerações arquiteturais do SPEED

Uma DHT, caracterizada pela eficiência para pesquisas e sensibilidade às mudanças em suas estruturas, é utilizada no SPEED como um recurso para auxiliar um ponto requisitante a encontrar a sua comunidade relacionada. As DHT utilizam um tipo de tabela de referências que indicam os pontos que possuem proximidade aos dados desejados. A rede DHT do SPEED é formada apenas por pontos semânticos, excluindo o dinamismo desses pontos (entrada e saída), a fim de evitar custos de manutenção desnecessários.

Em uma arquitetura baseada em *cluster*, caso um ponto participe de mais de um *cluster*, ineficiências poderão ser introduzidas [VDOVJAK *et al.* 2006]. PIRES (2009) cita como exemplo uma consulta que possa ser respondida por todos os *clusters* que incluem o ponto (redundante), resultando em uma duplicação dos resultados da consulta e aumento do esforço

de comunicação. Portanto, para evitar esse problema, no SPEED um ponto só poderá fazer parte de um *cluster*.

A formação de *clusters* de acordo com os esquemas exportados pelos pontos (ou seja, as ontologias locais) proporciona um ambiente mais adequado para estabelecer os mapeamentos desses esquemas exportados. Além disso, cada ponto de integração gerencia uma descrição dos pontos de dados que foram agregados ao *cluster* (ou seja, o ponto de integração detém o conhecimento semântico do seu *cluster*), facilitando o processamento de uma consulta. Uma consulta recebida por um ponto de integração é processada nos pontos de dados mais relevantes, dentro de seus *clusters* correspondentes. Como os pontos de dados estão agrupados de acordo com sua similaridade semântica, os resultados das consultas são mais precisos.

Quando um ponto requisitante deseja entrar no sistema, o seu esquema exportado é comparado às ontologias dos clusters existentes no índice semântico do ponto semântico. Essa comparação se faz necessária para que o ponto semântico saiba direcionar o ponto requisitante ao *cluster* que possuir uma similaridade semântica mais próxima a esse ponto. Observemos que se todas as ontologias dos *clusters*, de uma determinada comunidade semântica, fossem armazenadas em sua totalidade nos pontos semânticos, o custo para este gerenciar todas essas ontologias em termos de tempo e largura de banda da rede seria alto. Idealmente, a busca deveria iniciar por um *cluster* promissor e continuar até que o *cluster* similar seja encontrado. Para que o processo de localizar o *cluster* para o ponto requisitante torne-se mais rápido, as ontologias dos *clusters* são sumarizadas. Os termos mais relevantes de cada ontologia dos *clusters* são identificados e armazenados no índice semântico do ponto semântico, formando o sumário. Porém, ressalta-se que a ontologia sumarizada de um *cluster* não o representa em sua totalidade.

5.5. Processo de formação dos *clusters* no SPEED

Segundo PIRES (2009), a formação de clusters no SPEED é um processo, principalmente, incremental. Os pontos são adicionados aos *clusters* semânticos a qualquer momento, dependendo da similaridade semântica do ponto com o esquema do *cluster*. O sistema, considerando o esquema exportado do ponto requisitante realizará a busca pela comunidade semântica, da qual esse ponto possa fazer parte. Encontrada a comunidade semântica, o sistema buscará dentro dessa comunidade, o *cluster* que o ponto requisitante

fará parte. A seguir, explicaremos com mais detalhes como é realizada a formação de um *cluster*, a partir da entrada dos pontos requisitantes.

5.5.1. Pesquisando a comunidade semântica

Cada comunidade semântica é descrita por palavra-chave associada a um domínio de conhecimento específico. Exemplos de palavras-chave são “saúde”, “educação” e “geografia”. Essas palavras são definidas pelo administrador do sistema e utilizadas para localizar a comunidade semântica na rede DHT. Uma comunidade semântica é criada por grupos que desejam compartilhar informações sobre um domínio específico.

Em cada ponto semântico, responsável por gerenciar sua comunidade semântica, existe uma tabela chamada de *finger table*. Nesta tabela encontram-se armazenadas informações sobre o identificador e endereço de rede de cada ponto semântico existentes na DHT. Portanto, a inclusão de um novo ponto semântico implica na atualização de todas as *finger tables* existentes nos pontos semânticos.

Para conectar-se ao SPEED, o sistema realizará uma pesquisa na rede DHT a fim de encontrar a comunidade semântica mais adequada ao ponto requisitante. Para isso, esse ponto informa seu tema de interesse através de palavras-chave extraídas do seu esquema importado (ontologia local). Essa extração pode ser feita de forma automática ou manual por seu usuário. Em seguida, um *hashing* é aplicado sobre o tema de interesse do ponto requisitante para gerar um identificador, sendo este enviado de forma aleatória aos pontos semânticos da DHT, no intuito de localizar a comunidade semântica apropriada.

De acordo com a estratégia citada anteriormente, se o tema de interesse está contido no conjunto palavras-chave de uma comunidade semântica, então o ponto semântico correspondente será encontrado. Neste caso, uma mensagem contendo o endereço do ponto semântico é enviada ao ponto requisitante. A Figura 5-5 ilustra, através de um diagrama de seqüência UML, o processo de busca por uma comunidade semântica. Uma vez identificada à comunidade semântica, o sistema agora deverá informar de qual *cluster*, dentro dessa comunidade, o ponto requisitante fará parte.

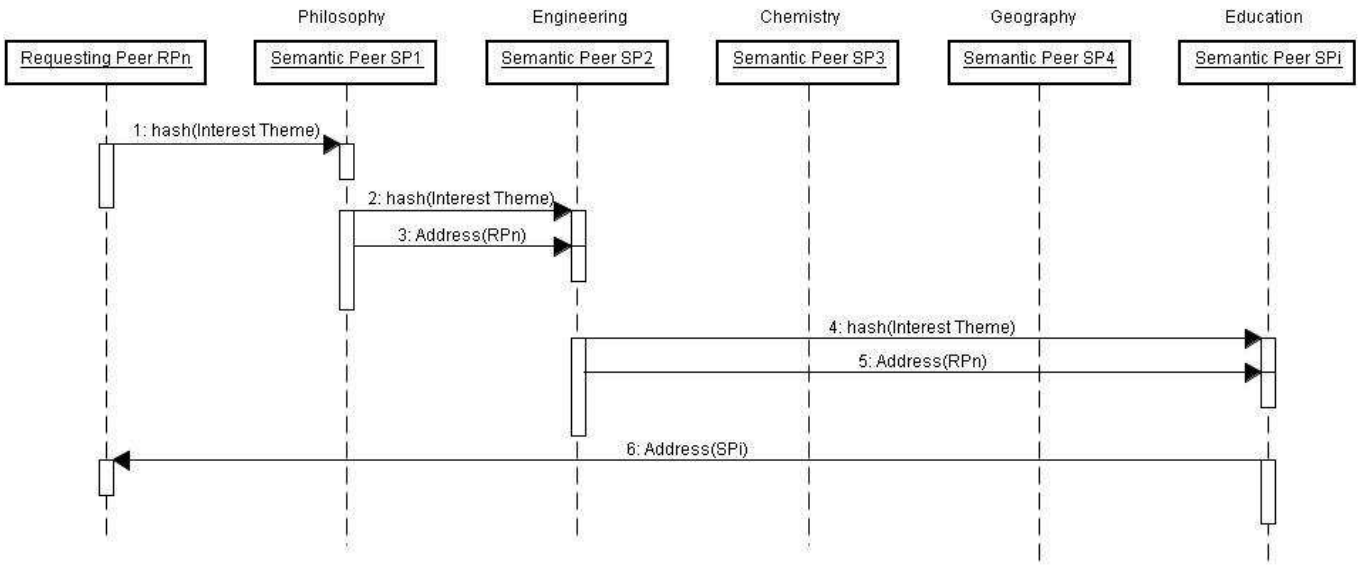


Figura 5-5-Diagrama de seqüência ilustrando com é encontrada uma comunidade semântica na DHT [PIRES 2009]

Caso o tema de interesse não seja encontrado na rede DHT, isso significa que a comunidade não existe na rede ou o tema fornecido não foi utilizado para descrever alguma comunidade semântica. Portanto, o ponto requisitante só fará parte do sistema se ele encontrar a sua comunidade semântica. Podemos pensar que o tema de interesse poderia descrever mais de uma comunidade semântica. Isso até poderia ser possível, mas a especificação do SPEED assume que as palavras-chave não podem especificar mais do que uma comunidade semântica.

A discussão sobre como realizar a busca do *cluster* semântico será abordada na próxima seção.

5.5.2. Pesquisando e ingressando em um *cluster*

Considerando que a comunidade semântica foi encontrada pelo ponto requisitante, a busca por um cluster semântico similar é iniciada. É realizada, no ponto semântico, através dos índices semânticos, a busca por um *cluster* inicial. A busca por um *cluster* semanticamente similar inicia-se pelo *cluster* inicial e continua por seus vizinhos semânticos, que se encontram na rede não estruturada. Um *cluster* é vizinho semântico de outro se a medida de similaridade entre eles está igual ou acima do *threshold* denominado ***neighbor threshold*** [PIRES 2009]. Uma medida de similaridade indica o grau de similaridade que existe entre duas ontologias, ou seja, o quão próximas, semanticamente, duas ontologias que representam os esquemas dos *clusters* estão.

A vizinhança semântica de um *cluster* semântico *CS* é o conjunto de *clusters* que possuem as medidas de similaridades entre suas ontologias maior ou igual ao *neighbor threshold*, sendo todos esses clusters pertencentes à mesma comunidade semântica. Nesse caso, os clusters que fazem parte da vizinhança do *cluster* CS, são ditos **vizinhos diretos**. Porém, se existe um *cluster* CS_i que não está contido nesse conjunto, mas é vizinho direto de algum(ns) dos clusters desse conjunto, CS_i é dito **vizinho indireto** de CS. Logo, a busca pelo *cluster* o qual o ponto requisitante deverá fazer parte começa pelo *cluster* inicial, podendo partir para seus vizinhos diretos e, em último caso, chegar aos seus vizinhos indiretos.

Em cada *cluster* visitado, a similaridade semântica do *cluster* (a ontologia do *cluster*) e a do ponto requisitante (ontologia local) é calculada. Haverá similaridade semântica entre duas ontologias (do *cluster* e do ponto requisitante), caso o valor da medida, resultante da comparação delas, esteja acima de determinado *threshold,* chamado ***cluster threshold***. Detalhes sobre a função de similaridade utilizada no SPEED encontram-se em [PIRES 2009].

Cada ponto de integração de cada *cluster* é visitado, sendo aplicada a medida de similaridade entre ele e o ponto requisitante. Se em algum desses *clusters* a medida de similaridade for inferior ao *cluster threshold*, o ponto requisitante fará parte deste *cluster*. Logo este ponto passará a ser um ponto de dados.

A partir do momento em que o ponto requisitante passou a ser um ponto de dados, um processo de *merge* entre a sua ontologia local e a ontologia do *cluster* é realizado. Portanto, uma nova ontologia será formada assim como um conjunto de correspondências semânticas entre a ontologia do *cluster* e a ontologia local as quais são necessárias ao processamento de consulta. Em seguida, um novo processo de sumarização da nova ontologia formada é executado, e a ontologia sumarizada será atualizada no índice semântico, do seu ponto semântico correspondente.

5.5.3. Formando um novo *cluster*

Se em nenhum dos testes de medida de similaridade entre os *clusters* semânticos e o ponto requisitante resultarem em um valor maior ou igual ao *cluster threshold*, um novo *cluster* será criado dentro da comunidade semântica e o ponto requisitante passará a ser o ponto de integração deste novo *cluster*. Sendo assim, a ontologia do *cluster* será a ontologia local do novo ponto. Uma versão sumarizada deste novo *cluster* será gerada e será adicionada ao índice semântico. Neste caso, a vizinhança semântica desse novo *cluster* será formada por todos os *clusters* visitados, tal que o *neighbor threshold* entre cada um desses e o novo *cluster*

seja igual ou superior ao limite estabelecido pelo sistema. A Figura 5-6 ilustra um diagrama de atividades que resume o processo de clusterização no SPEED.

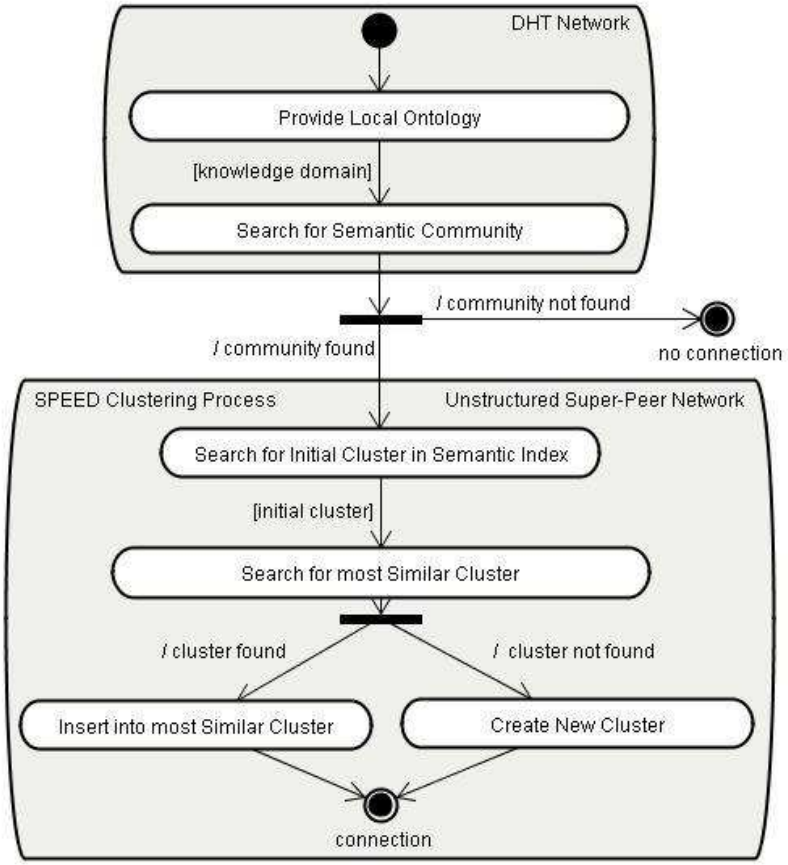


Figura 5-6 - Seqüência de ações para conexão de um ponto requisitante [PIRES 2009]

5.6. Problemas na manutenção dos *clusters*

Algumas considerações necessitam ser feitas sobre a manutenção dos clusters, pois caso as devidas manutenções não sejam realizas, o ponto requisitante pode ser direcionado para algum *cluster* com o qual ele possua pouca similaridade, ou poderá formar um novo cluster, desnecessariamente. Por isso, para que o conteúdo disponível no *cluster* semântico seja refletido corretamente, um mecanismo de manutenção automático e dinâmico necessita ser criado, segundo PIRES (2009), LODI *et al.*(2008), MAUROX *et al.* (2007), CASTANO *et al.* (2005) e KONSTANTINIDIS *et al.* (2008).

Para que a eficiência do sistema seja mantida, o mecanismo de manutenção automática deverá considerar:

5.6.1. Tolerância a Falhas e Seleção dos Pontos de Integração e Candidato

Quando um sistema é projetado de forma que pontos assumam determinados papéis, falhas tornam-se mais susceptíveis a acontecer [PIRES 2007]. Logo, a saída de um ponto, que possui um papel de controle e gerenciamento dentro de um sistema, não poderá comprometer a eficiência do sistema. Segundo VALDURIEZ *et al.* (2004), a qualidade do serviço e a eficiência de um PDMS devem existir mesmo na ocorrência de falhas. Neste caso, estratégias de tolerância a falhas devem estar pré-definidas nos ponto, considerando métricas para escolha do ponto que assuma o papel do ponto que falhou. Algumas pesquisas apontam estratégias para tolerância em sistemas que fazem uso da topologia super-ponto [LO *et al*, 2005, LIU *et al.* 2010, BRITO 2005, JOHNSTONE *et al.* 2005, LI *et al*. 2004, ROUSE *et al*. 2006, LIN *et al*. 2008, SIDIROURGOS *et al.* 2008, BRITO *et al.* 2005].

Como no SPEED o ponto de integração é responsável pelo processamento da consulta, indexação de metadados e integração de dados, uma falha neste ponto poderá, também, desencadear falhas que afetariam a eficiência do sistema. Neste caso, a presença de um ponto candidato a ponto de integração já é cogitada pela arquitetura do SPEED, como um mecanismo de tolerância a falhas. No entanto, o mesmo critério adotado para a escolha de um ponto de integração será utilizado para definir o ponto candidato que possa substituí-lo em caso de falha. PIRES (2007) sugere alguns critérios a serem observados para a seleção desses pontos que são: memória física, espaço em disco, poder de processamento e largura de banda. Além disso, o comportamento dos pontos de dados enquanto estão conectados ao sistema deve ser outro fator relevante. Logo, as seguintes características também devem ser consideradas: disponibilidade, tempo de resposta e quantidade de dados. Porém, os critérios para a definição do ponto de integração e do ponto candidato para o SPEED ainda estão sendo definidos.

Supondo que o ponto candidato já tenha sido definido no *cluster* e caso o ponto de integração venha a falhar, o sistema deverá estabelecer parâmetros que definam o momento exato que o ponto candidato deverá atuar como ponto de integração. Também será necessário que o conhecimento semântico do ponto de integração seja sincronizado com o ponto candidato. Essa sincronização se faz necessária, uma vez que se o ponto candidato assumir o papel do ponto de integração, todo o conhecimento deste último deverá estar presente no ponto candidato para que o sistema continue com o máximo de completude em seus resultados.

Soluções para a tolerância a falhas no SPEED estão fora do escopo deste trabalho, cabendo pesquisas em trabalhos futuros.

5.6.2. Balanceamento de Carga

Os comportamentos dinâmicos dos pontos de dados e de integração podem provocar situações em que um *cluster* semântico, no SPEED, possa possuir mais pontos de dados do que outros *clusters*, em uma mesma comunidade semântica. Como a conectividade dos pontos de dados aos *clusters* é baseada em relações semânticas, calculadas através de medidas de similaridades, o sistema poderá ter uma grande quantidade de pontos de dados fazendo parte de um mesmo *cluster*. Embora isso seja permitido, têm-se algumas implicações com relação à eficiência do sistema. Como o ponto de integração é responsável pelo gerenciamento de consultas, esse ponto poderá ficar sobrecarregado por lidar com uma quantidade excessiva de pontos de dados, principalmente no aspecto do processamento de consultas para cada participante do *cluster* semântico.

Um ponto de integração poderá, em algum momento, ficar sobrecarregado e um mecanismo de balanceamento de carga, no *cluster* do qual esse ponto faz parte, deverá ser realizado. Porém, uma métrica que defina quando um *cluster* estará sobrecarregado se faz necessária. Para que o sistema possa detectar (seja através de um determinado intervalo de tempo e/ou a cada alteração estrutural no estado do *cluster*) que está havendo um desbalanceamento de carga em seus *clusters*, um *threshold* necessita ser estabelecido através de determinados critérios a serem analisados. Esses critérios deverão considerar os níveis de balanceamento de carga tanto *intra* quanto *inter-cluster*.

5.6.2.1. Balanceamento com auxílio do ponto candidato

Uma possível solução para o desbalanceamento de carga seria a divisão de carga entre o ponto candidato e o ponto de integração, a fim de aliviar a carga que incide sobre este último. Conforme a Figura 5-7, caso o ponto de integração PI esteja sobrecarregado, ele enviará ao ponto candidato PC, novas solicitações de consultas a serem processadas. Para isso, o ponto de integração precisaria saber quem é (são) o(s) ponto(s) candidato(s) em cada *cluster*. Teríamos, então, uma solução de balanceamento de carga intra-*cluster*.

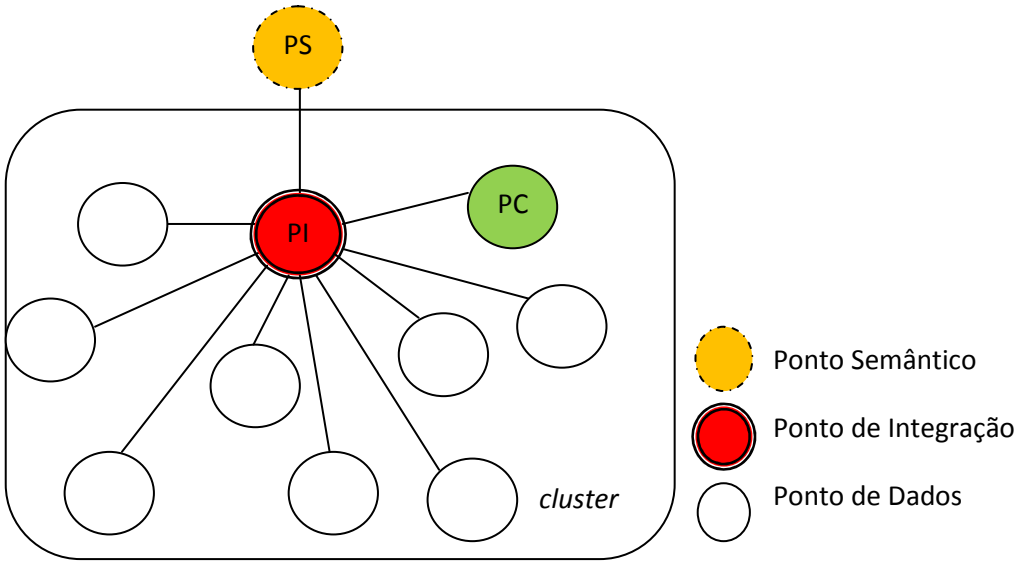


Figura 5-7 - Possível balanceamento de carga intra-*cluster* para o SPEED

Porém, se todos os pontos candidatos resolverem ajudar o ponto de integração (Figura 5-8), isso fará com que a quantidade de mensagens trocadas na rede aumente do *cluster*, e o ponto de integração ficará sobrecarregado para receber as respostas dos pontos candidatos. Além disso, todo o conhecimento semântico do ponto de integração deverá ser replicado para cada possível ponto candidato e ser mantida a sincronização entre eles a fim de manter a consistência do conhecimento semântico do *cluster*.

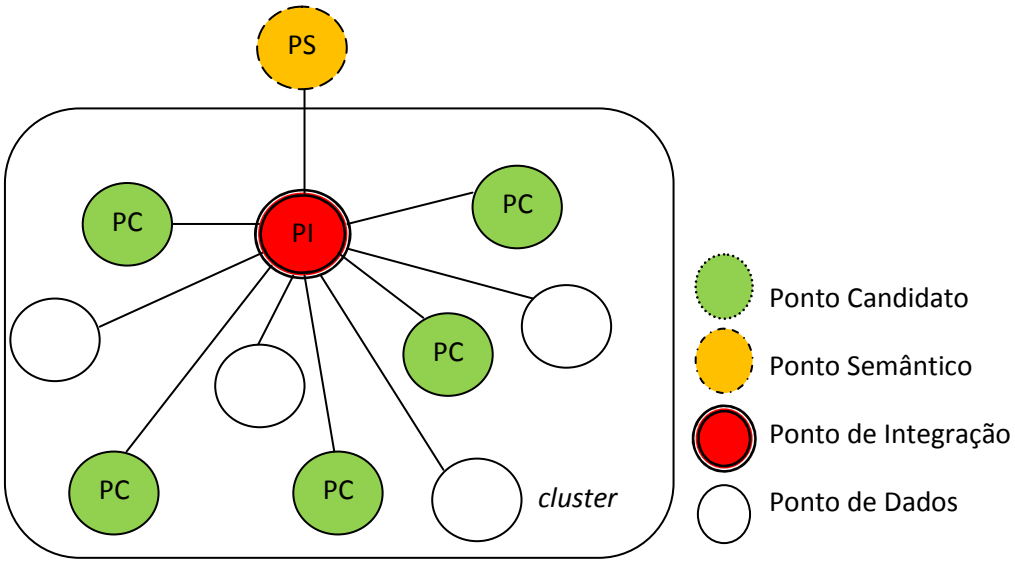


Figura 5-8 - Vários pontos candidatos balanceando a carga do *cluster*

5.6.2.2. Balanceamento através da divisão do cluster

Outra solução para o balanceamento de carga seria através da divisão do *cluster*, retirando os pontos de dados mais solicitados durante uma consulta e formar, com eles, outro *cluster*, caracterizando um balanceamento inter-*cluster*. Para exemplificar, considere a Figura 5-9 que representa um *cluster* que está sendo sobrecarregado devido ao exagerado acesso ao ponto de dados X, por este ser um *hot peer*.

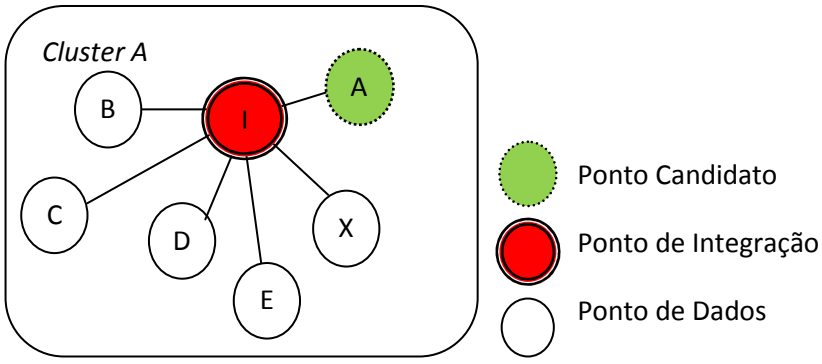


Figura 5-9 - Cluster sobrecarregado devido ao ponto X

Para que esse *cluster* fique balanceado, poderemos remover X do *cluster* A e com esse ponto criar um novo cluster. Mas essa divisão, dando a idéia de um balanceamento inter-*cluster*, através de uma estratégia de migração de pontos não é tão trivial quanto às soluções apresentadas anteriormente. Isso porque a formação do *cluster* é baseada em ontologias e a medida de similaridade mínima entre as ontologias do cluster e as do ponto deverá ser respeitada. Neste caso, fica claro que não há como ter um balanceamento de carga sem considerar os aspectos semânticos dos *clusters*. Portanto, os clusters tendem a estar balanceados, também, semanticamente. Logo, o ponto X para formar um novo *cluster*, os pontos que possuírem maior grau de similaridade com ele deverão estar no novo *cluster B*, conforme ilustra a Figura 5-10 (b). O mesmo equivale para os pontos que permanecerem no *cluster A*, Figura 5-10 (a). O balanceamento de carga, que resultou nos *clusters* A e B, considerou as medidas de similaridades hipotéticas descritas na Tabela 5-1, para o balanceamento semântico. Logo uma nova seleção do ponto de integração e do ponto candidato será realizada no *cluster* B e possivelmente no A.

Alguns possíveis critérios para escolha do ponto de integração I e D, e pontos candidatos A e D serão discutidos neste capítulo.

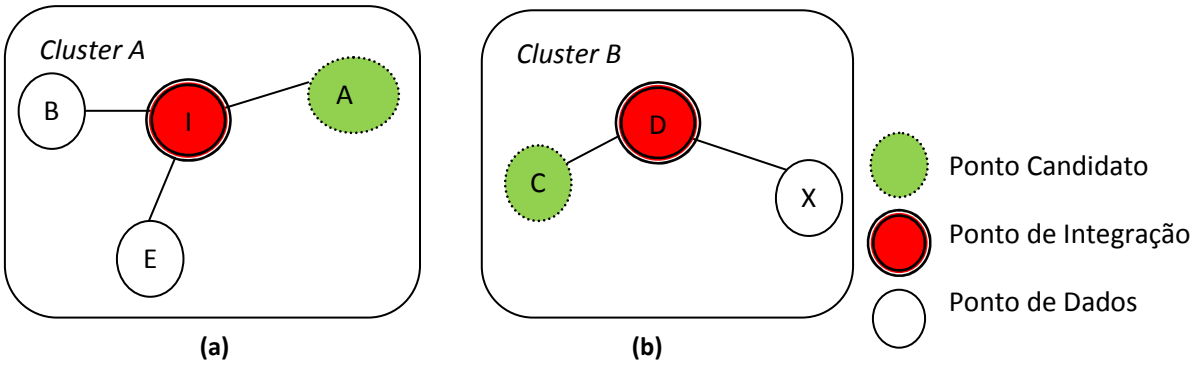


Figura 5-10 - Divisão de um cluster após um balanceamento inter-*cluster*

Os valores da Tabela 5-1 são referentes às medidas de similaridade hipotéticas entre as ontologias locais dos pontos de dados A, B, C, D, E e a ontologia do *cluster* contida no ponto de integração I.

Tabela 5-1 - Medidas de similaridades entre os pontos

	I	A	B	C	D	E	X
I	-	0.7	0.65	0.5	0.5	0.85	0.7
A	0.7	-	0.6	0.4	0.2	0.7	0.3
B	0.65	0.6	-	0.3	0.4	0.6	0.45
C	0.5	0.4	0.3	-	0.8	0.45	0.7
D	0.5	0.2	0.4	0.6	-	0.1	0.8
E	0.85	0.7	0.6	0.45	0.1	-	0.4
X	0.7	0.3	0.45	0.7	0.8	0.4	-

Um processo inverso à divisão do cluster poderia acontecer. Suponha que em um determinado intervalo de tempo, os pontos de integração dos *cluster* A e B fiquem ociosos. Um procedimento de reagrupamento desses *clusters* poderia ser executado, considerando o balanceamento semântico no cluster resultante dessa união.

5.7. Considerações Finais

Neste capítulo apresentamos o PDMS chamado SPEED cuja rede *overlay* foi projetada para auxiliar a conexão de pontos em comunidades semânticas. Foram descritos os três tipos de pontos (ponto de dados, ponto de integração e ponto semântico) que fazem parte da

arquitetura do sistema, assim como o papel que cada um desses pontos desempenha. A arquitetura do SPEED foi ilustrada, exibindo as topologias de redes P2P adotadas.

Vimos que o PDMS abordado neste capítulo possui a formação de comunidades semânticas com o intuito de prover melhor eficiência nos resultados das consultas provenientes de seus usuários. A formação das comunidades semânticas é realizada através do uso de ontologias (ontologias locais dos pontos, ontologias dos *clusters* e ontologias da comunidade semântica). Uma medida de similaridade global é utiliza para determinar a similaridade existente entre os pontos no SPEED. Apesar de o sistema ter sido concebido para auxiliar de forma automática a formação das comunidades semânticas, alguns problemas com a manutenção desses *clusters* precisam ser considerados. Esses problemas necessitam de soluções de auto-organização do *cluster* de forma que a eficiência do sistema não seja comprometida. Os problemas de manutenção considerados foram: os aspectos de tolerância a falhas e seleção dos pontos de integração e candidato para o *cluster*; e o balanceamento de carga dos *clusters*, considerando o conhecimento semântico que os mantém agrupados.

No capítulo seguinte, apresentamos nossa proposta de tese, a qual expõe outras situações de desbalanceamento semântico, baseado nos problemas citados anteriormente. A solução proposta servirá para os PDMS, baseados em ontologias, que possuem uma arquitetura semântica semelhante ao SPEED, sendo este sistema o nosso estudo de caso e ambiente de validação dos resultados.

Capítulo 6

Proposta de Tese

No SPEED, a conectividade dos pontos de dados aos *clusters* semânticos é baseada em relações semânticas, calculadas através de medidas de similaridades entre a ontologia local de cada ponto que chega e a ontologia do *cluster*. Portanto, em determinado instante, uma grande quantidade de pontos de dados poderá fazer parte de um mesmo *cluster*, devido a suas proximidades semânticas. Embora essa grande quantidade seja permitida, têm-se algumas implicações com relação à eficiência do sistema. Como o ponto de integração é responsável pelo gerenciamento de consultas (reformulação, integração dos resultados da consulta, dentre outros), esse ponto poderá ficar sobrecarregado por lidar com uma quantidade excessiva de pontos de dados, principalmente no aspecto do processamento de consultas para cada participante do *cluster*. Neste sentido, um mecanismo de balanceamento de carga deverá ser realizado de forma a não comprometer a eficiência do sistema. Porém, se esse mecanismo alterar as ontologias dos *clusters* (pela migração de pontos, divisão do cluster, dentre outras soluções), as medidas de similaridades entre os pontos de dados poderão tornar-se abaixo do *cluster threshold* especificado pelo sistema. Isso implicaria em uma situação onde pontos de dados estariam presentes em um *cluster* no qual eles não possuem muita similaridade. A esta situação, chamamos de desbalanceamento semântico.

Este trabalho parte da hipótese que uma vez que o SPEED seja capaz de, automaticamente, detectar e resolver situações de desbalanceamento de carga em seus *clusters*, sem que cause o desbalanceamento semântico destes, o sistema se apresentará mais eficiente aos seus usuários.

No Capítulo 4 foram discutidas algumas possíveis soluções para o balanceamento de carga no sistema SPEED, porém algumas delas não são possíveis de serem realizadas em ambientes que possuem *clusters* com conectividade semântica.

Este capítulo tem por objetivo apresentar a proposta deste trabalho de acordo com os problemas e desafios descritos nos capítulos anteriores. As Seções 6.1 e 6.2 são a base deste trabalho e sobre a qual nossa proposta será desenvolvida. Na Seção 6.1 conceituamos e descrevemos a problemática sobre o balanceamento semântico de *clusters*. Na Seção 6.2 mostramos considerações importantes sobre o balanceamento de carga dos *clusters*, destacando a sua relação com o balanceamento semântico. Por fim, discutimos nossas

contribuições e descrevemos a metodologia a ser aplicada, além da apresentação de um cronograma das atividades planejadas para este trabalho.

6.1. Balanceamento Semântico dos *Clusters*

No capítulo anterior vimos que o SPEED realiza a formação de *clusters* com pontos que possuem similaridades semânticas, com o intuito de melhorar o processamento de consulta sobre um grande volume de fonte de dados. Além disso, os *clusters* mantêm *links* para outros *clusters* semanticamente similares a fim de auxiliar os pontos requisitantes a encontrar o seu *cluster* apropriado, além de ajudar a direcionar uma consulta para os *clusters* mais relevantes e que sejam capazes de respondê-la.

Para que o SPEED funcione corretamente, as medidas de similaridades entre as ontologias do *cluster* e as dos pontos de dados deverão respeitar os *thresholds* estabelecidos pelo sistema. O mesmo equivale para um *cluster* e seus vizinhos semânticos. No entanto, em alguns casos veremos que a dinâmica existente nos PDMS pode gerar situações que possam provocar mudanças nas medidas de similaridades entre os pontos e seus *clusters*, fazendo com que essas medidas tornem-se abaixo dos limites determinados pelo sistema. Portanto, se forem identificadas a existência dessas medidas com valores inferiores aos *thresholds*, dizemos então que existe um desbalanceamento semântico no sistema. Logo, uma ação de balanceamento semântico deverá ser desempenhada para reorganizar os *clusters*, a fim de manter as medidas de similaridades que respeitem os *thresholds* estabelecidos.

Podemos classificar o balanceamento semântico em intra e inter-*cluster*. Consideramos que um *cluster* está semanticamente balanceado a nível intra-*cluster* se todas as medidas de similaridades existentes entre a ontologia do *cluster* e as ontologias locais, dos pontos pertencentes a esse *cluster*, estão dentro do limite determinado pelo sistema. No SPEED, este limite é o *cluster threshold* que é, na versão atual do sistema, definido pelo usuário. Na Figura 6-1 (a), o *Cluster* 1 está balanceado semanticamente no nível intra-*cluster*, pois as medidas de similaridades (contidas nas ligações) entre os pontos de dados e o ponto de integração estão acima do *cluster threshold* (supondo este sendo de 0.7).

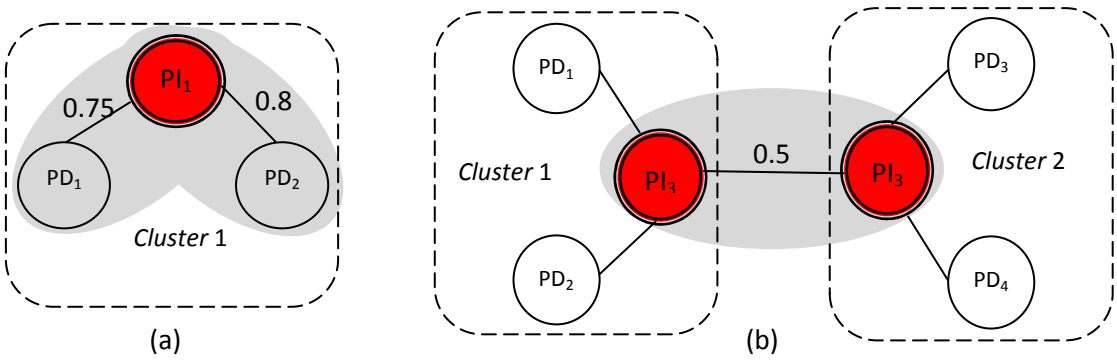


Figura 6-1 - Desbalanceamento intra e inter-*cluster*

Sobre o balanceamento inter-*cluster*, consideramos que um *cluster* está semanticamente balanceado se todas as medidas de similaridades existentes entre a sua ontologia e as ontologias dos *clusters* vizinhos estão dentro do limite determinado pelo sistema. No SPEED, este limite é o *neighbor threshold* e também é, atualmente, definido pelo usuário. Na Figura 6-1 (b) o *Cluster* 1 está balanceado semanticamente no nível inter-*cluster* com o *Cluster* 2. O valor na ligação entre os pontos de integração representa a medida de similaridade entre eles e está acima do *neighbor threshold* (supondo este sendo de 0.5).

A seguir, discutiremos possíveis situações onde um desbalanceamento semântico poderá ocorrer tanto a nível intra quanto inter-*cluster*.

6.1.1. Ações que contribuem para o desbalanceamento semântico intra-*cluster*

A dinamicidade presente nos PDMS permite que em um determinado instante de tempo um ponto de dados possa entrar no sistema, ou as ontologias locais de cada ponto possam ser alteradas a fim de refletir mudanças nos esquemas das fontes, proporcionando inclusão/exclusão de conceitos nessas ontologias. Essas situações podem provocar um desbalanceamento semântico intra-*cluster*.

Para entendermos como a entrada de um ponto poderá gerar um desbalanceamento semântico, consideremos o seguinte exemplo ilustrado pela Figura 6-2(a). Nessa figura podemos observar as ontologias locais dos pontos PD_1, PD_2 e PD_3 que formarão um *cluster* C.

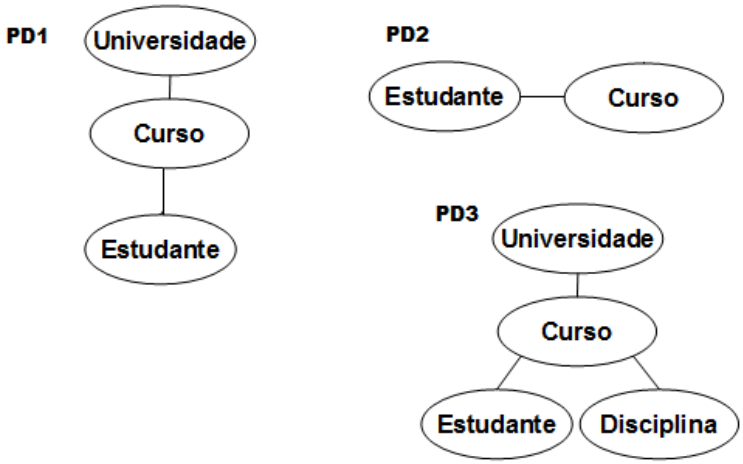


Figura 6-2 – Ontologias de pontos requisitantes

O Quadro 6-1 resume como se dá a entrada de um ponto de dados em um *cluster* e as colunas significam:

- Ponto requisitante: pontos que desejam fazer parte do *cluster*.
- Comparação Ontológica: comparação realizada entre as ontologias do ponto requisitante e a do *cluster*. A comparação é realizada através de uma função de medida de similaridade (MS), definida em [PIRES 2009] e considera o *cluster threshold* de 0.7.
- *Cluster*: mostra os pontos contidos no *cluster* C e a sua ontologia O(C).
- Similaridades: exibe as medidas de similaridades entre as ontologias de todos os pontos de dados e a ontologia do *cluster*.

Quadro 6-1- Desbalanceamento semântico com a entrada de um ponto de dados

Ponto Requisitante	Comparação Ontológica	*Cluster*	Similaridades
PD_1	Não é executada	C = {PD_1} O(C) = {universidade, curso, estudante}	MS(O(PD_1), O(C)) = 1.0
PD_2	MS(O(PD_2), O(C)) = 0.8	C = {PD_1, PD_2} O(C) = {universidade, curso, estudante}	MS(O(PD_1), O(C)) = 1.0 MS(O(PD_2), O(C)) = 0.8
PD_3	MS(O(PD_3), O(C)) = 0.85	C = {PD_1, PD_2, PD_3} O(C) = {universidade, curso, estudante, disciplina}	MS(O(PD_1), O(C)) = 0.85 MS(O(PD_2), O(C)) = 0.66 MS(O(PD_3), O(C)) = 1.0

Como podemos observar pelo o Quadro 6-1, a entrada do ponto de dados PD_3 gerou um desbalanceamento semântico entre o *cluster* e o ponto de dados PD_2. A medida de similaridade entre estes que era 0.8, com a entrada do ponto PD_3 foi decrementada para 0.66.

Mesmo que não haja a entrada de um novo ponto no *cluster*, este poderá ter uma situação de desbalanceamento semântico. Suponha que, com o tempo, a ontologia local de um ponto de dados *PD* sofreu uma alteração devido ao inclusão/exclusão de conceitos. Essa alteração poderá fazer com que a similaridade, antes aceitável entre a ontologia do *cluster* e as demais ontologias locais, torne-se abaixo do *cluster threshold* permitido pelo sistema.

6.1.2. Ações que contribuem para o desbalanceamento semântico inter-*cluster*

No SPEED, os vizinhos semânticos de um *cluster* são identificados considerando a similaridade entre a sua ontologia e as ontologias dos demais *clusters* da comunidade. Portanto, as alterações na ontologia do *cluster*, decorrente das ações que provocaram o desbalanceamento semântico intra-*cluster*, poderão provocar um desbalanceamento semântico inter-*cluster*. Devido a essas alterações as medidas de similaridade entre o *cluster* e seus vizinhos semânticos deverão ser recalculadas. Nesse recálculo, se a medida resultar em um valor abaixo do *neighbor threshold* podemos dizer que existe um desbalanceamento semântico inter-*cluster*.

Considere um *cluster C* e *OC* a sua ontologia. Se a medida de similaridade entre *OC* e a ontologia de cada vizinho semântico de *C* diminuiu para um valor inferior ao *neighbor threshold*, então o vizinho semântico será removido dos vizinhos semânticos de *C*. Porém, se a o valor da similaridade aumenta e torna-se maior ou igual ao *cluster threshold*, então os dois *clusters* vizinhos poderão ser agrupados, formando um único *cluster*, caso este agrupamento não forme um *cluster* com problemas de sobrecarga

6.1.3. Operações para o balanceamento semântico

Para que o balanceamento semântico seja executado, as operações ilustradas pela Figura 6-3 poderão ser executadas. As operações são: divisão de *cluster*, migração de pontos e junção de *cluster*.

A operação de divisão de um *cluster* consiste na saída de alguns de seus pontos de dados para a formação de outro *cluster*. Na Figura 6-3 (a), os pontos PD_1 e PD_5 tornaram-se menos similares com a ontologia do *Cluster* 1, portanto eles deverão deixar o *cluster*. Caso não existam outros *clusters* que possuam similaridades com esses pontos, PD_1 e PD_5 formarão um

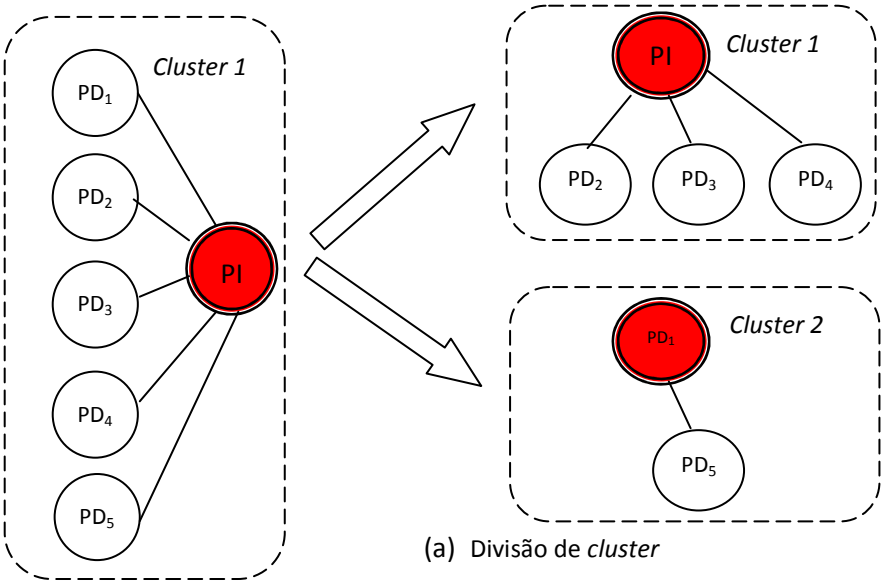
novo cluster (*Cluster* 2), através do processo de formação de *clusters* descrito no capítulo anterior. Observamos pela Figura 6-3 (a) que o PD_1 tornou-se o ponto de integração no *Cluster* 2. Se os pontos que estão deixando o *cluster* forem de integração e/ou candidato, uma nova seleção de pontos para ocupar esses papéis será realizada.

Já a operação de migração de pontos ocorre quando pontos de dados deixaram seus *clusters* de origem para entrarem em outro *cluster* de destino já existente. As razões para a saída desses pontos são as mesmas discutidas no parágrafo anterior. Porém, para que o *cluster* de destino receba os pontos migrados, estes deverão ter uma medida de similaridade com a ontologia do *cluster* de destino, maior ou igual ao *cluster threshold*. Podemos observar na Figura 6-3 (b) a migração do ponto de dados PD_4 do *Cluster* 1 para o *Cluster* 2, *cluster* com o qual PD_4 possui maior similaridade e que é vizinho semântico do *Cluster* 1.

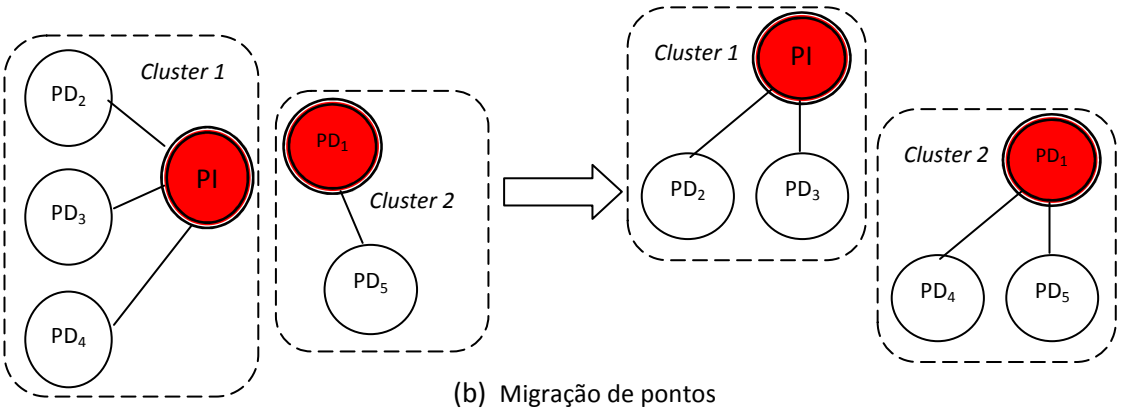
A operação de união de *clusters* consiste no agrupamento de *clusters* que são vizinhos semânticos, e que possuem alta similaridade, ou seja, a medida de similaridade entre as ontologias dos *clusters* é superior ao estabelecido pelo *neighbor threshold*. Neste caso, os *clusters* são reagrupados em um único *cluster* com a identificação de um ponto de integração para o novo cluster. Podemos observar na Figura 6-3 (c) a união dos *clusters* 1 e 2, agrupados no *Cluster* 1.

6.2. Balanceamento de Carga nos *Clusters*

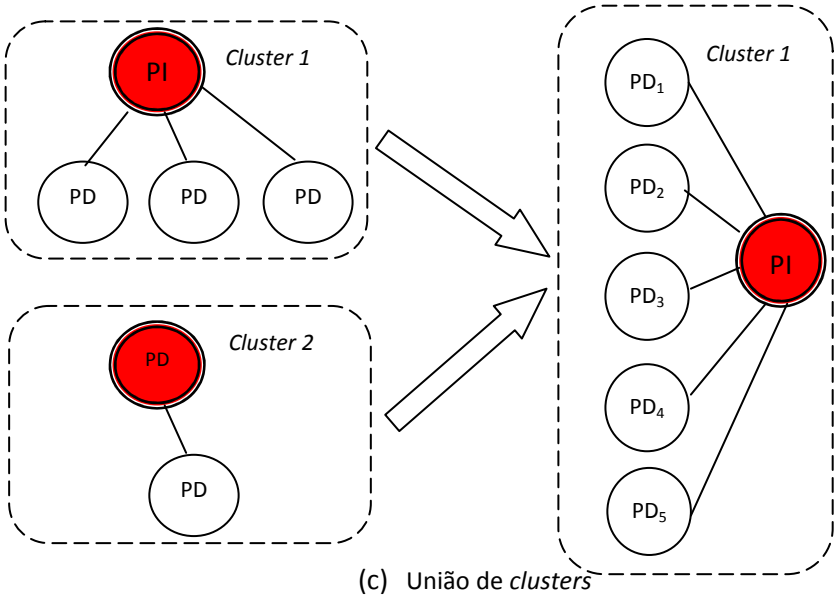
Conforme já foi discutido anteriormente, como a conectividade dos pontos de dados aos *clusters* é baseada em relações semânticas, fará parte do *cluster* qualquer ponto de dados que possua uma medida de similaridade permitida para esse *cluster*. Neste caso, podemos ter *clusters* com muitos pontos de dados. Como o ponto de integração é responsável pelo gerenciamento de consultas, esse ponto poderá ficar sobrecarregado por lidar com uma quantidade excessiva de pontos de dados, principalmente no aspecto do processamento de consultas para cada participante do *cluster* semântico. Em outra situação, o ponto de integração poderá estar sobrecarregado devido à presença de *hot peers* (pontos que possuem sobrecarga) em seu *cluster*. Logo, as conseqüências de uma sobrecarga nos pontos de integração, no SPEED, seriam percebidas através do aumento no tempo de resposta de uma consulta submetida por um usuário, fato este que tornaria o sistema menos eficiente. Neste trabalho, consideramos um *cluster* sobrecarregado como sendo um *cluster* cujo ponto de integração esteja sofrendo sobrecarga de trabalho.



(a) Divisão de *cluster*



(b) Migração de pontos



(c) União de *clusters*

Figura 6-3 - Operações sobre *clusters*

Para que o SPEED possa realizar um balanceamento de carga automático é necessário que algumas questões sejam consideradas. Essas questões estão relacionadas à identificação de um estado de desbalanceamento de carga por parte do sistema, e os fatores que causaram o desbalanceamento. Nas próximas seções serão discutidas essas questões.

6.2.1. Indicador de desbalanceamento de carga

Pensar em um indicador que aponte um estado de sobrecarga é trivial para uma solução de balanceamento de carga no ambiente do SPEED, por vários motivos. Um motivo para se ter um indicador seria auxiliar o próprio sistema a verificar, periodicamente, a existência de *clusters* sobrecarregados. Caso fosse verificada a existência de sobrecarga, o SPEED poderia automaticamente realizar operações de balanceamento de carga nesses *clusters*.

Outro motivo está relacionado ao roteamento de uma consulta. Considere que uma consulta, submetida em um ponto de dados, esteja sendo roteada entre os *clusters* relevantes a essa consulta. Ao chegar em um *cluster C*, a consulta poderá ter uma demora para a sua resposta devido ao fato de *C* estar sobrecarregado. Só depois da consulta ter sido respondida por *C* é que ela poderá continuar sendo roteada para os *clusters* vizinhos dele. Portanto, podemos concluir que se existisse um indicador capaz de informar à consulta roteada, que *C* estaria sobrecarregado, essa consulta poderia prosseguir pelos outros *clusters* não sobrecarregados, até *C* tornar-se menos sobrecarregado. Atualmente, em paralelo a este trabalho, uma pesquisa de doutorado está sendo realizada para tratar os aspectos de roteamento de consulta em ambientes similares ao do SPEED.

O indicador de sobrecarga poderia ser definido por meio de uma métrica que considerasse a capacidade de processamento do ponto de integração, conforme destacaram algumas pesquisas mencionadas no Capítulo 4. Chamaremos esse indicador de *load index* (índice de carga). Além disso, um valor de *threshold*, que chamaremos de *load threshold*, poderá ser definido para indicar o limite de carga de trabalho tolerável pelo sistema. Logo, caso o *load index* de um *cluster* esteja com um valor superior ao *load threshold*, podemos afirmar que existe um desbalanceamento de carga no *cluster*. Então, consideraremos que um *cluster* está balanceado se o seu *load index* está com o valor menor ou igual ao *load threshold*.

6.2.2. Fatores que causam desbalanceamento de carga

Por mais que exista um indicador de sobrecarga em um *cluster*, o sistema não poderá realizar um balanceamento de carga eficiente se a causa da sobrecarga for desconhecida. Um dos fatores que pode causar o desbalanceamento de carga do *cluster* é se sua quantidade de pontos for grande. A Figura 6-4 ilustra uma situação onde o ponto de integração PI possui conexões com *n* pontos de dados. Essa situação, conforme já foi discutido, sobrecarregará o ponto de integração, principalmente no momento do processamento de uma consulta que ele enviará para cada ponto do *cluster*.

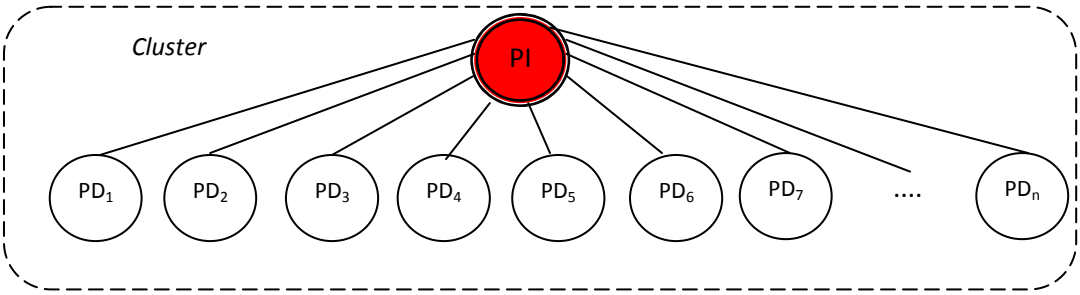


Figura 6-4 - *Cluster* com grande quantidade de pontos de dados

Outro fator que pode causar o desbalanceamento de carga é a presença de *hot peers* no *cluster*. Para exemplificar, considere as ontologias dos pontos de dados PD_1, PD_2, PD_3 e PD_4 representados pela Figura 6-5.

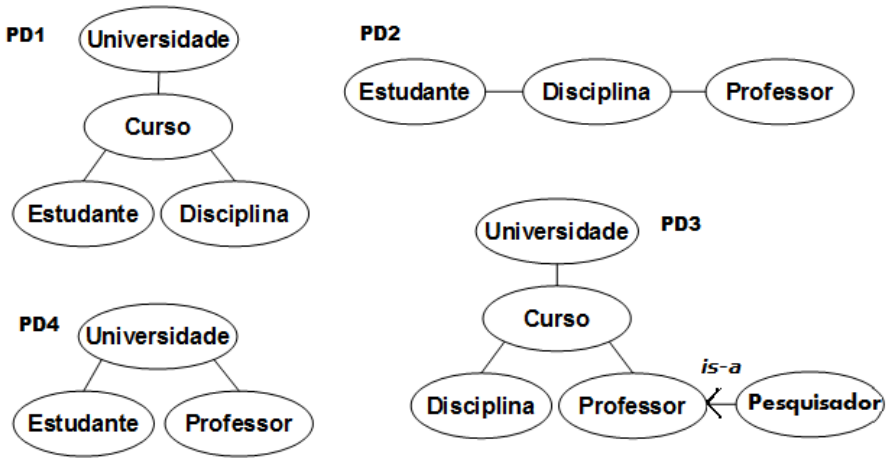


Figura 6-5 - Ontologias dos pontos de dados

As ontologias da Figura 6-5 representam os esquemas dos pontos de dados que fazem parte do *cluster* representado pela Figura 6-6. Suponha que uma grande quantidade de consultas busca informações sobre o conceito “pesquisador”, e que este está presente apenas no ponto de dados PD_3. Nessa situação o ponto PD_3 representa um *hot peer* para o *cluster*. A presença desse *hot peer* poderá prejudicar o tempo de resposta das outras consultas, as quais buscam outros conceitos contidos no *cluster*. Isso porque o ponto de integração terá que processar a grande quantidade de consultas destinadas ao ponto PD_3. Neste caso, uma estratégia para identificação da presença de *hot peers* se faz necessária para que o SPEED possa tomar medidas para o desbalanceamento de carga gerado devido à presença desse tipo de ponto no *cluster*. Uma estratégia a ser considerada é manter o histórico do volume de acesso em cada ponto de dados. Baseado nesse histórico, o sistema poderia inferir os pontos de dados mais consultados.

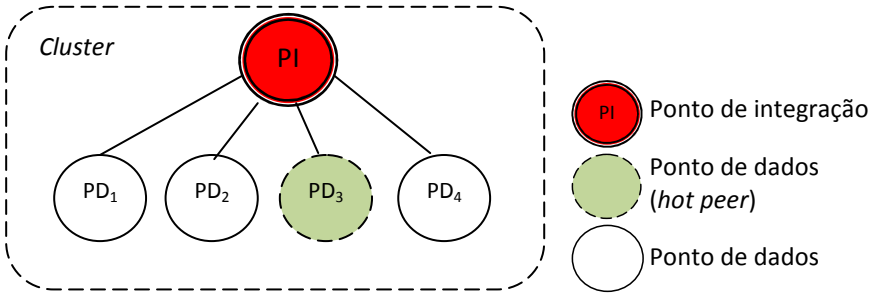


Figura 6-6 - *Cluster* com *hot peer*

A seguir, discutiremos sobre possíveis estratégias de balanceamento de carga para as situações decorrentes dos fatores discutidos nesta seção.

6.2.3. Considerações para o Balanceamento de Carga no SPEED

Um *cluster* ao tornar-se sobrecarregado por conta da grande quantidade de pontos existentes no *cluster*, conforme ilustramos pela Figura 6-5, teria como uma possível solução a participação do ponto candidato auxiliando nas tarefas de execução de consultas. Porém, caso a carga do *cluster* permaneça alta, e como o *cluster* só possui um ponto candidato, a solução seria a migração de alguns pontos de dados para outros *clusters* ou a divisão do *cluster* em novos *clusters*.

A migração dos pontos de dados para outros *clusters* não é tão simples quanto as estratégias apresentadas no Capítulo 4, devido ao agrupamento semântico do SPEED. Um ponto de dados *PD* pertencente a um cluster C_1, só poderia migrar para outro *cluster* destino C_2, caso *PD* tenha uma similaridade semântica aceitável com C_2. Uma implicação da migração do ponto seria uma possível situação de desbalanceamento semântico no *cluster* C_2, devido à entrada desse ponto, conforme discutimos nas Seções 6.1.1. e 6.1.2.

Já a adoção da divisão do *cluster* como estratégia de balanceamento de carga também teria restrições. Não podemos pensar em uma divisão quantitativa, ou seja, dividir o *cluster* pela quantidade de pontos. Por exemplo, suponha que a divisão de *cluster* fosse pela metade, formando dois novos *clusters*. Cada cluster deverá ter os pontos de dados mais similares possíveis. Essa observação deverá ser considerada, caso a divisão do *cluster* seja adotada como a solução de balanceamento de carga na presença de um *hot peer*. Na divisão de um cluster, um novo *cluster* seria formado com a participação do *hot peer* e os pontos mais similares com esse novo *cluster*. Além disso, as mesmas implicações discutidas na operação de divisão da Seção 6.1.3 também existirão neste balanceamento de carga.

Caso as ações de balanceamento de carga e semântico resultem em alterações na ontologia do *cluster*, o processo de geração do novo sumário para o *cluster* deverá ser executado. A ontologia sumarizada resultante desse processo deverá ser inserida no ponto semântico. Além disso, o ponto semântico deverá ser informado caso haja uma mudança de um ponto de integração no *cluster* ou se um novo ponto de integração surgiu devido à formação de um novo *cluster*.

Mediante as situações descritas nesta seção, percebemos que as soluções para o balanceamento de carga também deverão considerar os aspectos semânticos dos *clusters* que mantêm agrupados os pontos de dados.

6.3. Contribuições Esperadas

Propomos neste trabalho uma infra-estrutura para o SPEED que vise resolver os problemas de balanceamento de carga e semântico que possam existir nos *clusters* desse sistema. A idéia é inserir funcionalidades no SPEED para que ele seja capaz de detectar um estado de desbalanceamento e automaticamente organizar seus *clusters* de forma a deixá-los balanceados, tanto nos aspectos relativos à carga quanto à similaridade semântica dos *clusters*. Durante as pesquisas realizadas, nenhum dos sistemas P2P e PDMS, baseados em *clusters*, estudados apresentaram soluções de balanceamento de carga que considerassem os

aspectos semânticos dos seus *clusters*. A seguir, discutiremos as contribuições almejadas e que se referem ao desenvolvimento de soluções que possibilitem a reorganização dinâmica dos pontos de dados, a fim de realizar o balanceamento semântico e de carga, tanto a nível intra como inter-*cluster*.

No tocante ao balanceamento semântico, as contribuições referem-se ao desenvolvimento de soluções que permitam ao sistema detectar e solucionar desbalanceamentos semânticos tanto a nível intra quanto inter-*cluster*, mas que considerem o não desbalanceamento de carga dos *clusters*.

Para realizar o balanceamento semântico de um *cluster*, precisamos definir indicadores que informem quando o *cluster* está semanticamente desbalanceado. O levantamento desses indicadores é a primeira ação a ser desenvolvida e deverá considerar as ações que contribuem para o desbalanceamento semântico, discutidos anteriormente.

Entretanto, nem sempre as ações discutidas na Seção 6.1.1 constatam o estado de desbalanceamento semântico do *cluster*. Por exemplo, um ponto de dados poderá entrar em um *cluster* e não causar o desbalanceamento deste último. Para isso, um indicador deverá ser estabelecido para verificar se houve um desbalanceamento semântico. Um indicador a ser considerado são as medidas de similaridades existentes entre as ontologias locais e a do *cluster*, para a percepção de um desbalanceamento intra-*cluster*. Se a ação de entrada de um ponto foi realizada e houve mudanças nessas medidas de similaridade, para um valor inferior ao *cluster threshold*, então podemos afirmar que houve um desbalanceamento semântico do *cluster*. Identificado esse desbalanceamento, o conjunto de operações, discutido na Seção 6.1.3, deverá ser executado a fim de reorganizar o *cluster* de forma que ele volte ao seu estado de semanticamente balanceado. Porém, essas operações não deverão contribuir para uma sobrecarga do *cluster*. Por exemplo, em uma operação de migração, deverá ser considerado se o *cluster* de destino de um ponto encontra-se com desbalanceamento de carga, para não sobrecarregar ainda mais esse *cluster*. Já o indicador para detecção de um desbalanceamento inter-*cluster* seria a medida de similaridade entre o *cluster* e seus vizinhos semânticos.

Uma vez que os indicadores, os critérios e as ações a serem desempenhadas forem identificados, poderemos ter uma solução onde o próprio sistema (sem a intervenção humana) possa encarregar-se de identificar e realizar o balanceamento semântico dos seus *clusters*. Para isso, um conjunto de regras bem expressivas pode ser definido e inserido no sistema para esse fim. Um modelo promissor, e bastante comum, que pode ser utilizado para expressar essas regras é o ECA (*Event-Condition-Action*). Esse modelo tem sido utilizado com sucesso em uma variedade de campos da computação, tais como a Inteligência Artificial (sistemas multi-agentes), Banco de Dados Ativos, dentre outros [SADRI 2011].

Nas regras ECA, o evento especifica a situação que desencadeia a invocação de uma regra. Trata-se de algo que acontece em algum instante de tempo. Neste trabalho, podemos considerar as ações discutidas anteriormente (a entrada de um ponto ou atualização do esquema dos pontos) como sendo os eventos que podem desencadear um desbalanceamento semântico.

A condição formula em qual estado o objeto observado deverá estar para que uma ação seja executada. A condição sempre é verificada após o evento ter sido detectado. Uma condição a ser verificada, para detectar se houve um desbalanceamento semântico, é observar se as medidas de similaridades, entre as ontologias locais e a ontologia do cluster, foram alteradas para um valor abaixo do limiar permitido pelo sistema. Neste caso, o objeto observado será cada *cluster*.

A ação define o conjunto de atividades a ser realizado caso o evento seja detectado e a avaliação da condição, da regra, resultar em um valor verdadeiro. Logo, as ações corresponderão ao conjunto de operações a serem realizadas para que o *cluster* torne-se semanticamente balanceado. O Quadro 6-2 exemplifica algumas dessas regras para o balanceamento semântico intra-*cluster*.

Quadro 6-2 – Regras ECA para balanceamento semântico intra-*cluster*

<table>
<tr><th>Evento</th><th colspan="2">Condição</th><th>Ação</th></tr>
<tr><td>Entrada de um ponto de dados</td><td rowspan="2">Se existe(m) medida(s) de similaridade(s) entre as ontologias locais e a ontologia do cluster < cluster threshold</td><td rowspan="2">Verdade</td><td rowspan="2">Realizar balanceamento semântico intra-cluster (verificando a existência de clusters com desbalanceamento de carga);
Recalcular os vizinhos semânticos;
Caso necessário, realizar balanceamento inter-cluster.</td></tr>
<tr><td>Atualização da ontologia do ponto de dados</td></tr>
<tr><td colspan="2"></td><td>Falso</td><td>Não realizar o balanceamento (o cluster permanecerá inalterado).</td></tr>
</table>

A regra da Figura 6-7 é disparada no momento em que um ponto de dados entra em um *cluster*.

EVENT: entrada_ponto (ponto, *cluster*)

CONDITION: **IF** existem_similaridades_abaixo (*cluster*, *cluster threshold*) **THEN**

ACTION: //considerar a existência de clusters com desbalanceamento de carga

executar_balanceamento_semantico_intra-cluster(*cluster*)

recalcular_vizinhos_semanticos(*cluster*)

IF existem_desbalanceamento_inter_cluster(cluster) **THEN**

executar_balanceamento_semantico_inter_cluster(*cluster*)

END IF;

Figura 6-7 - Regra ECA para balanceamento semântico após a entrada de um ponto

No tocante ao balanceamento de carga, as contribuições referem-se ao desenvolvimento de soluções que realizem o balanceamento de carga automático do *cluster*, tanto nos níveis intra quanto inter-*cluster*. Porém, como já havíamos discutido anteriormente, as soluções deverão resultar em *clusters* com carga balanceada, mas que garanta o balanceamento semântico dos mesmos. Por mais que um *cluster* esteja semanticamente balanceado, as operações de balanceamento de carga poderão desencadear o seu desbalanceamento semântico.

Proporemos para o mecanismo de balanceamento de carga uma métrica que considere os aspectos da capacidade de processamento dos pontos de integração, a fim de mensurar quando eles estão sobrecarregados, ou seja, uma métrica que defina o *load index* de um *cluster*. Além disso, devemos definir um *load threshold* ideal para o sistema.

Para solucionar as conseqüências dos fatores discutidos na Seção 6.2.2, proporemos algoritmos que visem realizar o balanceamento de carga intra e inter-*cluster*. Ainda na Seção 6.2.2., discutimos algumas possíveis soluções para os fatores que provocam esse desbalanceamento. Nossa proposta é que o sistema possa, na presença de um desbalanceamento de carga, tentar resolver este problema através do balanceamento intra-*cluster* (com a participação do ponto candidato auxiliando o ponto de integração). Acreditamos que se o balanceamento for atingido, não teremos o risco de termos os *clusters* desbalanceados por não alterarmos a estrutura do *cluster*.

Entretanto, caso o balanceamento de carga ainda não seja obtido, o sistema recorrerá às soluções inter-*clusters*. Para isso, o SPEED poderá realizar operações de divisão do *cluster* ou a migração de seus pontos para outros *clusters*. Essas operações provocarão mudanças nas

estruturas dos *clusters* e, conseqüentemente, poderão desencadear desbalanceamentos semânticos no *cluster* e em seus vizinhos. Portanto, proporemos algoritmos de balanceamento inter-*cluster* que considerem as operações descritas na Seção 6.1.3.

Assim como para o balanceamento semântico, regras ECA também poderão ser utilizadas para permitir que o SPEED possa detectar a ocorrência de desbalanceamento de carga em seus *clusters* e automaticamente balanceá-los. Um evento poderia ser um instante de tempo (*t*) atingido. Logo, a cada instante *t* o sistema verificaria se seus *clusters* estão sobrecarregados. Já a condição para a regra ECA seria a verificação se o *load index* está com um valor superior ao *load threshold*, o que indica a presença de um *cluster* sobrecarregado. As ações serão as atividades a serem executadas pelo próprio sistema para que o balanceamento de carga seja obtido, sem que nenhum *cluster* torne-se semanticamente desbalanceado.

O Quadro 6-3 exemplifica uma regra ECA para o balanceamento de carga no SPEED que é disparada quando um determinado tempo *t*, definido pelo sistema, é atingido. Então, o sistema verificará, em cada *cluster*, se existem sobrecargas neles. Em caso verdadeiro, o próprio sistema executará ações de balanceamento de carga, considerando os aspectos semânticos que mantêm a conectividade semântica do *cluster*.

Quadro 6-3 - Regra ECA para balanceamento de carga do *cluster*

Evento	Condição		Ação
Intervalo de tempo (*t*)	$Cluster_{load\_index} > load\ threshold$	Verdade	Realizar balanceamento de carga, considerando o não desbalanceamento semântico dos *clusters*.
		Falso	Não realizar o balanceamento (o *cluster* permanecerá inalterado).

A regra da Figura 6-8 é disparada no momento em que cada intervalo de tempo *(t)* é atingido.

EVENT: tempo *t*

C: Cluster

CONDITION:

IF C_{load_index} > *load threshold* **THEN**

ACTION:

executar_balanceamento_carga(*C*);

IF existem_similaridades_abaixo (*C*, *cluster threshold*) **THEN**

executar_balanceamento_semantico_intra-cluster(*C*)

recalcular_vizinhos_semanticos(*C*)

IF existem_desbalanceamento_inter_cluster(C) **THEN**

executar_balanceamento_semantico_inter_cluster(*C*)

END IF;

END IF;

END IF;

Figura 6-8 - Regra ECA para balanceamento de carga com o tempo *(t)* atingido

Portanto, os algoritmos propostos neste trabalho realizarão o balanceamento semântico considerando o balanceamento de carga e vice-versa.

6.4. Metodologia

Respaldaremos a nossa proposta por meio de pesquisas e análises comparativas de soluções já existentes para o problema de balanceamento de carga. Especificações, implementações e testes das soluções serão apresentadas a fim de validar este trabalho de pesquisa. Este trabalho será realizado de acordo com as etapas descritas a seguir.

1ª Etapa – Identificar o momento oportuno para a realização de um balanceamento

- Nesta etapa especificaremos, em detalhes, em quais momentos uma verificação de desbalanceamento tanto de carga quanto semântico poderá acontecer nos *clusters*.

2ª Etapa – Definir indicadores de desbalanceamento

- Definir métricas que auxiliem o SPEED a identificar se seus *clusters* estão com desbalanceamento de carga e/ou semântico.
- Definir a métrica para o cálculo do *load index*.
- Definir um *load threshold* ideal para o sistema.

3ª Etapa – Propor estratégias para balanceamento semântico e de carga

- Identificar as operações sobre ontologias a serem realizadas com as ações das operações de divisão e junção de *clusters* e migração de pontos.
- Considerar situações onde operações de balanceamento semântico possam implicar em *clusters* com desbalanceamento de carga.
- Destacar as conseqüências das operações de balanceamento de carga sobre o balanceamento semântico dos *clusters*.
- Propor um algoritmo para balanceamento de carga e semântico nos níveis intra-*cluster* e inter-*cluster*.

4ª Etapa – Avaliação das soluções para o balanceamento de carga e semântico

- Implementar as estratégias para os balanceamentos.
- Definir cenários para a avaliação das soluções de balanceamentos.
- Definir métricas e critérios para avaliar as estratégias adotadas para os balanceamentos.
- Testar e realizar experimentos para avaliar as estratégias para os balanceamentos.

5ª Etapa – Escrita de artigos para referendar o andamento da pesquisa e os seus resultados

- Escrever artigos científicos contendo os resultados obtidos com as estratégias de balanceamento validadas.

6ª Etapa – Escrita da tese

7ª Etapa – Preparação e defesa da tese.

6.5. Cronograma

Apresentamos nesta seção o cronograma para a realização do trabalho proposto. Neste cronograma encontram-se os prazos dados para cada uma das atividades já realizadas e

para aquelas que ainda serão desenvolvidas. As atividades serão distribuídas considerando o período de Julho de 2009 à Junho de 2013.

6.5.1. Atividades realizadas

I. Créditos em disciplinas
II. Levantamento do estado da arte
III. Estudo sobre estratégias de balanceamento de carga em sistemas P2P com *clusters*
IV. Estudo sobre estratégias de balanceamento de carga em PDMS com *clusters*
V. Preparação para a Qualificação e Proposta de Tese
VI. Defesa da Qualificação e Proposta de Tese

6.5.2. Atividades a serem realizadas

VII. Definição das estratégias de balanceamento de carga e semântico para o SPEED
VIII. Desenvolvimento dos algoritmos de balanceamento de carga e semântico
IX. Preparação do ambiente para realização dos experimentos
X. Implementação das estratégias
XI. Testes e avaliação das estratégias
XII. Escrita de artigos científicos
XIII. Escrita da tese
XIV. Defesa da tese

o

Quadro 6-4, a seguir, ilustra a distribuição das atividades ao longo deste doutoramento.

Quadro 6-4 - Cronograma de atividades

Atividades	2009											
	Jan	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
I												
	2010											
I												
II												
	2011											
III												
IV												
V												
VI												
VII												
IX												
XII												
	2012											
VIII												
IX												
X												
XI												
XII												
XIII												
	2013											
XII												
XIII												
XIV												

Referências Bibliográficas

[ABERER *et al.* 2003] ABERER, K., MAUROUX, P. C., DATTA, A., DESPOTOVIC, Z., HAUSWIRTH, M., PUNCEVA, M., *et al*. (2003). P-Grid: A Self-organizing Structured P2P System. Proceedings of the *SIGMOD Record, vol. 32* , pp. 29-33. San Diego, USA.

[ABERER *et al.* 2004] ABERER, K., MAUROUX, P., HAUSWIRTH, M., PELT, T. (2004). GridVine: Building Internet-Scale Semantic Overlay Networks. Proceedings of the 3rd International Semantic Web Conference (ISWC2004) , pp. 107–121. Hiroshima, Japão.

[ACM 2011] ACM CCS (2011). ACM CSS (Computing Classification System) . Acesso em 27 de Agosto de 2011, disponível em http://www.acm.org/about/class.

[AHMED *et al.* 2011] AHMED, R., BOUTABA, R. (2011). A Survey of Distributed Search Techniques in Large Scale Distributed Systems. IEEE COMMUNICATIONS SURVEYS & TUTORIALS, vol. 13, n° 2, pp. 150-167.

[ANDROUTSELLIS-THEOTOKIS *et al.* 2004] ANDROUTSELLIS-THEOTOKIS, S., SPINELLIS, D. (2004). A survey of peer-to-peer content distribution techonologies. *ACM Computing Survey* , vol. 36, pp. 335–371.

[ARENAS *et al.* 2003] ARENAS, M., KANTERE, V., KIRINGA, I., MILLER, R., MYLOPOULOS, J., KEMENTSIETSIDIS, A. (2003). The hyperion project: from data integration to data coordination. ACM SIGMOD Record, vol. 32, pp. 53–58.

[ASPNES *et al.* 2004] ASPNES, J., KIRSCH, J., KRISHNAMURTHY, A. (2004). Load balancing and locality in range-queriable data structures. *Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*, pp. 115-124. Newfoundland, Canada.

[AYYASAMY *et al*. 2010] AYYASAMY, S., SIVANANDAM, S. (2010). A Cluster Based Replication Architecture for Load Balancing in Peer-to-Peer Content Distribution. *International Journal of Computer Networks & Communications (IJCNC), vol.2*, pp. 158-172.

[BELLAHSÈNE *et al.* 2004a] BELLAHSÈNE, Z., ROANTREE, M. (2004a). Querying Distributed Data in a Super-Peer Based Architecture. Proceedings of the 15th International Conference on Database and Expert Systems Applications (DEXA'04), vol. 3180, pp. 296-305. Zaragova, Espanha.

[BELLAHSÈNE *et al.* 2004b] BELLAHSÈNE, Z., KING, N., & ROANTREE, M. (2004b). Services for Large Scale P2P Networks. European Research Consortium for Informatics and Mathematics News Journal (ERCIM'04) Disponível em Disponível: http://www.computing.dcu.ie/~nking/ercim04.pdf. Acessado em 10 de agosto de 2010.

[BELLAHSÈNE *et al.* 2006a] BELLAHSÈNE, Z., LAZANITIS, C., McBRIEN, P., RIZOUPOLOS, N. (2006a). iXPeer: Implementing layers of abstraction in P2P Schema Mapping using AutoMed. Proceedings of the 2nd Workshop on Innovations in Web Infrastructure (IWI 2006). Edinburgh, United Kingdom. Disponível em http://pubs.doc.ic.ac.uk/iXPeer-layers-of-abstraction-p2p/iXPeer-layers-of-abstraction-p2p.pdf. Acessado em 13 de agosto de 2010.

[BELLAHSÈNE *et al.* 2006b] BELLAHSÈNE, Z., LAZANITIS, C., McBRIEN, P., RIZOUPOLOS, N. (2006b). Querying Distributed Data in a Super-Peer Based Architecture. Proceedings of the 2nd Workshop on Innovations in Web Infrastructure (IWI 2006). Edinburgh, United Kingdom. Disponível em http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.89.511&rep=rep1&type=pdf. Acessado em 31 de agosto de 2010.

[BENEVENTANO *et al.* 2003] BENEVENTANO, D., BERGAMASCHI, S., FERGNANI, A., GUERRA, F., VINCINI, M., MONTANARI, D. (2003). A Peer-to-Peer Agent-Based Semantic Search Engine. Proceedings of the 11th Italian Symposium on Advanced Database Systems, pp. 367-378. Cosenza, Italy.

[BERGAMASCHI *et al*. 2002] BERGAMASCHI, S., GUERRA, F. (2002). Peer-to-peer paradigm for a semantic search engine. Proceedings of the 1st international conference on Agents and peer-to-peer computing, pp. 81-86. Bologna, Italy.

[BHASKARAM *et al*. 2003] BHASKARAM, R., KATZ, R. (2003). Load balancing and stability issues in algorithms for service compositon. *Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1477-1487. San Francisco, USA.

[BIANCHINI *et al.* 2009] BIANCHINI, D., DE ANTONELLIS, V., MELCHIORI, M. (2009). P2P-SDSD: On-the-fly service-based collaboration in distributed systems. Journal of Metadata, Semantics and Ontologies, vol. 5, n° 3, pp. 222–237.

[BONIFATI *et al*. 2005] BONIFATI, A., CHANG, E., LAKSHMANAN, A., HO, T., POTTINGER, R. (2005). HePToX: Marrying XML and heterogeneity in your P2P databases. Proceedings of the of the 31st international conference on Very large data bases, pp. 1267-1270. Trondheim, Norway.

[BOYD *et al*. 2004] BOYD, M., KITTIVORAVITKUL, S., LAZANITIS, C., McBRIEN, P., RIZOPOULOS, N. (2004). AutoMed: A BAV Data Integration System for Heterogeneous Data Sources. Proceedings of the 6th Advanced Information Systems Engineering Conference (CAiSE), pp. 82-97. Riga, Latvia.

[BREITMAN 2005] BREITMAN, K. (2005). *Web Semântica: A Internet do Futuro.* Editora LTC, 2005.

[BRITO 2005] BRITO, G. A. (2005). Integração de objetos de aprendizagem no sistema ROSA - P2P. Dissertação de Mestrado. Instituto Militar de Engenharia. Rio de Janeiro, Brasil.

[BRITO *et al.* BRITO, G., MOURA, A. M. (2005). ROSA-P2P: a Peer-to-Peer System for Learning

2005] Objects Integration on the Web. Proceedings of the 11th Brazilian Symposium on Multimedia and the Web (WebMedia'05), pp. 1-9. Poços de Caldas, Brasil.

[CARO *et al.* 1998] CARO, G. D., DORIGO, M. (1998). Antnet: Distributed stigmergetic control for communications network. Journal of Artificial Intelligence Research, vol. 9 ,pp. 317–365.

[CIGLARIC *et al.* 2006] CIGLARIC, M., VIDMAR, T. (2006). Ant-inspired query routing performance in dynamic peer-to-peer networks. Parallel and Distributed Processing Symposium, 20th International IEEE Computer Society , pp. 287-292.

[COSTA 2009] COSTA, L. R. (2009). Roteamento de consultas em bancos de dados peer-to-peer utilizando colônias de formigas e ontologias. Dissertação de Mestrado.Universidade Estadual Paulista. São José do Rio Preto, Brasil.

[CRUZ *et al*. 2004] CRUZ, I., XIAO, H., HSU, F. (2004). Peer-to-peer Semantic Integration of XML and RDF Data Sources. Third International Workshop on Agents and Peer-to-Peer Computing (AP2PC), vol. 3601, pp. 108-119. Heidelberg, Germany.

[DORIGO 1997] DORIGO, M., GAMBARDELLA, L. M. (1997). Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, vol. 1 , pp. 53–66.

[DRUSCHEL 2001] DRUSCHEL, P. A. (2001). Past: Persistent and anonymous storage in a peer-to-peer networking environment. Proceedings of the 8th IEEE Workshop on Hot Topics in Operating Systems (HotOS-VIII) , pp. 65–70.

[E-mule 2010] E-mule. (2010). Acesso em 28 de julho de 2010, disponível em: http://www.emule-project.net/home/perl/general.cgi?l=30

[FAYE *et al*. 2007] FAYE, D., NACHOUKI, G., VALDURIEZ, P. (2007). Semantic query routing in senpeer, a P2P data management system. Proceedings of the 1st international conference on Network-based information systems. Regensburg, Germany.

[FILLOTTRANI 2005] FILLOTTRANI, P. R. (2005). The multi-agent system architeccture in SEWASIE. Journal of Computer Science &Tecnology, vol.5, n° 4 , pp. 225-231.

[FIORANO 2003] FIORANO. (2003). Super-Peer Architectures for Distributed Computing. White Paper, Fiorano Software, Inc. Disponível em http://www.fiorano.com/whitepapers/superpeer.pdf.

[FRANCONI 2004] FRANCONI, E., KUPER, G., LOPATENKO, A., ZAIHRAYEU, I.(2004). Queries and Updates in the CoDB Peer to Peer Database System. Proceedings of the VLDB'04, pp. 1277-1280. Toronto, Canada.

[GANESAN *et al.* 2004] GANESAN, P., BAWA, M., GARCIA-MOLINA, H. (2004). Online balancing of range-partitioned data with applications to peer-to-peer systems. Proceedings of the Thirtieth international conference on Very large data bases, vol. 30, pp. 444-455.

Toronto, Canada.

[GAROFALAKIS *et al*. 2009] GAROFALAKIS, J., MICHAIL, T.-A. (2009). Load Balancing in a Cluster-Based P2P System. Proceedings of Fourth Balkan Conference in Informatics, pp. 133-138.Thessaloniki, Greece.

[GHODSI 2006] GHODSI, A. (2006). Distributed k-ary System: Algorithms for Distributed Hash Tables. PhD Thesis. KTH-Royal Institute of Technology. Stockholm, Sweden.

[GIANOLLI *et al.* 2005] GIANOLLI, P. R., GARZETTI, M., JIANG, L., KEMENTSIETSIDIS, A., KIRINGA, I., MASUD, M. (2005). Data Sharing in the Hyperion Peer Database System. Proceedings of the 31st international conference on Very large data bases, pp. 1291-1294. Trondheim, Norway.

[GNUTELLA2 2010] GNUTELLA2. (2010). *Gnutella2 Developer Network*. Acesso em Julho de 2010, disponível em http://g2.trillinux.org

[GODFREY *et al*. 2004] GODFREY, B., KARP, R., LAKSHMINARAYANAN, K., SURANA, S., STOICA, I. (2004). Load balancing in dynamic structured P2P systems. Proceedings of the INFOCOM. Hong Kong, China.

[GREEN *et al.* 2007] GREEN, T., KARVOUNARAKIS, G., IVES, Z. Ives, TANNEN, V. (2007). ORCHESTRA: Facilitating Collaborative Data Sharing. Proceedings of the VLDB '07. Vienna, Austrian.

[GROOVE 2010] GROOVE. (2010). Acesso em 15 de Julho de 2010, disponível em http://technet.microsoft.com/en-us/ee263742.aspx#tab=1

[GRUBER 1993] GRUBER, T. R. (1993). A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition. Journal Knowledge Acquisition, pp. 199-220. London, UK.

[GUARINO 1998] GUARINO, N. (1998). Formal Ontology in Information Systems. Proceedings of the FOIS´98. Trento, Italy.

[HALEVY *et al*. 2004a] HALEVY, A. Y., IVES, Z., MADHAVAN, J., MORK, P., SUCIU, D., TATARINOV, I. (2004a). The Piazza peer data management system. IEEE Transactions on Knowledge and Data Engineering , pp. 787-798.

[HALEVY *et al*. 2004b] HALEVY, A., TATARINOV, I. (2004b). Efficient query reformulation in peer-data management systems. Proceedings of the SIGMOD Conference International Conference on Management of Data, pp. 539-550.Paris, France.

[HALEVY *et al*. 2003a] HALEVY, A., IVES, Z., SUCIU, D., TATARINOV, I. (2003a). Schema mediation in peer data management. Proceedings of the 19th IEEE International Conference on Data Engineering.

[HALEVY *et al*. 2003b] HALEVY, A., IVES, Z., MONK, P., TATARINOV, I. (2003b). Piazza: data management infrastructure for semantic. Proceedings of the 12th World Wide Web Conference (WWW), pp. 556-567. Budapest, Hungary.

[HALEVY *et al*. 2006] HALEVY, A., RAJARAMA, A., ORDILLE, J. (2006). Data Integration: The Teenage Years. Proceedings of the 32nd International Conference on Very large data bases, pp 9-16. Seoul, Korea.

[HERSCHEL *et al*. 2005] HERSCHEL, S., HEESE, R. (2005). Humboldt Discoverer: A semantic P2P index for PDMS. Proceedings of the International Workshop Data Integration and the Semantic Web. Porto, Portugal. Disponível em http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.144.503&rep=rep1&type=pdf.

[HOSE 2006] HOSE K., J. A. (2006). An Extensible, Distributed Simulation Environment for Peer Data Management Systems. Conference on Extending Database Technology, LNCS 3896 , pp. 1198-1202. Munique, Alemanha.

[HUANG *et al*. 2008] HUANG, H., XU, Y., SUN, Y., HUANG, L. (2008). Cluster-based load balancing multi-path routing protocol in wireless sensor networks. Proceedings of the 7th World Congress on Intelligent Control and Automation, pp. 6692 – 6696. Chongqing, China.

[ISHMANOV *et al*. 2009] ISHMANOV, F, KIM, S. W. (2009). Distributed Clustering Algorithm with Load Balancing in Wireless Sensor Network. Proceedings of the 2009 WRI World Congress on Computer Science and Information Engineering, vol. 01, pp. 19-23. Los Angeles, USA.

[JAGADISH *et al.* 2005] JAGADISH, H. V., OOI, B. C., VU, Q. (2005). BATON: A balanced tree structure for peer-to-peer networks. Proceedings of the 31st international conference on Very large data bases , pp. 661-672. Trondheim, Noruega.

[JAIN *et al*. 1984] JAIN, R. K., CHIU, D. W., HAWE, W. R. (1984). A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Techinical Report DEC-TR-301, Digital Computer Corporation.

[JOHNSTONE *et al*. 2005] JOHNSTONE, S., SAGE, P., MILLIGAN, P. (2005). iXChange – A Self-Organising Super Peer Network Model. Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC'05), vol. 00, pp. 164-169. Cartagena, Spain.

[JOUNG *et al*. 2009] JOUNG Y., CHUANG, F. (2009). "OntoZilla: An ontology-based, semi-structured, and evolutionary peer-to-peer network for information systems and services". Journal of Future Generation Computer Systems, vol. 25, n° 1, pp. 53-63.

[JUNIOR 2008] JUNIOR, H. C. (2008). Ontologias Emergentes: Uma abordagem para construção de ontologias a partir de mapeamentos ponto-a-ponto. Dissertação de Mestrado. Instituto Militar de Engenharia. Rio de Janeiro, Brasil.

[KAMIENSKI *et al*. 2005] KAMIENSKI, C., SOUTO, E., ROCHA, J., DOMINGUES, M., CALLADO, A., SADOK, D. (2005). Colaboração na Internet e a Tecnologia Peer-to-Peer. XXV Congresso da Sociedade Brasileira de Computação – SBC2005. São Leopoldo, Brasil.

[KANTERE 2007] KANTERE, V. (2007). Query Processing in P2P Overlay. PhD Thesis. National

Thecnical University of Athens. Athens, Greece.

[KANTERE *et al.* 2008] KANTERE, V; TSOUMAKOS, D.; SELLIS, T. (2008). A framework for semantic grouping in P2P databases. Journal Information Systems, vol. 33, pp. 611-636.

[KARGER *et al.* 2004] KARGER, D. R., RUHL, M. (2004). Simple efficient load balancing algorithms for peer-to- peer systems. *Proceedings of SPAA*, pp. 36–43. Barcelona, Spain.

[KARVOUNARAKIS *et al*. 2010] KARVOUNARAKIS, G., IVES, Z. G., TANNEN, V. (2010). Quering Data Provenance. Proceedings of the SIGMOD, 2010. Indianopolis, USA.

[KAZAA 2010] KAZAA. (2010). Acesso em 14 de Julho de 2010, disponível em http://www.kazaa.com/us/index.htm

[KIMBALL *et al.* 2002] KIMBALL, R., ROSS, M. (2002). The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling. 2nd Edition. Editor Robert Elliott.

[LAFFRANCHI 2004] LAFFRANCHI, M. M. (2004). Uma solução peer-to-peer para a implantação de jogos multiusuário baseada no padrão emergente MPEG-4 MU. Tese de Doutorado. Universisade Federal de São Carlos. São Carlos, Brasil.

[LI *et al*. 2005a] LI, J., VUONG, S. (2005a). Ontology-Based Clustering and Routing in Peer-to-Peer Networks. Proceedings of the 6th International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 791 - 795. Dalian, China.

[LI *et al*. 2005b] LI, Z. J., LIAO, M. H. (2005b). Modeling Load Balancing in Heterogeneous Unstructured P2P Systems. Journal of Computer Science , pp. 323-331.

[LODI *et al*. 2008a] LODI, S., MANDREOLI, F., MARTOGLIA, R., PENZO, W., SASSATELLI, S. (2008a). Semantic Peer: Here are the Neighbors You Want!. Proceedings of the 11th International Conference on Extending Database Technology (EDBT '08), pp. 26-37. Nantes, France.

[LODI *et al*. 2008b] LODI, S., MANDREOLI, F., MARTOGLIA, R., PENZO, W., SASSATELLI, S. (2008b). Boosting a Network of Semantic Peers. Proceedings of the Sixteenth Italian Symposium on Advanced Database Systems (SEDB '08), pp. 318-325. Mondello, Italy.

[LOEST 2007] LOEST, S. R. (2007). Um sistema de backup cooperativo tolerante a intrusões baseado em redes P2P. Dissertação de Mestrado. Pontifícia Universidade Católica do Paraná. Curitiba, Brasil.

[LUA *et al*. 2004] LUA, E., CROWCROFT, J., PIAS, M., SHARMA, R., LIM, S. (2004). A Survey and Comparison of Peer-to-Peer Overlay Network Schemes. IEEE Communications Survey and Tutorial, pp. 72-93.

[LV 2002] LV, Q. e. (2002). Search and Replication in Unstructured Peer-to-Peer Networks. Proceedings of the 16° ACM International Conference on Supercomputing(ICS'02), pp. 84-95. New York, USA.

[MA *et al*. 2006] MA, M., YANG, Y. (2006). Clustering and load balancing in hybrid sensor networks with mobile cluster heads. Proceedings of the 3rd international conference on Quality of service in heterogeneous wired/wireless networks.

[MANDREOLI *et al*. 2007] MANDREOLI, F., MARTOGLIA, R., PENZO, W., SASSATELLI, S., VILLANI, G. (2007). SRI@work: Efficient and Effective Routing Strategies in a PDMS. Proceedings of the 8th International Conference on Web Information Systems Engineering (WISE '07), pp. 285-297. Nancy, France.

[MANDREOLI *et al*. 2009] MANDREOLI, F., MARTOGLIA, R., SASSATELLI, S., VILLANI, G. (2008). Building a PDMS Infraestrutura for XML Data Sharing with SUNRISE. Proceedings of the 2008 EDBT workshop on Database technologies for handling XML information on the web, pp. 51-59. Nantes, France.

[MAUROX *et al*. 2007] MAUROX, P. C., AGARWAL, S., BUDURA, A., HAGHANI, P., ABERER, K. (2007). Self-Organizing Schema Mappings in the GridVine Peer Data Management System. Proceedings of the 33rd international conference on Very large data bases, pp. 1334-1337. Vienna, Austrian.

[McBRIEN *et al*. 2006] McBRIEN, P.J., POULOVASSILIS, A. (2006). P2P query reformulation over Both-as-View data transformation rules. Proceedings of the DBISP2P, pp. 310-322. Seoul, Korea.

[MICHAIL 2002] MICHAIL, D. V. (2002). Lobster- A load balanced P2P content sharing network. Masters thesis. Technical University of Crete. Crete, Greece.

[MICHLMAYR 2006] MICHLMAYR, E. (2006). Ant algorithms for search in unstructured peer-to-peer networks. Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW ’06) , pp. 142–146. Atlanta, USA.

[MONDAL *et al*. 2003] MONDAL, A., GODA, K., KITSUREGAWA, M. (2003). Effective load-balancing of peer-to-peer systems. Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid, pp.81-88. Área Central de Singapura, Singapura.

[MONTANELLI *et al.* 2007a] MONTANELLI, S., S., CASTANO, S., FERRARA, A. (2007a). Consensus-driven Formation of Semantic Communities of Peers. Relatório técnico. Università degli Studi di Milano. Milano, Italy.

[MONTANELLI *et al.* 2007b] MONTANELLI, S., BIANCHINI, D., AIELLO, C., BALDONI, R., BOLCHINI, C.,BONOMI, S., CASTANO, S., CATARCI, T., ANTONELLIS, V., FERRARA, A.,MELCHIORI, M., QUINTARELLI, E., SCANNAPIECO, M., SCHREIBER,A., TANCA, L. (2007b). Emergent Semantics and Cooperation in MultiKnowledge Environments: the ESTEEM Architecture. Proceedings of the VLDB International Workshop on Semantic Data and Service Integration (SDSI’2007). Vienna, Austrian.

[MONTANELLI *et al.* 2011] MONTANELLI, S., BIANCHINI, D., AIELLO, C., BALDONI, R., BOLCHINI, C.,BONOMI, S., CASTANO, S., CATARCI, T., ANTONELLIS, V., FERRARA, A.,MELCHIORI, M., QUINTARELLI, E., SCANNAPIECO, M., SCHREIBER,A., TANCA, L. (2011). The ESTEEM

platform: enabling P2P semantic collaboration through emerging collective knowledge. Journal of Intelligent Information Systems, vol. 36, n° 2.

[MONTRESOR 2004] MONTRESOR, A. (2004). A Robust Protocol for Building Superpeer Overlay Networks. Proceedings of the 4th International Conference on Peer-to-Peer Computing (P2P'04), pp. 202-209. Zurich, Switzerland.

[NAPSTER 2010] NAPSTER (2010). Acesso em 14 de Julho de 2010, disponível em http://www.napster.com

[NEIJDL *et al*. 2003] NEJDL, W., SIBERSKI, W., SINTEK, M. (2003). Design issues and challenges for RDF- and schema-based peer-to-peer systems. ACM SIGMOD Record, vol. 32, pp. 41 - 46.

[NG *et al*. 2002] NG, W. S., OOI, B. C., TAN, K.-L. (2002). BestPeer: A Self-Configurable Peer-to-Peer System. In Proceedings of the 18th International Conference on Data Engineering, p. 272. San Jose, USA.

[NOVÁK 2006] NOVÁK, D. (2006). Load Balancing in Peer-to-Peer Data Networks. Procedings of 2nd Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, pp. 1-7. Mikulov, Austrian.

[NOWELL *et al*. 2002] NOWELL, D. L., BALAKRISHNAN, H., KARGER, D. (2002). Analysis of the Evolution of Peer-to-Peer. Proceedings of the 21st annual symposium on Principles of distributed computing, pp. 233 - 242. Monterey, USA.

[O'NEIL *et al*. 1993] O'NEIL, E., O'NEIL, P., WEIKUM, G. (1993). The LRU-K page replacement algorithm for database disk. Proceedings of the 1993 ACM Sigmod Internacional Conference on Management of Data, vol. 22, pp. 297-306.

[OOI *et al*. 2003] OOI, B. C., SHU, Y., TAN, K.-L. (2003). Relational Data Sharing in Peer-based Data Management Systems. ACM SIGMOD Record, vol. 32, pp. 59-64.

[OTTO 2007] OTTO F., M. P. (2007). A Model for Data Management in Peer-to-Peer Systems. International Journal of Computing and ICT Research, ISSN 1996-1065 (online),vol.1, No. 2 ,pp. 67.

[PIRES 2007] PIRES, C. (2007). Um Sistema P2P de Gerenciamento de Dados com Conectividade Baseada em Semântica. Trabalho de Qualificação e Proposta de Tese. Universidade Federal de Pernambuco.Recife, Brasil.

[PIRES 2009] PIRES, C. (2009). Ontology-based *Cluster*ing in a Peer Data Management System. Tese de Doutorado. Universidade Federal de Pernambuco.Recife, Brasil.

[PIRES *et al*. 2011] PIRES, C.E., SOUZA, D. Y., LÓSCIO, B. ; BELIAN, R., TEDESCO, P. C. A. R. , SALGADO, A. C. (2011). Using Ontologies to Enhance Data Management in Distributed Environments. In: Ibero American Meeting on Ontological Research, 2011, Gramado. Proceedings of the Ibero American Meeting on Ontological Research (To appear).

[PITOURA *et al.* 2006] PITOURA, T., NTARMOS, N., TRIANTAFILLOU, P. (2006). Replication, Load Balancing and Efficient Range Query Processing in DHTs. Proceedings of the 10th International Conference on Extending Database Technology (EDBT). Munich, Germany.

[QIAO *et al*. 2009] QIAO, Y., BOCHMANN, G. (2009). Applying a diffusive load balancing in a clustered P2P system. Proceedings of 9th Internacional Conference on New Technologies of Distributed Systems (NOTERE), pp. 189-199. Montreal, Canada.

[RATNASAMY 2001] RATNASAMY, S. F. (2001). A scalable content-addressable network. Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, pp. 161-172. San Diego, USA.

[REZENDE 2009] REZENDE, E. D. (2009). Modelo Estrutural para Compartilhamento de Arquivos Peer-to-Peer. Dissertação de Mestrado. Universidade Estadual Paulista. São José do Rio Preto, Brasil.

[RISSON *et al*. 2006] RISSON, J., MOORS, T. (2006). Survey of Research towards Robust Peer-to-Peer Networks. The International Journal of Computer and Telecommunications Networking archive, vol. 50, pp. 1-36. New York, USA.

[ROTH *et al*. 2007] ROTH, A; NAUMANN, F.(2007). System p: Completeness-driven query answering in peer data management systems. Proceedings of the BTW '07, pp. 625-628. Fort Collins, USA.

[ROUSSE *et al*. 2006] ROUSSE, C., BERMAN, S. (2006). A Scalable P2P Database System with Semi-Automated Schema Matching. Proceedings of the 26th IEEE International Conference Workshops on Distributed Computing Systems (ICDCSW'06), pp. 78. Lisboa, Portugal.

[SADRI 2011] SADRI, F. (2011). Ambient intelligence: A survey. In: Journal ACM Computing Surveys (CSUR), vol. 43, n° 4.

[SARMA *et al.* 2008] SARMA, A. D., DONG, X., HALEVY, A. Y. (2008). Bootstrapping pay-as-you-go data integration systems. Proceedings of the SIGMOD, pp. 861-874. Vancouver, Canada.

[SEWASIE 2010] SEWASIE. (5 de agosto de 2010). SEmantic Webs and AgentS in Integrated Economies. Acesso em 5 de agosto de 2010, disponível em http://www.sewaise.org.

[SOULSEEK 2010] SOULSEEK (2010). Soulseek - Slsknet - File sharing software. Acesso em 14 de Julho de 2010, disponível em http://www.slsknet.org.

[SOUZA 2007] SOUZA, D. Y. (2007). Reformulação de consulta baseada em semântica para PDMS. Trabalho de Qualificação e Proposta de Tese. Universidade Federal de Pernambuco.Recife, Brasil.

[SOUZA 2009] SOUZA, D. Y. (2009). Using Semantics to Enhance Query Reformulation in Dynamic Distributed Environments. Tese de Doutorado. Universidade Federal de Pernambuco.Recife, Brasil.

[STOICA 2001] STOICA, I. M. (2001). Chord: A scalable peer-to-peer lookup service. Proceedings of the Conference on Applications, technologies, architectures, and protocols for computer communications, pp. 149-160. San Diego, USA.

[STOICA *et al*. 2003] STOICA, I., RAO, A., LAKSHMINARAYANAN, K., SURANA, S., KARP, R. (2003). Load Balancing in Structured P2P Systems. Proceedings of the IPTPS. Berkeley, USA.

[SUNG 2005] SUNG, L. G. (2005). A Survey of Data Management in Peer-to-Peer Systems. School of Computer Science, University of Waterloo. Disponível em http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.6553&rep=rep1&type=pdf

[SUNRISE 2010] SUNRISE. (2010). ISGroup - Information Systems Group. Acesso em 26 de agosto de 2010, disponível em http://www.isgroup.unimo.it/sunriseProject.asp

[SURI *et al*. 2010] SURI, P. K., SINGH, M. (2010). An efficient decentralized Load Balancing Algorithm for grid. Proceedings of the 2nd International Advance Computing Conference (IACC), pp. 10-13. Patiala, India.

[TATARINOV *et al*. 2003] TATARINOV, I., IVES, Z., MADHAVAN, J., HALEVY, A., SUCIU, D., DALVI, N. (2003). The Piazza Peer Data Management Project. ACM SIGMOD Record, vol. 32, pp. 47-52.

[TOWSLEY 2003] TOWSLEY, D. (2003). Peer-to-peer Networking. Tutorial no SBRC2003. Acesso em 4 de abril de 2011, disponível em http://www.cin.ufpe.br/~gprt/gtp2p/tutoriais/p2p03-tutorial_towsley.pdf.

[TRIANTAFFILOU *et al*. 2003] TRIANTAFFILOU, P., XIRUHAKI, C., KOUBARAKIS, M., NTARMOS, N. (2003). Towards high performance peer-to-peer content and resource sharing systems. Proccedings of CIDR. Asilomar, USA.

[USCHOLD *et al*. 1999] USCHOLD, M., JASPER, R. (1999). A Framework for Understanding and Classifying Ontoloy Applications. Proceedings of the 12th International Workshop on knowledge Acquisition Modeling e Management. Banff, Canada.

[VALDURIEZ *et al*. 2004] VALDURIEZ, P., PACITTI, E. (2004). Data Management in Large-scale P2P Systems. Proceedings of the International Conference on High Performance Computing for Computational Science. Valencia, Spain. Disponível em http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.4652&rep=rep1&type=pdf

[VU *et al.* 2010] VU, Q. M., LUPU, M., OOI, B. C. (2010). Peer-to-Peer Computing - Principles and Applications. Editora Springer.

[W3C 2011] W3C. (2011). W3C, World Wide Web Consortium. Acesso em 6 de setembro de 2011, disponível em http://www.w3.org/TR/webont-req.

[WIEDERHOLD 1992] WIEDERHOLD, G. (1992). Mediators in the Architecture of Future Information Systems. IEEE Computer, vol. 25, pp. 38–49.

[WSDL 2011] WSDL (2011). Web Service Definition Language (WSDL). Acesso em 23 de agosto de 2011, disponível em http://www.w3.org/TR/wsdl.

[XIAO 2006] XIAO, H. (2006). Query processing for heterogeneous data integration. PhD Thesis. University of Illinois. Chicago, USA.

[XPEER 2010] XPEER. (2010). XPeer Project. Acesso em 26 de agosto de 2010, disponível em http://www.di.unipi.it/~manghi/XPeerWeb/homeXPeer.htm

[XU 2011] XU, J. (2011). Supporting Domain Heterogeneous Data Sources for Semantic Integration. PhD Thesis. University of British Columbia. Vancouver, Canada.

[YAGOUBI 2010] YAGOUBI, B., MEDDEBER, M. (2010). Distributed Load Balancing Model for Grid Computing. ARIMA Journal, vol. 12, pp. 43-60.

[ZAFALON 2009] ZAFALON, G. F. (2009). Algoritmos de alinhamento múltiplo e técnicas de otimização para esses algoritmos utilizando Ant Colony. Dissertação de Mestrado. Universidade Estadual Paulista. São José do Rio Preto, Brasil.

[ZHAO 2001] ZHAO, B. K. (2001). Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Techical Report: CSD-01-1141, Computer Science Division.

[ZHAO 2006] ZHAO, J. (2006). Schema Mediation and Query Processing in Peer Data Management Systems. Master Thesis. University of British Columbia. Kelowna, Canada.

[ZHU *et al*. 1998] ZHU, H., YANG, T., ZHENG, Q., WATSON, D., IBARRA, O. H., SMITH, T. (1998). Adaptive Load Sharing for Clustered Digital Library Servers. Proceedings of the 7th IEEE international Symposium on High Performance Distributed Computing. Chicago, USA.

[ZHUANG *et al*. 2004] ZHUANG, Z., LIU, Y., XIAO, L. (2004). Dynamic Layer Management in Super-Peer Architectures. Proceedings of the International Conference on Parallel Processing (ICPP'04), pp. 29-36. Montreal, Canada.