Gerenciamento de Domínios de Aplicação através do Uso de Ontologias

Deise de Brum Saccol[1], Nina Edelweiss, Renata de Matos Galante[2], Márcio Roberto de Mello

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{deise,nina, galante}@inf.ufrgs.br, mrmello@gmail.com

Trabalho de Pós-Graduação

Abstract. *In P2P systems, files from the same application domain are spread over the network, which causes inefficiency when the user poses queries. To solve this problem, this work proposes to cluster documents from the same domain into super peers. Thus, files related to the same universe of discourse are grouped and the query processing is restricted to a subset of the network. In our proposal, the clustering task is performed by the ontology manager and it is composed of three phases: ontology generation, document and ontology matching, and metadata management. This paper describes the ontology manager and presents the first phase in more details.*

Resumo. *Em sistemas P2P, arquivos do mesmo domínio de aplicação estão espalhados pela rede, o que pode gerar ineficiência na busca de recursos quando o usuário submete consultas. Para solucionar este problema, este trabalho propõe agrupar documentos do mesmo domínio de aplicação em super peers. Desta forma, arquivos relacionados a um mesmo universo de discurso são agrupados e o processamento de consultas é realizado somente em um subconjunto da rede. Neste trabalho, a tarefa de agrupamento é realizada por um módulo gerenciador de ontologias, composto por três etapas: geração da ontologia, casamento entre a ontologia e arquivos, e gerenciamento de metadados. Este artigo descreve o gerenciador de ontologias e apresenta a primeira etapa em mais detalhes.*

1. Introdução

O termo *peer-to-peer* (P2P) refere-se a sistemas de aplicações que usam recursos distribuídos para realizar tarefas em um contexto descentralizado [Gnutella 2007][Kazza 2007][eMule 2007]. A usabilidade destes sistemas é dependente principalmente das técnicas utilizadas para encontrar e recuperar os recursos desejados. A qualidade de tais resultados pode ser medida por algumas métricas, tais como: tamanho do conjunto de resultados, satisfação do usuário com os resultados retornados e tempo gasto no processamento [Yang and Garcia-Molina 2002].

[1] Este trabalho é parcialmente financiado pelo CNPQ (processo 142396/2004-4).
[2] Este trabalho foi parcialmente suportado pelo projeto Pronex FAPERGS número 0408933.

No entanto, a otimização da busca de recursos encontra dois problemas. O primeiro problema é como localizar arquivos relevantes para uma consulta, com baixo custo. Arquivos pertencentes a um mesmo domínio de aplicação e que são necessários para responder uma determinada consulta podem estar espalhados em vários *peers*. O uso da técnica *flooding* (inundação) é necessário para acessar todos os arquivos existentes. Esta técnica é cara, uma vez que todos os *peers* recebem a mensagem de consulta e geralmente apenas alguns deles estão aptos a respondê-la. Outra abordagem para balancear custo é usar alguma técnica variante da *flooding*, tais como busca em largura e em profundidade na árvore formada pelos *peers*. No entanto, o uso destas variantes não garante o retorno ótimo dos resultados, uma vez que nem todos os *peers* recebem a mensagem da consulta.

O segundo problema advém da falta de semântica dos recursos. Considerando duas aplicações que necessitam trocar dados, uma abordagem possível é construir um adaptador que transforme dados e estrutura entre elas. Entretanto, a construção destes adaptadores é difícil e requer conhecer a organização dos dados em ambas aplicações. Além disso, a complexidade e o tempo de desenvolvimento tendem a ser quadráticos em relação ao número de aplicações componentes [Staab and Studer 2004]. Uma outra solução pode utilizar metadados para descrever a semântica dos recursos. Mas este cenário traz duas questões [Mena and Illarramendi 2001]: (i) como lidar com diferentes conceitos usados para descrever a mesma informação; e (ii), como adquirir e manter os metadados para tratar o problema de compartilhamento de vocabulários.

Para solucionar estes problemas, este artigo detalha o gerenciador de ontologias, uma abordagem que pode ser utilizada para melhorar as técnicas tradicionais de busca em sistemas P2P. O trabalho foca em dois pontos principais: (i) o tráfego extra produzido por técnicas tradicionais de busca sempre que os resultados ótimos são desejados; e (ii) a falta de semântica no armazenamento e busca das informações. A fim de evitar tal tráfego desnecessário, esta proposta baseia-se no agrupamento de arquivos em *super peers*, com base no domínio de aplicação a que estes pertencem. Desta forma, todos os *peers* que possuem arquivos pertencentes a um mesmo domínio de aplicação são conectados logicamente (agrupados) ao *super peer* descrito pela ontologia representativa deste domínio; consequentemente, consultas de um certo domínio são roteadas apenas dentro da sub-rede formada pelos *peers* que pertencem a este domínio.

O uso de ontologias tem como objetivo tornar explícito o conteúdo dos recursos, independente da estrutura de como a informação está armazenada. Desta forma, o agrupamento de *peers* adiciona semântica aos arquivos ao mesmo tempo em que reduz o tráfego desnecessário na rede. Esta abordagem permite aos usuários formular consultas com base na ontologia e deixar ao sistema a responsabilidade pelo gerenciamento da heterogeneidade e distribuição dos *peers*. Tais funcionalidades são realizadas pelo gerenciador de ontologias como parte do *DetVX* [Saccol, Edelweiss and Galante 2007], um ambiente para detecção, gerenciamento e consulta a réplicas e versões de documentos XML em um contexto P2P.

A principal contribuição deste artigo é a especificação do gerenciador de ontologias, componente usado para permitir o agrupamento de arquivos com base no domínio de aplicação. Também é apresentada uma visão geral do framework *DetVX*, visando esclarecer a interação entre o gerenciador de ontologias e os demais módulos do ambiente. Por fim, o artigo detalha uma das etapas do gerenciador de ontologias, a geração de ontologias, implementada na ferramenta *Ontogen*.

Este artigo está organizado como segue: a seção 2 apresenta o *framework* utilizado como base neste trabalho. A seção 3 descreve o gerenciador de ontologias, componente

principal do mecanismo aqui proposto. Na seção 4 é apresentado o funcionamento do gerador de ontologias, a partir da integração dos esquemas dos arquivos participantes. Trabalhos relacionados e conclusões são descritos nas seções 5 e 6, respectivamente.

2. O Framework *DetVX*

No ambiente *DetVX* (Detector de Réplicas e Versões de Documentos XML), cada arquivo é armazenado em um *peer*. Réplicas e versões destes arquivos podem estar armazenadas no mesmo ou em diversos *peers.* Os arquivos compartilhados podem pertencer a domínios de aplicação diferentes, como por exemplo: arquivos referentes a currículos, arquivos referentes a projetos de pesquisa, arquivos referentes a cadastro de alunos, etc. Assume-se que cada arquivo pertence a um único domínio de aplicação, descrito por uma ontologia, conforme mostrado na Figura 1. Desta forma, a busca por réplicas e versões de um documento dá-se somente entre os arquivos que pertencem a um mesmo domínio (isto é, que são descritos pela mesma ontologia).

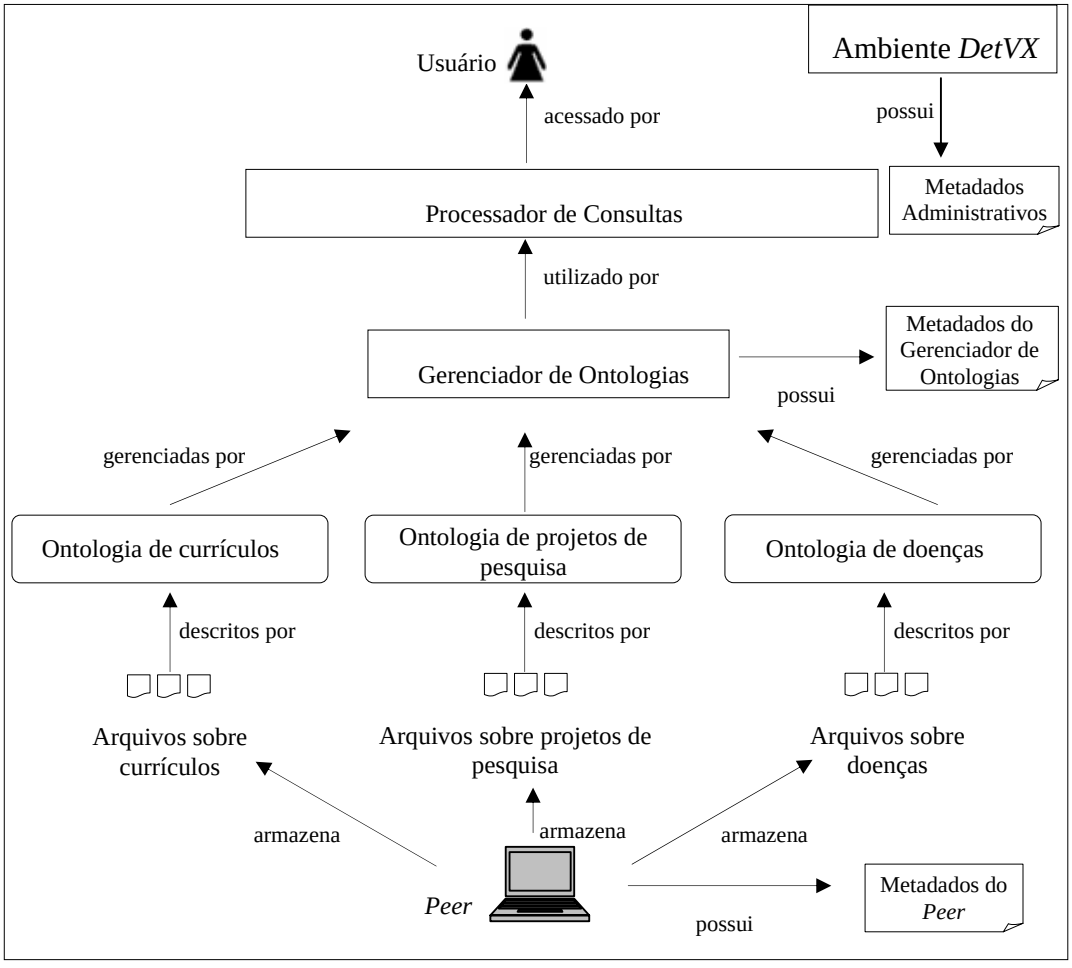

Figura 1. Funcionamento geral do *DetVX*

O uso de domínios de aplicação traz duas vantagens no ambiente proposto:

1) Restringe o espaço de busca na detecção de réplicas e versões: não é necessário buscar por uma versão ou réplica de documentos entre arquivos pertencentes a domínios de aplicação diferentes;

2) Aumenta a eficiência no processamento de consultas: consultas relativas a um determinado domínio de aplicação não são processadas sobre arquivos que não possuem conceitos pertencentes a este domínio.

Conforme observado na Figura 1, um *peer* pode armazenar arquivos descritos por diferentes ontologias. Cada *peer* possui um conjunto de metadados, os quais descrevem o IP do *peer* e informações sobre os seus arquivos, como: identificadores de arquivos, identificadores de documentos, resultados *hash* de arquivos, tempos de registro e última data de atualização de cada arquivo no *peer*. As ontologias são gerenciadas por um componente chamado de gerenciador de ontologias, o qual possui um conjunto de metadados necessário ao gerenciamento dos diversos domínios de aplicação. Finalmente, o usuário acessa os arquivos através da submissão de consultas ao processador de consultas, formuladas com base em uma determinada ontologia. Esta consulta é relacionada a um domínio de aplicação. O *peer* processa a consulta e retorna os resultados ao usuário.

Um determinado documento pode estar replicado ou multiversionado em vários *peers*. Se o usuário submete uma consulta a um *peer* solicitando o endereço atual de uma pessoa, o sistema deve retornar somente a última versão desta informação. Para descobrir qual(is) arquivo(s) é(são) necessário(s) acessar para responder uma determinada solicitação, o sistema consulta os metadados disponíveis no ambiente. A fim de fornecer as funcionalidades para o *framework*, este trabalho propõe a arquitetura mostrada na Figura 2.

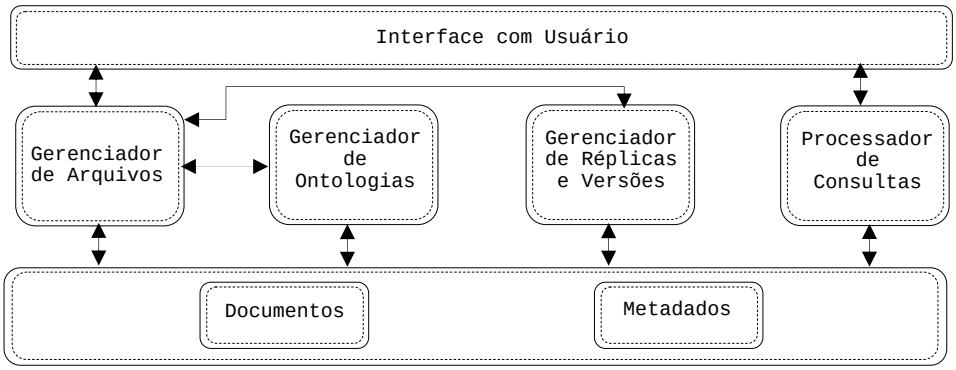

Figura 2. Arquitetura do Framework *DetVX*

Os módulos propostos na arquitetura possuem as funcionalidades descritas a seguir. O módulo de gerenciamento de arquivos é responsável por registrar arquivos no sistema, além de verificar periodicamente se o *peer* teve alguma modificação em seus arquivos compartilhados. O módulo de gerenciamento de ontologias é responsável pela manutenção do repositório de ontologias e pela associação de ontologias a arquivos (agrupamento de arquivos). O módulo de gerenciamento de réplicas e versões é responsável por identificar e representar réplicas e versões de documentos. O módulo de processamento de consultas é responsável por verificar quais os arquivos necessários para processar uma determinada consulta, processá-la e retornar os resultados ao usuário final.

A seção a seguir detalha o módulo de gerenciamento de ontologias.

3. Gerenciador de Ontologias

Este módulo é responsável pela manutenção do repositório de ontologias e pela associação de ontologias à arquivos. As ontologias são armazenadas no repositório de ontologias, um conjunto de arquivos OWL armazenados em um sistema de arquivos. As atividades do

gerenciador de ontologias são mostradas na Figura 3 e detalhadas a seguir.

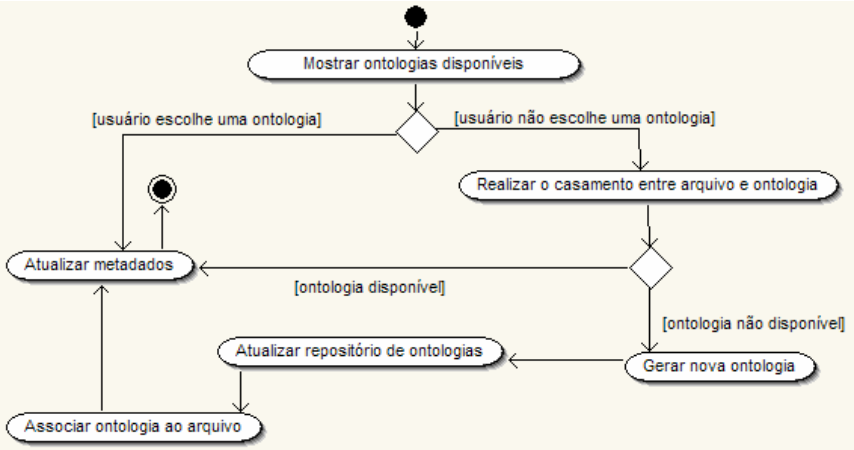

Figura 3. Diagrama de atividades do gerenciador de ontologias

- *mostrar ontologias disponíveis* - esta atividade retorna uma breve descrição de cada ontologia armazenada no repositório de ontologias, contendo os principais termos e relações que representam o domínio de conhecimento. Desta forma, o usuário pode escolher uma ontologia para ser utilizada para cada conjunto de documentos pertencentes a um mesmo domínio de aplicação. Uma vez que um *super peer* é descrito por apenas uma ontologia, escolher uma ontologia significa selecionar um *super peer* para se conectar;

- *realizar o casamento entre o arquivo e uma ontologia* – caso o usuário não escolha uma das ontologias disponíveis, então esta atividade é responsável por sugerir uma ontologia que descreva os conceitos existentes no(s) arquivo (s). Esta tarefa assume que existe uma ontologia compatível para o documento. Desta forma, o *peer* que armazena o arquivo é conectado ao *super peer* descrito por esta ontologia. Para descobrir qual ontologia melhor descreve um arquivo, este trabalho baseia-se em medidas de similaridade. A ontologia que apresenta maior similaridade (acima de um limiar) é escolhida para representar o domínio de aplicação do arquivo. Um *thesaurus* é usado para auxiliar na descoberta dos relacionamentos terminológicos, conforme descrito em [Noll, Saccol and Edelweiss 2007].

- *gerar nova ontologia* - esta tarefa será realizada sempre que o sistema não encontrar uma ontologia adequada para um (conjunto de) arquivos (s) localizados no *peer*. A geração de ontologias é feita a partir da integração dos esquemas dos arquivos compartilhados e é detalhada na seção 4.

- *atualizar o repositório de ontologias* - esta atividade é responsável por inserir a ontologia criada no repositório de ontologias;

- *associar uma ontologia ao arquivo* - supondo que uma nova ontologia foi criada a partir de arquivos XML, esta ontologia precisa ser associada a estes arquivos. Alguns metadados precisarão ser atualizados;

- *atualizar metadados* - esta atividade atualiza os metadados do gerenciador de ontologias. Os metadados descritos permitem especificar qual a ontologia (`ontology Oid`) que descreve um determinado arquivo (`fileID`), um rótulo descritivo do domínio (`domain`) e o arquivo em que a ontologia encontra-se armazenada (`file`), conforme mostrado na Figura 4.

```
<AssociatedOntologies>
  <ontology Oid="Ont1"><domain>Research  Projects</domain> <file>ResearchProjects.owl</file>
    <Files>
      <peer id="P1"><fileID>F2</fileID> <fileID>F7</fileID> <fileID>F8</fileID></peer>
    </Files></ontology>
  <ontology Oid ="Ont2"><domain>Diseases</domain> <file>Diseases..owl</file>
    <Files>
      <peer id="P1"><fileID>F66</fileID></peer>
      <peer id="P2"><fileID>F17</fileID> <fileID>F28</fileID></peer>
    </Files></ontology>…</AssociatedOntologies >
```

Figura 4. Metadados do gerenciador de ontologias

Estes metadados são mantidos sempre que a atividade de casamento de ontologias e arquivos é realizada. Para um arquivo (`fileID`) armazenado no *peer* (`peer id`), associado a uma ontologia (`ontology`), os metadados descrevem o respectivo domínio (`domain`) a que este arquivo pertence. Dada uma consulta pertencente a um determinado domínio, o sistema acessa os metadados e verifica quais arquivos podem responder a consulta.

As atividades detalhadas na Figura 4 são estruturadas na arquitetura a seguir.

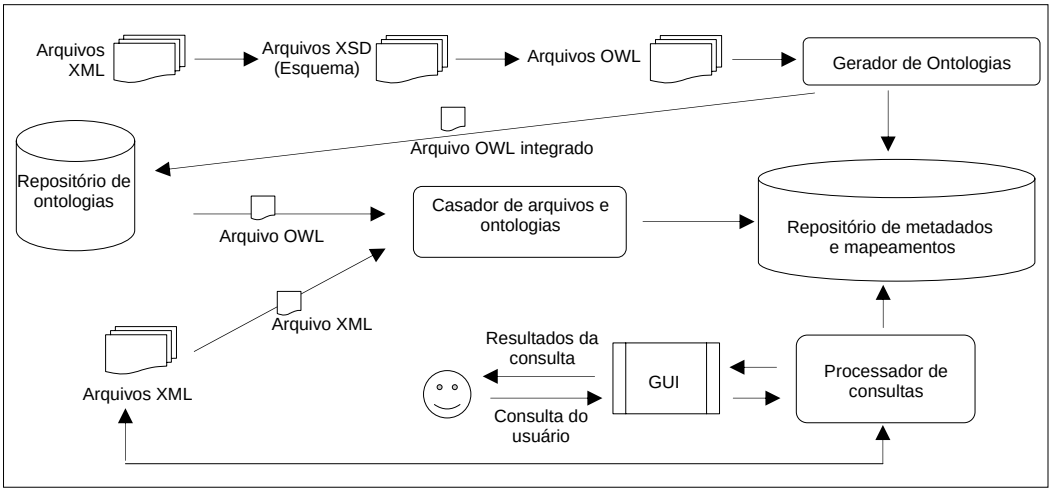

Figure 5. Arquitetura do gerenciador de ontologias

O gerenciador de ontologias encapsula a heterogeneidade dos recursos, fornecendo informações sobre as ontologias existentes e os arquivos compartilhados. O gerenciador de ontologias também é responsável por gerar novas ontologias quando nenhuma das existentes é apropriada para um certo (conjunto de) arquivo (s) XML; a geração da ontologia é feita a partir da integração dos esquemas dos arquivos envolvidos. O casamento entre arquivos e ontologias é responsável pela identificação da ontologia que melhor descreve um arquivo. Finalmente, o processador de consultas permite ao usuário escolher uma ontologia e formular consultas com base nesta ontologia.

A próxima seção descreve o gerador de ontologias.

4. Gerador de Ontologias

Este módulo é responsável por gerar uma ontologia que descreva os conceitos e relacionamentos existentes em um domínio de aplicação. A ontologia é criada a partir da integração de esquemas, como descrito. Primeiramente, gera-se o esquema XML correspondente ao arquivo compartilhado. O esquema *xsd* é então traduzido para um formato OWL, o qual basicamente enfatiza os elementos complexos (i.e., não léxicos) e atômicos (i.e., léxicos). Através da aplicação de várias regras de integração, um esquema

integrado é gerado, também representado em OWL.

O mecanismo considera três tipos de elementos no processo de integração: atômico *x* atômico, complexo *x* não complexo, e complexo *x* atômico. O processo de integração segue uma abordagem binária e utiliza um *thesaurus* para auxiliar na inferência de relacionamentos terminológicos, a fim de realizar a equivalência de conceitos. A geração de ontologias está implementada na ferramenta *Ontogen*, conforme descrito a seguir.

4.1 Integração dos Esquemas

O objetivo geral da ferramenta *OntoGen* é a integração de esquemas XML, gerando ao final do processo uma ontologia que represente os esquemas integrados. No primeiro nível estão os documentos de entrada, que podem ser documentos XML ou esquemas XML representados na linguagem *XML Schema*. Os esquemas XML são repassados diretamente para a próxima etapa, enquanto os documentos XML precisam passar por uma etapa a mais, em que a ferramenta deve extrair o esquema do documento XML de entrada, baseando-se nos dados e na estrutura do documento. Esta etapa é feita automaticamente, utilizando uma funcionalidade da biblioteca *Castor* [Castor 2007]. No último nível está o artefato de saída da ferramenta: o esquema global. Para obter o esquema global, os esquemas locais devem passar por duas etapas que compõem os dois módulos principais da implementação: a conversão do XSD e a integração semântica.

O módulo de conversão do XSD é responsável pela conversão de cada esquema XML em modelos conceituais canônicos. O uso de um modelo conceitual facilita a etapa posterior, a integração semântica, pois permite uma maior abstração do esquema XML, deixando de lado os detalhes técnicos de estruturação e colocando o foco na semântica dos seus dados [Mello 2002]. Em [Garcia 2005] é apresentada uma ferramenta que tem o objetivo de extrair um modelo conceitual canônico de um documento *XML Schema*. Esta ferramenta foi gentilmente cedida pelo grupo de banco de dados da UFSC (GBD – UFSC), para ser incorporada na ferramenta *OntoGen* que está sendo proposta neste trabalho.

O módulo de integração semântica é responsável pela integração das ontologias geradas na etapa de conversão do XSD. Na integração semântica são aplicadas as regras de unificação, para que seja gerada ao final do processo uma única ontologia, no mesmo formato das ontologias geradas anteriormente. Essa ontologia deve representar todas as ontologias de entrada e será considerada a ontologia global, objeto de saída da ferramenta.

Os dois módulos principais da integração semântica são detalhados a seguir.

4.1.1 Agrupamento de Sinônimos

É responsável por verificar equivalência semântica entre conceitos de diferentes ontologias, agrupando os conceitos equivalentes em *clusters* de afinidade. Este módulo recebe duas ontologias, representadas na linguagem OWL. Para cada conceito da primeira ontologia, o agrupador de sinônimos procura um conceito semanticamente equivalente na segunda ontologia. Uma função de afinidade é a responsável por comparar dois conceitos, devolvendo um coeficiente de afinidade que indica se os dois conceitos são semanticamente equivalentes ou não. Os conceitos semanticamente equivalentes são agrupados em *clusters* de afinidade.

Função de Afinidade: a função de afinidade é a função responsável por verificar se dois conceitos, de ontologias diferentes, são semanticamente iguais ou não. Por questões de simplificação, a função está baseada unicamente no nome dos conceitos. Se os nomes dos conceitos forem iguais, ou forem sinônimos, segundo um dicionário de sinônimos, estes dois

conceitos serão tomados como semanticamente iguais e a função de afinidade retornará um. Caso contrário, a função de afinidade retornará zero. O dicionário de sinônimos utilizado é o *WordNet*. Exemplos:

Afinidade("*Author*","*Paper*") = 0 Afinidade("*Paper*","*Paper*") = 1 Afinidade("*Author*","*Writer*") = 1

Cluster de Afinidade: um *cluster* de afinidade pode ser composto por no mínimo um e no máximo dois conceitos, pois a estratégia de unificação é binária. Ele será composto por dois conceitos quando a função de afinidade para estes conceitos retornar o valor um, e por um conceito apenas, caso este conceito não tenha conceito semanticamente igual na outra ontologia. Exemplo:

- Conceitos da ontologia 1: *author*, *paper*, *university*;

- Conceitos da ontologia 2: writer, paper, *title*;

Clusters gerados: c1 = {author, writer}, c2 = {paper, paper}, c3 = {university}, c4 = {title}

O conjunto de *clusters* de afinidade obtido através do agrupamento de sinônimos de duas ontologias é repassado para a etapa de unificação.

4.1.2 Unificação

É responsável por integrar *clusters* de conceitos, a fim de gerar o esquema global. Essa unificação é baseada no conjunto de *clusters* de afinidade, gerado no módulo de agrupamento de sinônimos, e nas regras de unificação definidas em [Mello 2002]. As regras de unificação propostas em [Mello 2002] foram simplificadas e adaptadas para a proposta deste trabalho, detalhado em [Mello 2007]. Para cada *cluster* de afinidade será gerado um conceito na ontologia global, resultado da unificação dos dois conceitos pertencentes ao *cluster*. Caso o *cluster* seja composto por um conceito apenas, este conceito será mapeado diretamente para a ontologia global, não sendo necessário aplicar as regras de unificação.

Existem três casos possíveis de unificação, para três tipos de *clusters* de afinidade: unificação léxica (conceitos léxicos), unificação não léxica (conceitos não léxicos) e unificação mista (conceito léxico e conceito não léxico). As regras implementadas baseiam-se em:

Unificação de Nomes: caso o *cluster* seja composto por dois conceitos, *Ci* e *Cj*, e estes dois conceitos tenham nomes diferentes, o nome do conceito gerado a partir do *cluster* será igual ao nome do conceito *Ci*.

Unificação de Tipos: caso o *cluster* léxico seja composto por dois conceitos, *Ci* e *Cj*, e estes dois conceitos tenham tipos de dados diferentes, o tipo de dados do conceito gerado será igual ao tipo mais genérico dentre os tipos de dados dos conceitos *Ci* e *Cj*. Exemplos:

{date, date} => date; {integer, float} => float; {string, integer} => string; {integer, date} => string;

Unificação de Relacionamentos: para cada relacionamento *Ri* da primeira ontologia é feita uma busca por um relacionamento *Rj* na segunda ontologia que possua *afinidade* com *Ri*. Caso o relacionamento *Ri* possua afinidade com o relacionamento *Rj*, estes dois relacionamentos devem ser unificados, observando as regras descritas em [Mello 2007].

Unificação de Cardinalidades: dadas duas cardinalidades, *Cardi* e *Cardj*, a cardinalidade unificada será a mais abrangente. Exemplos:

{"(0,1)", "(1,N)"} => "(0,N)"; {"(1,1)", "(0,1)"} => "(0,1)"; {"(1,1)", "(1,N)"} => "(1,N)";

A descrição completa da ferramenta *Ontogen* pode ser encontrada em [Mello 2007].

5. Trabalhos Relacionados

Vários trabalhos tem referenciado o problema da heterogeneidade entre fontes de dados. Em [Broekstra, Ehrig, Haase and et al. 2003], um modelo de dados é proposto para representar informação semântica que combina características de ontologias (hierarquia de conceitos) com uma descrição e um modelo que possibilita manipular visões heterogêneas sobre um determinado domínio. [Boyd, Kittivoravitkul, Lazanitis, McBrien and Rizopoulos 2004] descreve o repositório *AutoMed*, o qual fornece a implementação para a abordagem *both-as-view* de integração de dados. [Xu and Embley 2003] propõe *TIQS*, uma abordagem para integração de dados que usa casamento semi-automático de esquemas para mapeamentos, com base no esquema conceitual global pré-definido.

Sistemas baseados em palavras-chave não recuperam o documento necessário se um sinônimo é utilizado como parte da consulta [Freenet 2007]. Esta situação ocorre porque diferentes, mas relacionados, termos podem ser usados para descrever informação equivalente. Além disso, sistemas automáticos falham nas tarefas de encontrar e extrair informação relevante, e integrar esta informação distribuída por várias fontes [Fensel 2001]. Para tratar estas questões, anotações (metadados) da proposta apresentada neste artigo possibilitam definições mais ricas dos documentos, possibilitando ao usuário a submissão de consultas independentes da localização e estrutura dos recursos.

O problema da agregação de *peers* em *super peers* é uma questão importante. Esta tarefa pode ser realizada de acordo com algumas características, como por exemplo, assunto e localidade. Na proposta deste trabalho, utiliza-se uma ontologia como critério de agrupamento de *peers*. Nesta etapa, é necessário descobrir a qual domínio de aplicação um determinado documento pertence. Para isso, alguns trabalham encontrados na literatura apresentam técnicas para realização de casamento entre esquemas e ontologias, baseado em técnicas de similaridade. Casamento entre esquemas e ontologias tem como objetivo identificar correspondências semânticas entre estruturas de metadados ou modelos e ontologias.

Um dos trabalhos encontrados na literatura com este propósito é o sistema COMA++ [Coma++ 2007], uma ferramenta de casamento entre esquemas e ontologias. O sistema suporta os modelos XSD (*W3C XML Schema*), OWL (*Web Ontology Language*), XDR (*XML Data Reduced*) e esquemas relacionais [Aumueller, Do, Massmann and Rahm 2005]. A medida de similaridade entre dois modelos é determinada por um função de similaridade entre dois elementos pertencentes a taxonomia (descritos pela ontologia). No entanto, não é detalhado como é feito o cálculo da similaridade entre os modelos e a taxonomia.

6. Conclusões

Motores de busca existentes fornecem suporte para recuperação automática de informações. No entanto, a tarefa de extrair a informação relevante fica para o usuário. Desta forma, há alguns desafios que devem ser superados, tais como [Fensel 2001]: falta de mecanismos para representar e traduzir informações heterogêneas e descrições de conteúdo. Conseqüentemente, há uma clara necessidade por mais semântica e interoperabilidade.

Considerando sistemas P2P, há um gargalo extra: aumentar a qualidade dos resultados enquanto otimiza-se o espaço de busca. Neste cenário, duas questões devem ser tratadas: como encontrar arquivos relevantes para a consulta do usuário (a baixo custo) e como aumentar a semântica dos recursos disponíveis. Para solucionar estes problemas, este trabalho propõe uma abordagem baseada em ontologias que pode ser utilizada para aperfeiçoar as técnicas tradicionais de busca em sistemas P2P. O mecanismo proposto reduz

o tráfego e aumenta a semântica com relação ao armazenamento e busca das informações. A otimização do espaço de busca é feita pelo agrupamento de arquivos em *super peers*. O aumento da semântica é feito pela adoção de ontologias na descrição de conceitos e relacionamentos.

Referências Bibliográficas

Aumueller, D., Do, H., Massmann, S. E Rahm, E.. (2005) Schema and Ontology Matching with COMA++. Proceedings of the ACM SIGMOD International Conference on Management of Data, p. 906-908, Baltimore, Maryland.

Boyd, M.., Kittivoravitkul, S., Lazanitis, C., McBrien, P., and Rizopoulos, N. (2004) AutoMed: A BAV Data Integration System for Heterogeneous Data Sources. In Proc. of the 16th Advanced Information Systems Engineering Conference, Riga, Latvia

Broekstra, J., Ehrig, M., Haase, P., and et al. (2003) A Metadata Model for Semantics-Based Peer-to-Peer Systems. In: Proc. of the WWW'03 Workshop on Semantics in Peer-to-Peer and Grid Computing.

Castor (2007). The Castor Project. Disponível em: <http://www.castor.org>. Acesso em: dezembro de 2007.

Coma++ (2007) Coma++. Disponível em: http://dbs.uni-leipzig.de/Research/coma.html. Acesso em dezembro/2007

eMule (2007) eMule. Disponível em: http://www.emule-project.net. Acesso em dezembro/2007

Fensel, D. (2001) Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce. Springer.

Freenet (2007) Freenet. Disponível em: http://freenet.sourceforge.net

Garcia, L. G. (2005) Uma ferramenta para engenharia reversa de esquemas XML em esquemas conceituais no ambiente BInXS. Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis.

Gnutella (2007). Gnutella. Disponível em: http://www.gnutella.com. Acesso em dezembro/2007

Kazza (2007) Kazza. Disponível em: http://www.kazaa.com. Acesso em dezembro/2007

Mello, M. R. (2007) OntoGen: Uma Ferramenta para Integração de Esquemas XML. Trabalho de Conclusão de Curso (Ciência da computação). Instituto de Informática, UFRGS, Porto Alegre. Disponível em: http://www.inf.ufrgs.br/~deise/TCC.pdf

Mello, R. dos S. (2002) Uma abordagem bottom-up para integração semântica de esquemas XML. Tese (doutorado em ciência da computação). Programa de Pós-Graduação em Computação – Instituto de Informática, UFRGS, Porto Alegre.

Mena, E., and Illarramendi, A. (2001). Ontology-based query processing for global information systems. Springer.

Noll, R.; Saccol, D. B. ; Edelweiss, N.. (2007) Uma proposta para análise de similaridade entre documentos XML e ontologias em OWL. In: I Sessão de Pôsteres (em conjunto com o Simpósio Brasileiro de Banco de Dados (SBBD)), João Pessoa

Saccol, D.B., Edelweiss, N. and Galante, R.M. (2007). Detecting, Managing and Querying Replicas and Versions in a Peer-to-Peer Environment. In: 1st IEEE TCSC Doctoral Symposium (7th IEEE Intl. Symp. on Cluster Computing and the Grid), Rio de Janeiro.

Staab. S. and Studer, R. (2004) Handbook on Ontologies (International Handbooks on Information Systems). Springer.

Xu, L. and Embley, D. (2003) Using schema mapping to facilitate data integration.

Yang, B. and Garcia-Molina, H. (2002) Efficient Search in Peer-to-Peer Networks. In: Proc. of the Intl. Conf. on Distributed Computing Systems, Vienna, Austria.