Universidade de São Paulo – USP

OntoEditor
Um editor web de ontologias

Leandro Henrique Mendonça de Oliveira

NILC-TR-07-03

Março, 2007

Série de Relatórios do Núcleo Interinstitucional de Lingüística Computacional
NILC - ICMC-USP, Caixa Postal 668, 13560-970 São Carlos, SP, Brasil

1 - Introdução 1
2 – O Ambiente de Trabalho do OntoEditor 3
2.1 – Descrição das Funções dos Menus 5
2.1.1 – Menu: Nova Ontologia 5
2.1.2 – Menu: Abrir Ontologia 7
2.1.3 – Menu: Excluir Ontologia 11
2.1.4 – Menu: Visualizar Ontologia 11
2.1.4.1 – A Visualização *Folder-tree* 12
2.1.4.2 – A Visualização Hiperbólica 14
2.1.4.3 – A Visualização em Grafos 16
2.2 – Recursos e Padrões Utilizados 18
2.3 – O Modelo de Dados 19
2.4 – Conjunto de Arquivos 20
3 – Conclusão e Melhoramentos 23
Referências Bibliográficas 24
Anexo 1 - Modelo Físico de Dados 26

1 - Introdução

Diversas linguagens, técnicas e ferramentas têm sido propostas para a organização do conhecimento e sua visualização na Web. Na descrição e organização do conhecimento, alguns padrões já começam a ser estabelecidos, tais como XML e RDF (W3C, 2005, Ceri, et. al. 2000). Tais propostas tentam estabelecer e representar o conhecimento, bem como os o estudo de mecanismos de análise e reconhecimento de suas relações. As interfaces dos sistemas de visualização também são objetos de estudo e procuram dar soluções viáveis para o reconhecimento e manipulação de informações que, por sua vez, podem ser utilizadas por uma comunidade leiga e diversificada, que possui diferentes níveis de educação, capacidades e necessidades.

Nesse contexto, algumas ontologias estão sendo desenvolvidas para estabelecer um consenso sobre o significado de conceitos e termos específicos de diversos domínios do conhecimento (Venâncio, et. a. 2003). Uma ontologia que relaciona hierarquicamente os conceitos e objetos relativos a um determinado domínio permite localizar um objeto (deste domínio) numa base de conhecimento, de forma que um usuário pode se referir a este de maneira inequívoca.

Dessa maneira, o uso da ontologia evita a ambigüidade, pois uma visão específica da ontologia ajuda o usuário a entender o significado dos conceitos e o contexto em que se inserem. Cada um dos conceitos, por sua vez, deve compor uma estrutura bem definida, possibilitando a especificação de diversos tipos de relações entre eles, que ajudem a descrever formalmente o domínio específico. Neste sentido, uma ontologia é precisamente o tipo de recurso que permite uma concreta realização conceitual de um domínio e deve ser feita de forma explícita detalhada.

Ainda nesse contexto, outro aspecto fundamental de uso de ontologias se refere à possibilidade de que um determinado conceito possa se localizar em mais de uma classe de conceitos. Isto é possível graças aos mecanismos de herança, já que uma das vantagens das ontologias de conceitos é a possibilidade de herança múltipla, por meio da qual um conceito filho pode se atribuir a mais de um pai e aparecer, deste modo, em lugares diferentes da ontologia. É por esse motivo que atualmente há uma tendência em utilizar estrutura de grafos direcionados, ao invés de árvores, para representar ontologias. Isso é justificado porque o uso de grafos permite a ocorrência indeterminada das relações polivalentes e complexas entre os conceitos, concebida pelos aspectos polissêmicos presentes no domínio especializado.

Atualmente, existem vários editores de ontologias, que além de sua edição permitem sua visualização. Alguns bons exemplos desses editores são: *Protégé 2001* (Noy, et. al. 2002), o *OntoEdit* citado por (Staab and Maedche, 2001), o *Inxight StarTree* (Inxight, 2005), o *TreeBolic Generator*[1] e o *HyperEditor*[2]. Uma característica comum entre eles é que são aplicações *stand-alone* que executam fora do ambiente Web, impedindo sua utilização por vários usuários ao mesmo tempo. Entretanto, apesar dessa característica, os editores *StarTree, TreeBolic Generator* e *HyperEditor* foram

[1] Disponível para download em: http://treebolic.sourceforge.net/en/home.htm

[2] Atuamente, o HyperEditor é desenvolvido e distribuído pela Embrapa Informática Agropecuária (www.cnptia.embrapa.br).

construídos em Java e portanto sua migração para o ambiente web, na forma de Applets, pode ser possível.

Os dois primeiros editores citados acima são ferramentas que implementam a visualização de ontologias na forma *folder-tree*. Nesse tipo de visualização, quando um nó é selecionado, seu conteúdo é apresentado à direita da seleção, e assim sucessivamente até o último nível da hierarquia (comumente chamado de nó folha). Nesse caso, quando a estrutura da ontologia possui vários níveis de abstração, a visualização fica prejudicada.

De outra forma, os editores *StarTree, TreeBolic Generator* e *HyperEditor*, apresentam a visualização da ontologia na forma de árvore hiperbólica (*Hyperbolic Tree*) (Lamping, et. al. 1995). Segundo Freitas et al. (2001), esta visualização representa hierarquias através de um layout radial disposto em um plano hiperbólico mapeado para um plano de duas dimensões (2D). Além disso, apresenta aspectos de construção - como o efeito *fisheye* (Furnas, 1986) - aliados a mecanismo simples de navegação pela indicação de um nó de interesse, que é exibido no centro da representação em detalhe, cujo contexto é mantido pela exibição do restante da estrutura da ontologia com nós diminuindo de tamanho até serem suprimidos na borda do círculo radial. A abordagem do plano hiperbólico pode manipular uma estrutura em árvore usando o conceito de foco e contexto[3]. Ou seja, o plano hiperbólico permite o usuário navegar através dos nós e visualizar a relação da porção visível do plano com a estrutura inteira sobre uma única tela (Hao et. al., 1999). Com isso, amplia-se o grau de cognição humana sobre determinado assunto.

O efeito *fisheye* (também chamado de "olho-de-peixe", ou visão detalhada) fornece um esquema que, em geral, é suficiente para lidar com a navegação e orientação de grandes redes de informação. Na movimentação dessa interface, os nós da ontologia aumentam e diminuem de tamanho, saindo e entrando em foco, demonstrando grande flexibilidade e agilidade na tela. Inicialmente, na visualização da ontologia, o nó raiz tem o foco principal enquanto que os outros nós apresentam-se deformados ou semi-ocultos em detrimento da parte focalizada, podendo, entretanto, ser expandidos quando o usuário arrastar os nós com o mouse ou, ainda, por meio de pesquisa direta pelo nó.

Na visualização hiperbólica, o efeito *fisheye* é obtido através do cálculo do tamanho dos nós folhas e da distância do centro do nó focalizado. Os nós mais distantes dos focos são menos detalhados que os nós mais próximos. Daí a razão da visão detalhada. A expansão e poda dos nós na estrutura são operações que mantém sempre uma "sub-visualização", reduzindo no usuário a sensação de perda de contexto.

Assim, as árvores hiperbólicas são uma representação dinâmica da estrutura hierárquica de uma ontologia, e representa uma maneira eficiente de exibir árvores complexas com exatidão. Um dos conceitos que permeiam esse tipo de representação é o nível variável de detalhes, cujos objetos em que o usuário está interessado são exibidos com todos os detalhes centralizados na tela, enquanto os demais objetos são colocados ao lado, com detalhes reduzidos.

De acordo com estudo realizado pelo Xerox Palo Alto Research Center, citado por Inxight (2005), a técnica de árvore hiperbólica para navegação e visualização de coleções de informação hierárquica muito grandes mostrou ser 62% melhor para navegação que o padrão *folder-tree*.

---

[3] Também chamado de *focus+context* em inglês.

Além destes editores e visualizadores, ainda podemos citar aqueles cuja visualização da ontologia seguem a modelo de grafos. Um bom exemplo é o conjunto de bibliotecas *TouchGraph*[4], construídas na linguagem Java que podem, dependendo da implementação, ser adaptado para o ambiente Web como *Applets*. A principal vantagem em utilizar o *TouchGraph* está no fato que a estrutura da ontologia neste programa é vista como um grafo e não como árvore, característica de fundamental importância para representar o real comportamento de algumas ontologias, quando se considera a possibilidade polissêmica dos conceitos representados.

Nesse contexto, o presente trabalho visa apresentar o desenvolvimento do OntoEditor, uma ferramenta para edição de ontologias via Internet, que implementam as visualizações arbórea (*folder-tree*), hiperbólica (*Hyperbolic Tree*) e de grafos (usando o *TouchGraph*). Construído sobre várias tecnologias de desenvolvimento web, como por exemplo, Navegador da Árvore Hiperbólica[5], Applets Java, JavaScript e PHP em conjunto com banco de dados MySQL, as principais contribuições deste trabalho são: 1) um visualizador de ontologias cuja interface combina a flexibilidade das visualizações em árvore hiperbólica e grafos, 2) a capacidade de converter a estrutura de uma ontologia no formato texto em estruturas de visualização a partir de uma operação de *upload* e 3) a flexibilidade de estar acessível via Internet para o público geral e especializado, possibilitando qualquer usuário criar e visualizar suas ontologias a qualquer tempo.

A seção seguinte deste relatório especifica todas as características técnicas do editor de ontologia. O ambiente de trabalho, as funções implementadas, os formatos de arquivos, as tecnologias utilizadas e o modelo de dados adotados serão apresentados. Na seção 3 as conclusões e os melhoramentos esperados nas futuras versões desta ferramenta serão delineados.

2 – O Ambiente de Trabalho do OntoEditor

Todas as funcionalidades do ambiente de trabalho do OntoEditor baseiam-se no paradigma de navegação da web. Os elementos de formulário e de navegabilidade são utilizados para formar um ambiente de trabalho já conhecido pelos usuários da Internet.

Conforme pode ser visto na Figura 1, a área de trabalho do OntoEditor é dividida em três partes (representado por três *frames*). A primeira parte, superior, identificada pelo numero 1 na Figura 1, representa o menu de opções do Editor. A ação inerente a cada opção deste menu é aberta na segunda parte, identificada pelo número 2 na Figura 1, cuja área esta disponível para o usuário executar suas tarefas: de abertura, seleção, exclusão e criação de ontologias. A terceira e última parte, identificada pelo número 3, somente é utilizada nas opções *Abrir Ontologia*, *Excluir Ontologia* e *Visualizar Ontologia*. Essa parte é usada para mostrar a estrutura *folder-tree* (como uma estrutura de pastas) de uma ontologia, servindo também para seleção dos nós da ontologia nas atividades de edição.

[4] http://www.touchgraph.com/

[5] O Navegador da Árvore Hiperbólica utilizado neste relatório foi desenvolvido pela Embrapa Informática Agropecuária e esta disponível na rede AgroLivre (http://www.agrolivre.gov.br/) para download.

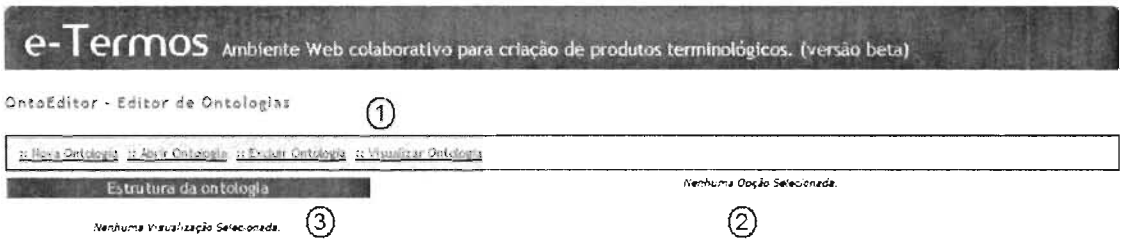

Figura 1 – Área de trabalho do OntoEditor

Uma das grandes vantagens do OntoEditor é permitir a criação de ontologias a partir de arquivos texto (com extensão TXT) tabulados, que representam a estrutura da ontologia. A título de exemplificação, utilizaremos uma "ontologia exemplo" para acompanhar todas as fases de criação e edição de uma ontologia.

O nome dessa "ontologia exemplo" será EXEMPLO DE ONTOLOGIA e sua estrutura em formato texto está representada na Figura 2. Observe que cada linha do arquivo representa um nó da ontologia, e que a quantidade de tabulações em cada linha representa o nível hierárquico de cada nó na estrutura. Dessa maneira, o nó de nome "ITEM1" é o nível 0, o nó "ITEM11" é o nível 1 e assim sucessivamente. É importante salientar que não há limites para a quantidade e a profundidade dos nós que podem se criados.

```
EXEMPLO DE ONTOLOGIA
ITEM1
        ITEM11
        ITEM12
                ITEM121
                ITEM122
                        ITEM1221
                ITEM123
        ITEM13
                ITEM131
        ITEM14
ITEM2
        ITEM21
        ITEM22
                ITEM221
                        ITEM2211
ITEM3
        ITEM22
                ITEM11
ITEM4
        ITEM41
```

Figura 2 – Estrutura em formato do texto de uma ontologia (EXEMPLO DE ONTOLOGIA)

Ao longo das seções a seguir todas as tarefas que atualmente podem ser executadas em cada menu serão descritas.

2.1 – Descrição das Funções dos Menus

Como visto na Figura 1, existem 4 diferentes opções no menu, sendo elas: *Nova Ontologia, Abrir Ontologia, Excluir Ontologia* e *Visualizar Ontologia*. Cada opção do menu possui telas específicas com campos de preenchimentos ou de seleção, os quais o usuário deve preencher. A subseções desta seção tratam especificamente de cada opção do menu.

2.1.1 – Menu: Nova Ontologia

A opção do menu "Nova Ontologia" possibilita o usuário criar uma nova ontologia. Na versão atual do OntoEditor o usuário pode criar uma ontologia por meio da submissão de arquivo (operação de upload) em formato texto que contém a estrutura da ontologia. Entretanto, nas futuras versões o usuário também poderá montar a estrutura de maneira on-line (interativa) e por meio de uma área de texto (item c, abaixo). Além dessa opção, ainda está previsto a entrada da estrutura da ontologia em formato OWL (*Web Ontology Language*). A OWL é uma linguagem semântica de anotação, com o objetivo de publicar e partilhar ontologias na Internet.

Ao clicar em "Nova Ontologia" uma seqüência de campos de preenchimento são oferecidos ao usuário. Conforme pode ser visto na Figura 3, esses campos compreendem em:

a) **Nome da Ontologia**: é nome que a ontologia a ser criada deve receber. Usando o EXEMPLO DE ONTOLOGIA, esse campo poderia ter a mensagem: "*Exemplo de Ontologia*";
b) **Descrição da Ontologia**: compreende uma breve descrição da ontologia. Informações como a área especifica a qual é pertencente e o motivo da criação podem ser inseridos. No EXEMPLO DE ONTOLOGIA, essa descrição poderia ser: "*Essa ontologia explica a criação de uma nova ontologia*";
c) **Estrutura da Ontologia**: representa uma área de texto onde a estrutura da ontologia pode ser editada. A edição da ontologia nessa área deve obedecer o formato de arquivo definido na Figura 2. Apesar de estar prevista, a versão atual do OntoEditor ainda não implementou essa função.
d) **do Arquivo**: essa opção permite o usuário realizar a operação de *upload* de um arquivo texto que contem a estrutura da ontologia a ser criada. Esta opção é uma alternativa ao item *c* (estrutura da ontologia), e sua grande vantagem é a liberdade que o usuário tem de poder editar seu arquivo em qualquer programa editor de texto. Da mesma maneira que a edição da ontologia na área de texto, o arquivo que contém a estrutura da ontologia deve obedecer ao formato mostrado na Figura 2.
e) **Cor do Nós**: permitir o usuário selecionar a cor dos nós na visualização da ontologia, tanto no formato hiperbólico quanto no formato de grafos.
f) **Cor do Texto:** permitir o usuário selecionar a cor do texto dos rótulos, (ou *labels* dos nós) na visualização da ontologia, tanto no formato hiperbólico quanto no formato de grafos.

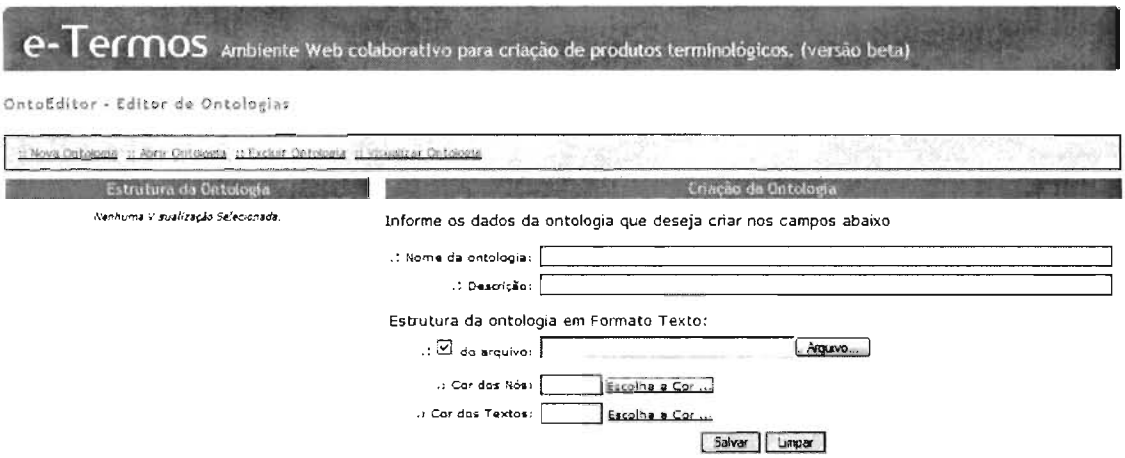

Figura 3 – Campos de preenchimento para a criação de uma nova ontologia

Instanciando a criação de uma nova ontologia no exemplo EXEMPLO DE ONTOLOGIA e uma vez preenchido os campos necessários, a criação da ontologia ocorre quando o usuário clica no botão **Salvar**. O processo de criação da ontologia compreende três fases:

Fase 1: Verificação da validade da estrutura em texto da ontologia: compreende a tarefa de verificar se o formato da estrutura do texto da ontologia está de acordo com o padrão definido (Figura 2);

Fase 2: Inclusão da ontologia no banco de dados: compreende o processo de inserir os dados da ontologia no banco de dados. Cada ontologia ganha um número identificador único, de quatro dígitos, criado de maneira aleatória;

Fase 3: Criação dos diretórios e arquivos fontes: compreende o processo de criação do diretório e dos arquivos fontes da ontologia. O nome desse diretório é igual ao número identificador único criado na Fase 2. Os arquivos fontes que são criados nessa fase correspondem àqueles que possibilitam a visualização da ontologia (conforme será descrito na Subseção 2.2.4), bem como aqueles de extensão HTML que permitem tal visualização. Os arquivos HTML a que se refere essa fase são criados a partir dos "*templates*" que estão localizados no diretório do OntoEditor no *servidor web*[6].

Uma vez criada a ontologia, o restante dos menus passam a ter suas tarefas específicas válidas, dando assim a liberdade ao usuário de selecionar qualquer opção desejada.

A Figura 4 mostra a confirmação de criação da ontologia EXEMPLO DE ONTOLOGIA.

[6] Uma descrição completa dos arquivos de codificação do OntoEditor será feita na Seção 2.5.

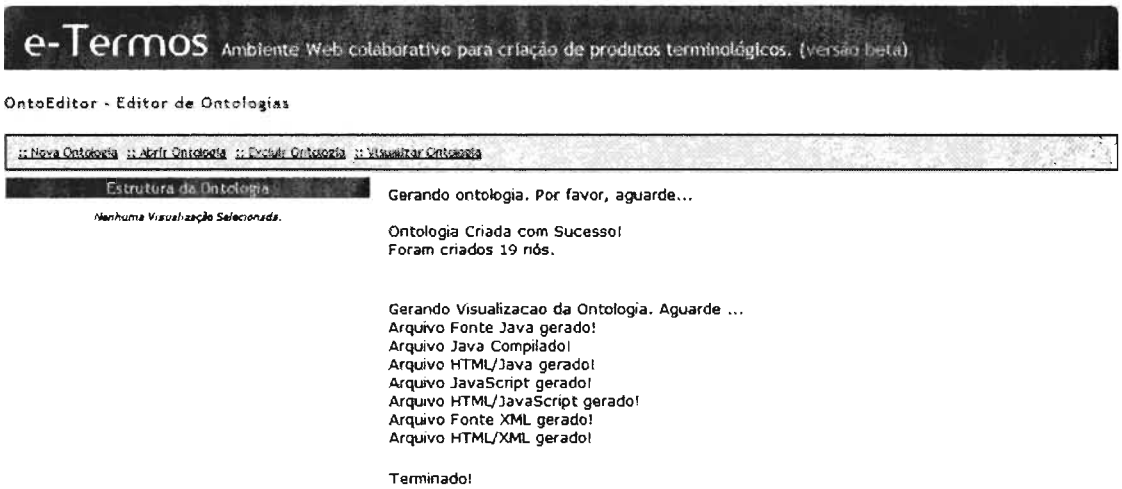

Figura 4 – Interface de confirmação de criação de uma nova ontologia

2.1.2 – Menu: Abrir Ontologia

Uma vez criada uma determinada ontologia, o usuário é capaz de abri-la para edição (alteração, inclusão e exclusão dos nós) ou para visualização *folder-tree* na parte 3 do ambiente, conforme mostrado na Figura 1.

Como pode ser visto na Figura 5, ao clicar no menu "Abrir Ontologia", é mostrado ao usuário um campo de seleção para que a ontologia desejada seja escolhida. Nesse campo de seleção, aparecem todas as ontologias que foram criadas e armazenadas no banco de dados.

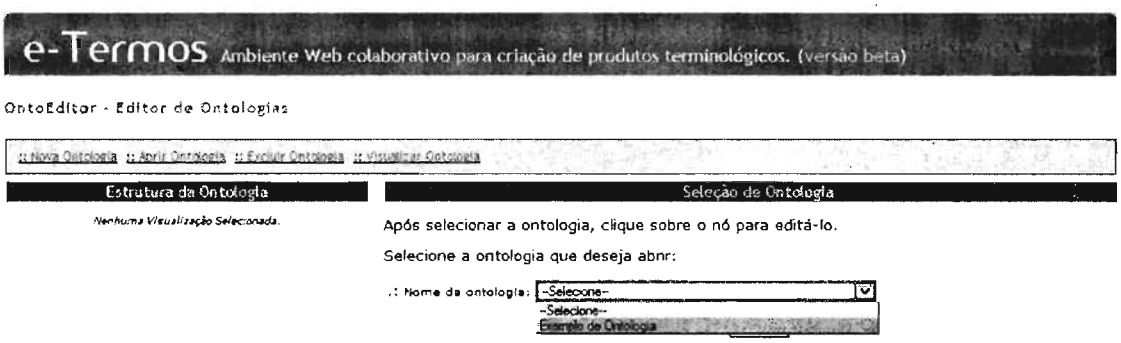

Figura 5 – Interface de seleção da ontologia do menu Abrir Ontologia

Tendo escolhido a ontologia, a visualização *folder-tree* da ontologia é automaticamente aberta na parte 3 (lado esquerdo) da área de trabalho.

A partir da visualização *folder-tree* o usuário pode navegar pelos nós da ontologia por meio das ações de "abrir" e "fechar" em cada nó clicando no sinal de (+) ou (–). A visualização do EXEMPLO DE ONTOLOGIA pode ser vista na Figura 6.

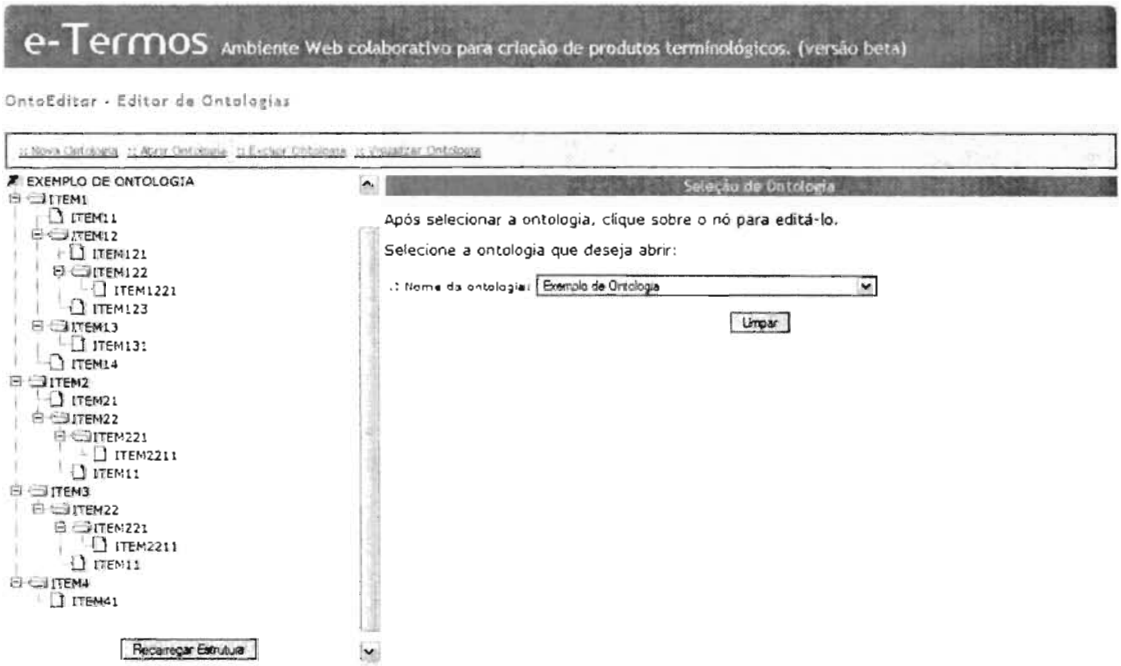

Figura 6 – Exemplo de visualização da ontologia do menu Abrir Ontologia para edição de um nó.

Usando esta visualização, o usuário pode editar um determinado nó, bastando para isso selecionar o nó desejado e clicando sobre ele. Automaticamente, quando o usuário clica sobre um nó, todas as informações sobre ele são mostradas na parte 2 da área de trabalho, juntamente com as opções: 1) *Alterar Nó*, 2) *Excluir Nó,* 3) *Incluir Novo Nó* e 4) *Criar Novo LINK* conforme pode ser visto na Figura 7. Observe nesta Figura que o nó de nome "ITEM21" foi selecionado e todos os detalhes de seu registro foram mostrados à direita. Observe que existem vários campos referentes ao nó, como a cor do texto, cor do nó, hint e URL. Esses campos referem-se às visualizações hiperbólica e de grafos e serão discutidos mais adiante na Subseção 2.2.4.

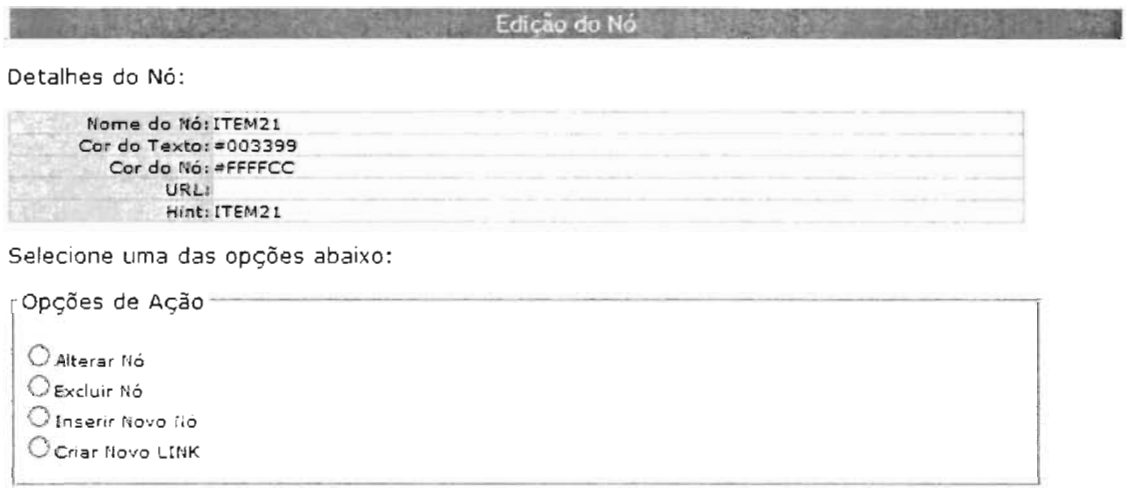

Figura 7 – Interface que mostra as opções de edição de um determinado nó na ontologia

As opções mostradas na Figura 7 permitem ao usuário editar um determinado nó selecionado, podendo alterar seus dados de registro, excluí-lo, inserir um novo nó a partir dele e criar um novo link (ligação para outro nó). As interfaces de alteração, exclusão, inclusão e criação de um novo link de um determinado nó são mostradas nas Figuras 8, 9, 10 e 11, respectivamente. Nestas Figuras, o nó "ITEM21" do EXEMPLO DE ONTOLOGIA foi utilizado para demonstrar tais ações.

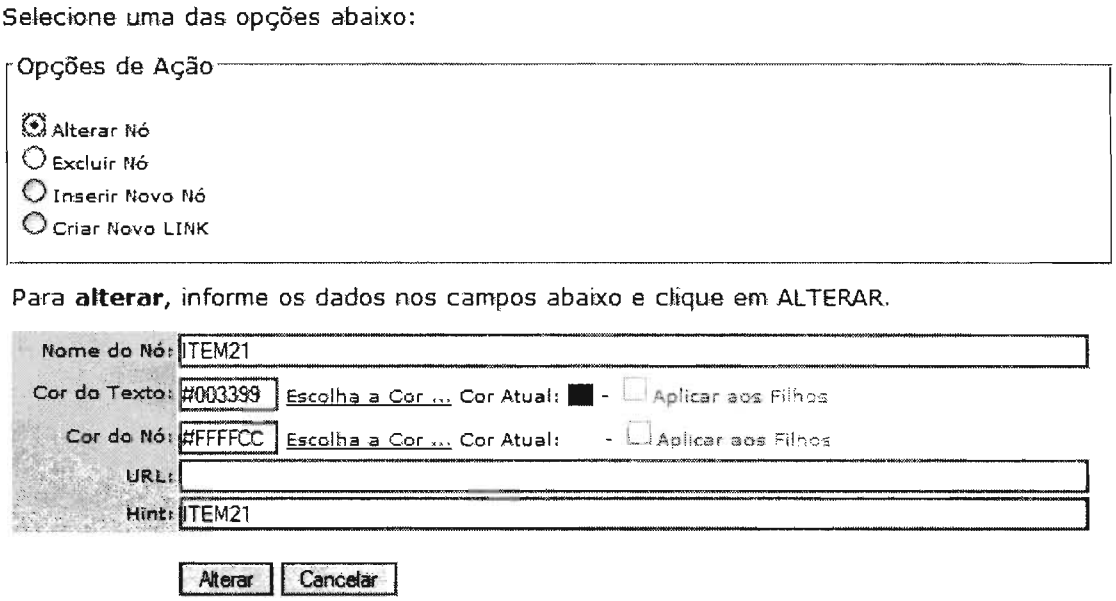

Figura 8 – Interface de alteração de um nó.

Na interface de alteração da Figura 8, o usuário pode, se desejar, editar todos os campos do nó selecionado. Inclusive a cor do nó e do texto, mostrados nas visualizações podem ser alterados. Caso o nó selecionado possua nós filhos, suas cores também pode ser alteradas recursivamente, bastando para isso selecionar a caixa *"Aplicar aos Filhos"*. Após o preenchimento dos campos, o usuário deve clicar no botão **Alterar** para efetuar a tarefa.

Ao tentar excluir um determinado nó pertencente a uma ontologia, o OntoEditor alerta o usuário sobre a existência de links (filhos e pai) do nó que deseja excluir. Isso porque, na versão atual, a exclusão de um nó que possui filhos somente pode ser efetuada se seus filhos possuírem outros nós pais. Caso contrário a exclusão será impedida. Essa condição existe porque em alguns casos, a exclusão de um determinado nó faz com que outros nós hierarquicamente subordinados fiquem órfãos, deixando a ontologia inconsistente. Como pode ser visto na Figura 9, o exemplo do nó "ITEM21" possui 1 link, somente com seu pai (ITEM 2), e nenhum filho. Nesse caso específico, o nó pode ser excluído pois satisfaz a condição expressa. Se o usuário confirmar a exclusão clicando em **Excluir**, tanto o nó "ITEM21" quanto seus links serão excluídos.

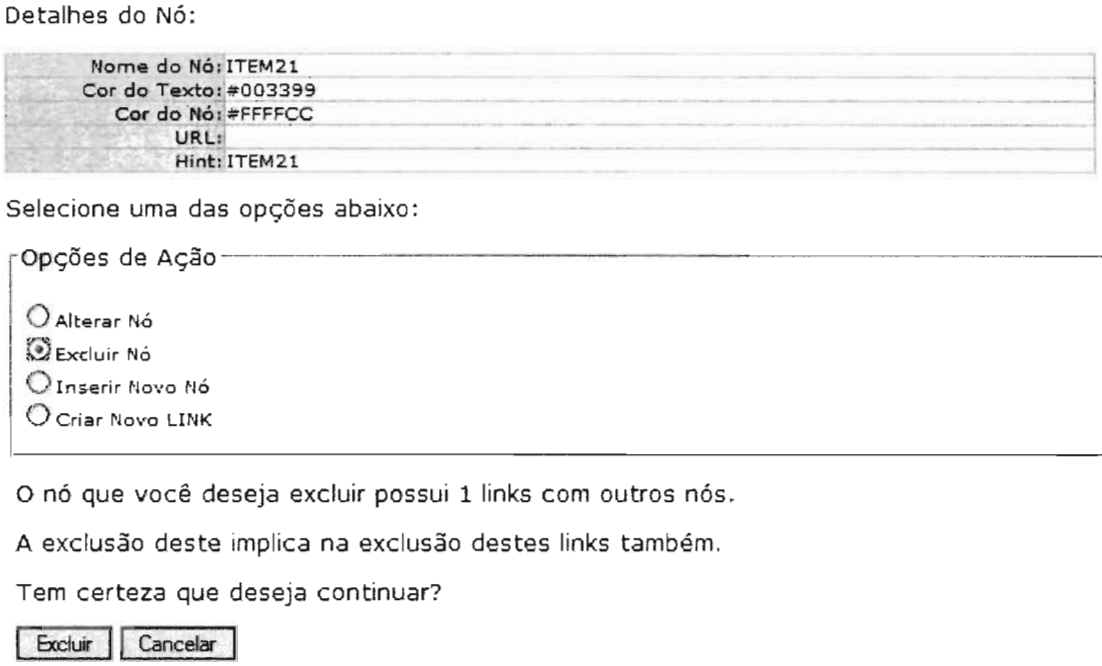

Figura 9 – Interface de exclusão de um nó.

Para a inclusão de um novo nó, conforme mostrado na Figura 10, o usuário deve preencher todos os campos no novo nó e logo em seguida clicar em **Incluir**. O novo sempre será incluído como filho do nó selecionado. Usando o exemplo da Figura 10, o nó NOVO ITEM será incluído como filho do nó "ITEM21".

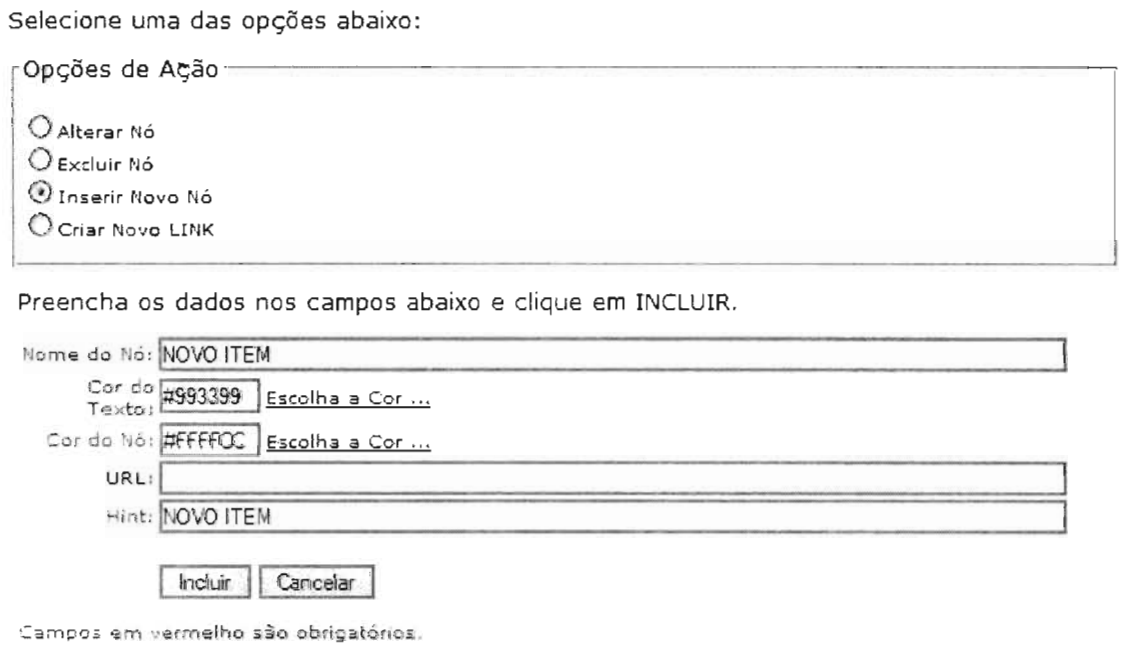

Figura 10 – Interface de inclusão de um novo nó.

Toda e qualquer edição dos nós de uma ontologia, bem como a inclusão, altera a sua estrutura, tornando-se necessário a reedição dos arquivos fontes e de visualização. Assim, com intuito de manter a consistência dos arquivos fontes e as edições do usuário, cada edição realizada na ontologia provoca a execução da Fase 3, descrita na subseção 2.1.1.

2.1.3 – Menu: Excluir Ontologia

A opção excluir ontologia é a tarefa mais simples no ambiente de trabalho do OntoEditor. Uma vez selecionada esta opção, o usuário deve selecionar a ontologia que deseja excluir e logo em seguida clicar em **Excluir**, como pode ser visto na Figura 11.

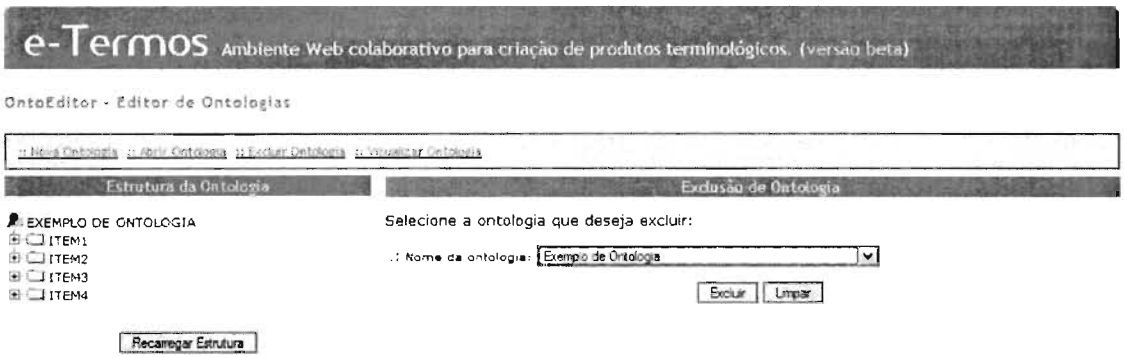

Figura 11 - Interface de seleção da ontologia do menu Excluir Ontologia

No entanto, apesar de simples, essa tarefa é muito perigosa e deve ser feita com cuidado pelo usuário, pois sua ação exclui toda a estrutura da ontologia e seus arquivos fontes, não sendo mais possível acessá-la. Devido a esse fato, um mecanismo de confirmação foi criado para evitar exclusões acidentais do usuário. Dessa maneira, após clicar no botão **Excluir**, uma confirmação, como mostrada na Figura 12, é solicitada.

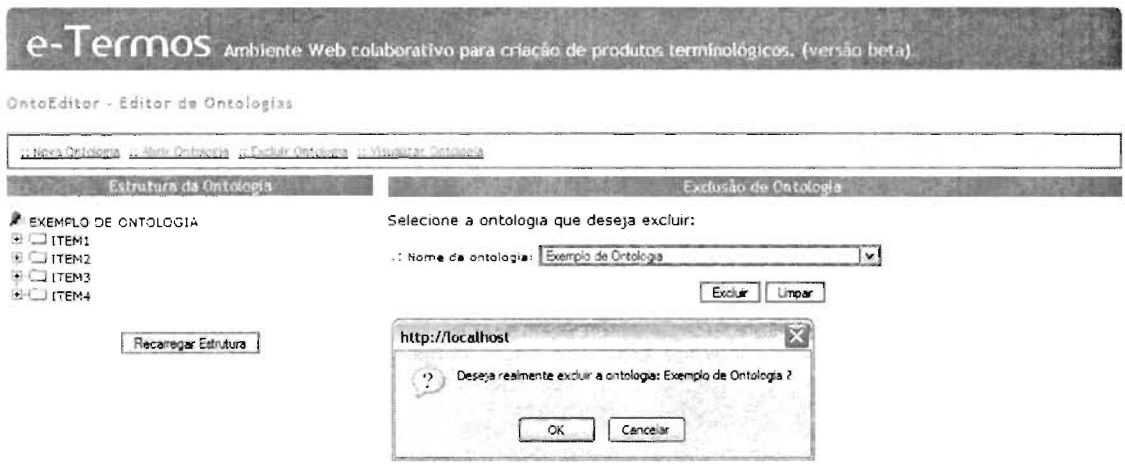

Figura 12 – Tela de confirmação de exclusão de uma determinada ontologia

2.1.4 – Menu: Visualizar Ontologia

Acessando o menu visualizar ontologia, o usuário tem a opção de visualizar uma determinada ontologia, previamente criada, de três maneiras: 1) a visualização *folder-tree* 2) visualização hiperbólica e 3) visualização em grafos.

Uma vez selecionada a ontologia, por meio do campo de seleção mostrado na Figura 13, o OntoEditor abre automaticamente a visualização *folder-tree* na parte 3 do ambiente de trabalho. Conforme dito anteriormente, essa visualização permite a navegação na estrutura da ontologia através das ações de "abrir" e "fechar" de cada nó, acionadas pelo clique nos ícones (+) ou (–). Para as visualizações hiperbólica e de grafos, o usuário deve, depois de selecionada a ontologia desejada, clicar nos botões **Visualizar Hiberbólica** e/ou **Visualizar Grafos**, respectivamente. Ao clicar em um destes botões, uma nova janela do *browser* é aberta contendo a estrutura da ontologia no formato de visualização desejado.

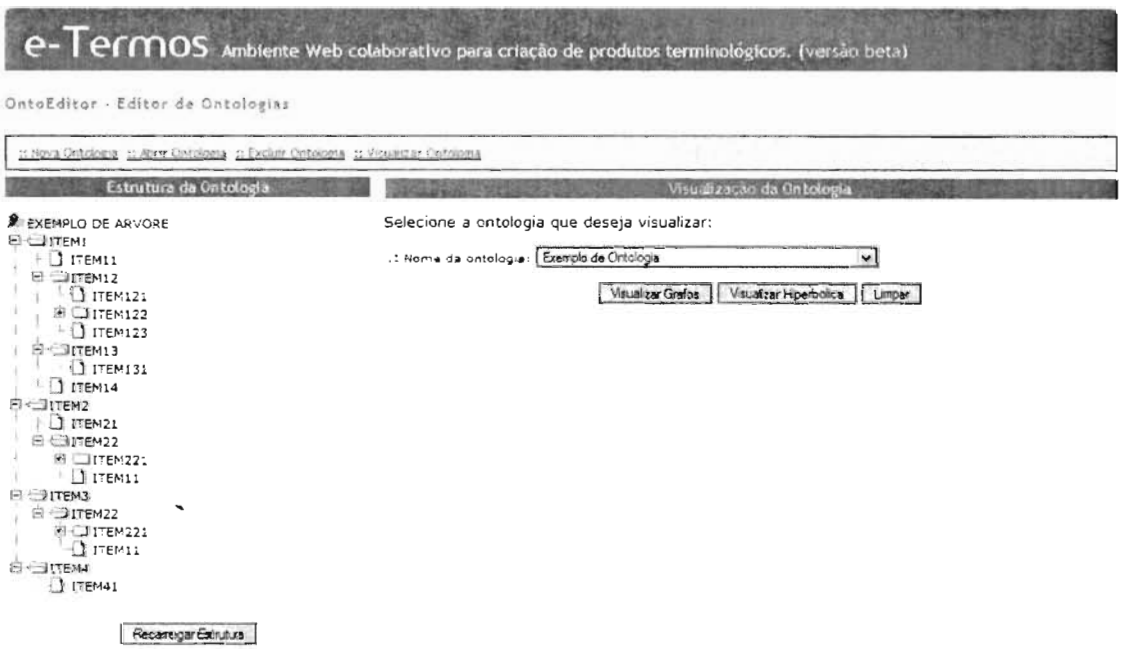

Figura 13 - Interface de seleção da ontologia do menu Visualizar Ontologia

2.1.4.1 – A Visualização *Folder-tree*

A visualização *folder-tree* representa a estrutura hierárquica da ontologia em forma de árvore que se expande da esquerda para a direita. Inicialmente, no momento que uma determinada ontologia é selecionada, sua estrutura é carregada na forma contraída, sem abrir as ramificações. Uma vez carregada a estrutura, o usuário pode abrir ou fechar um determinado ramo (ou nó) da árvore expandindo-a de maneira customizada. Veja abaixo, na Figura 14 a visualização *folder-tree* do EXEMPLO DE ONTOLOGIA.

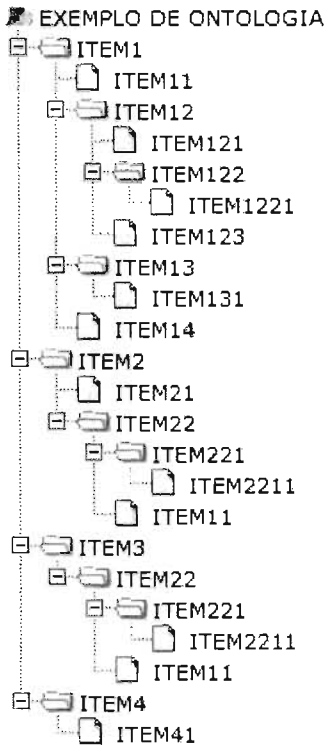

Figura 14 – Visualização *folder-tree* do EXEMPLO DE ONTOLOGIA expandida

A quantidade de ramificações e, conseqüentemente, a quantidade dos nós da árvore que representa a ontologia no OntoEditor é ilimitado, podendo existir quantos níveis forem necessários. Isso é possível porque, internamente, a estrutura de visualização é armazenada em um arquivo (um dos arquivos fontes do OntoEditor) que mantém a relação hierárquica entre os nós da ontologia.

Além disso, é importante notar que embora a visualização *folder-tree* represente uma estrutura de árvore, é possível representar uma ontologia que possui o caráter da herança de conceitos (conforme visto na Seção 1). Isso também pode ser observado na Figura 14, pelo nó "ITEM22" que possui ligações com os nós representados por "ITEM3" e "ITEM2". Note que o "ITEM22" aparece duas vezes na estrutura, duplicando inclusive a hierarquia inferior a ele (seus nós filhos).

A Figura 15 mostra o formato do arquivo fonte da visualização *folder-tree* do EXEMPLO DE ONTOLOGIA. Observe que cada linha do arquivo representa um nó da ontologia, contendo, do lado esquerdo, seu respectivo número identificador.

```
var TREE_ITEMS = [
['<!--160-->EXEMPLO DE ONTOLOGIA','',
['<!--161-->ITEM1','',
['<!--162-->ITEM11',''],
['<!--163-->ITEM12','',
['<!--164-->ITEM121',''],
['<!--165-->ITEM122','',
['<!--166-->ITEM1221',''],
],
['<!--167-->ITEM123',''],
],
['<!--168-->ITEM13','',
```

```
['<!--169-->ITEM131',''],
],
['<!--170-->ITEM14',''],
],
['<!--171-->ITEM2','',
['<!--172-->ITEM21',''],
['<!--173-->ITEM22','',
['<!--174-->ITEM221','',
['<!--175-->ITEM2211',''],
],
['<!--162-->ITEM11',''],
],
],
['<!--176-->ITEM3','',
['<!--173-->ITEM22','',
['<!--174-->ITEM221','',
['<!--175-->ITEM2211',''],
],
['<!--162-->ITEM11',''],
],
],
['<!--177-->ITEM4','',
['<!--178-->ITEM41',''],
],
],
];
```

Figura 15 - Formato do arquivo fonte da visualização *folder-tree* do EXEMPLO DE ONTOLOGIA

2.1.4.2 – A Visualização Hiperbólica

A visualização hiperbólica é uma estrutura de visualização de hierarquias baseada na técnica *focus+context* (foco+contexto). Ela destina maior espaço para o nó que está em foco e mostra o contexto (outros nós ao redor do nó focado) com tamanho progressivamente reduzido à medida que se distancia do foco. O nó focado é sempre a raiz da árvore, ou aquele que o usuário selecionar. Para implementar essa idéia, utiliza-se a geometria hiperbólica. A hierarquia é traçada em um plano hiperbólico e este é mapeado em uma elipse. Isso produz o efeito visto na Figura 16, do EXEMPLO DE ONTOLOGIA, dos nós centrais aparecem maiores e os periféricos, menores. O usuário pode alterar o foco clicando em qualquer nó. Quando isso ocorre, o nó clicado é transladado para o centro e todos os outros se rearranjam na periferia.

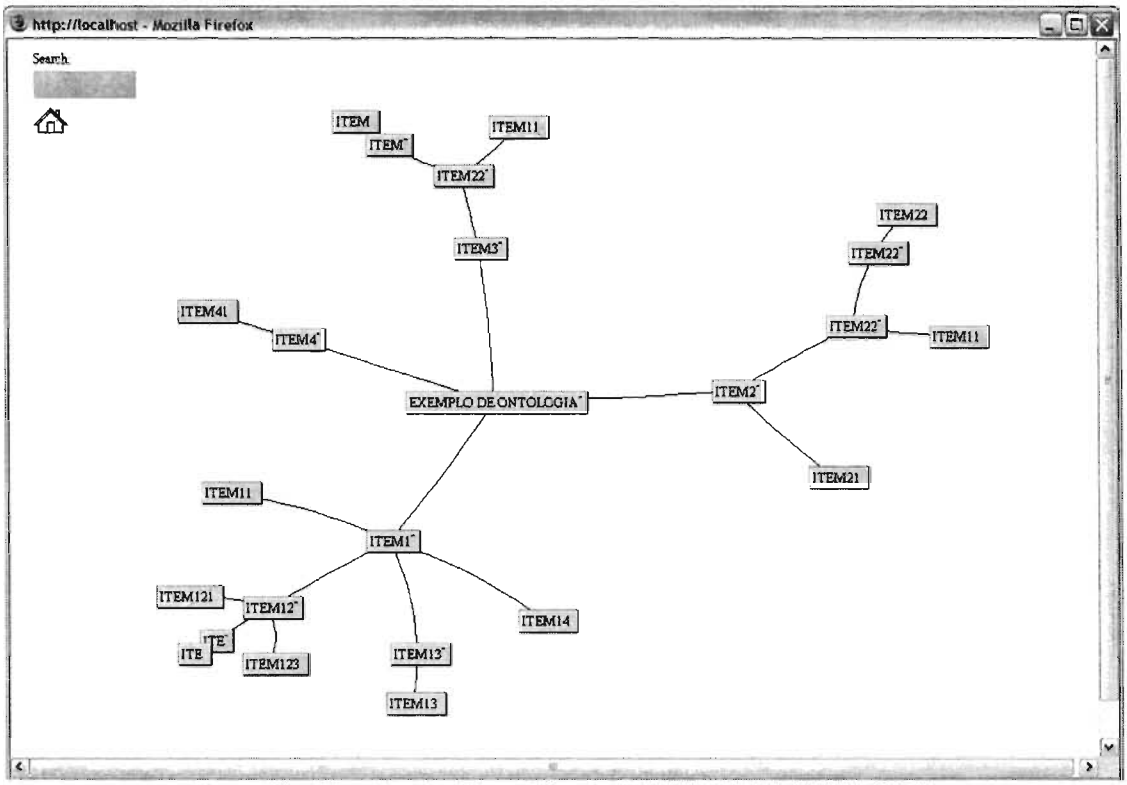

Figura 16 – Visualização Hiperbólica do EXEMPLO DE ONTOLOGIA

Apesar da Figura 16 mostrar o EXEMPLO DE ONTOLOGIA com poucos nós, a visualização hiperbólica é indicada para a visualização de grandes estruturas de ontologia, pois, mesmo com milhares de nós, é possível observar as informações dos nós no entorno do foco.

O OntoEditor utiliza um formato de arquivo compilado, com extensão HTZ, que contém a estrutura da ontologia. Cada nó pertencente à ontologia contém, em sua estrutura interna as seguintes informações: *Nome do Nó*, *URL*, que é endereço na Internet que pode acessado quando ocorre o duplo clique do mouse sobre o nó, *Hint,* que representa um texto "dica" que armazena informações adicionais do nó naquele contexto da ontologia, a *Cor do Texto* e a *Cor do Nó*.

A visualização hiperbólica também oferece um serviço de busca textual para localização de nós, que marca o caminho em vermelho desde a raiz até os nós que contêm a expressão de busca e, ainda, marca com uma bolinha vermelha os nós que contêm a expressão. O resultado da função de busca para o nó "ITEM22" pode ser visto na Figura 17.

Observe na Figura 17 que assim como na visualização *folder-tree*, também ocorre a duplicação dos nós que possui herança conceitual, como mostra a marca vermelha ate o "ITEM22" em dois caminhos diferentes na mesma estrutura da ontologia.

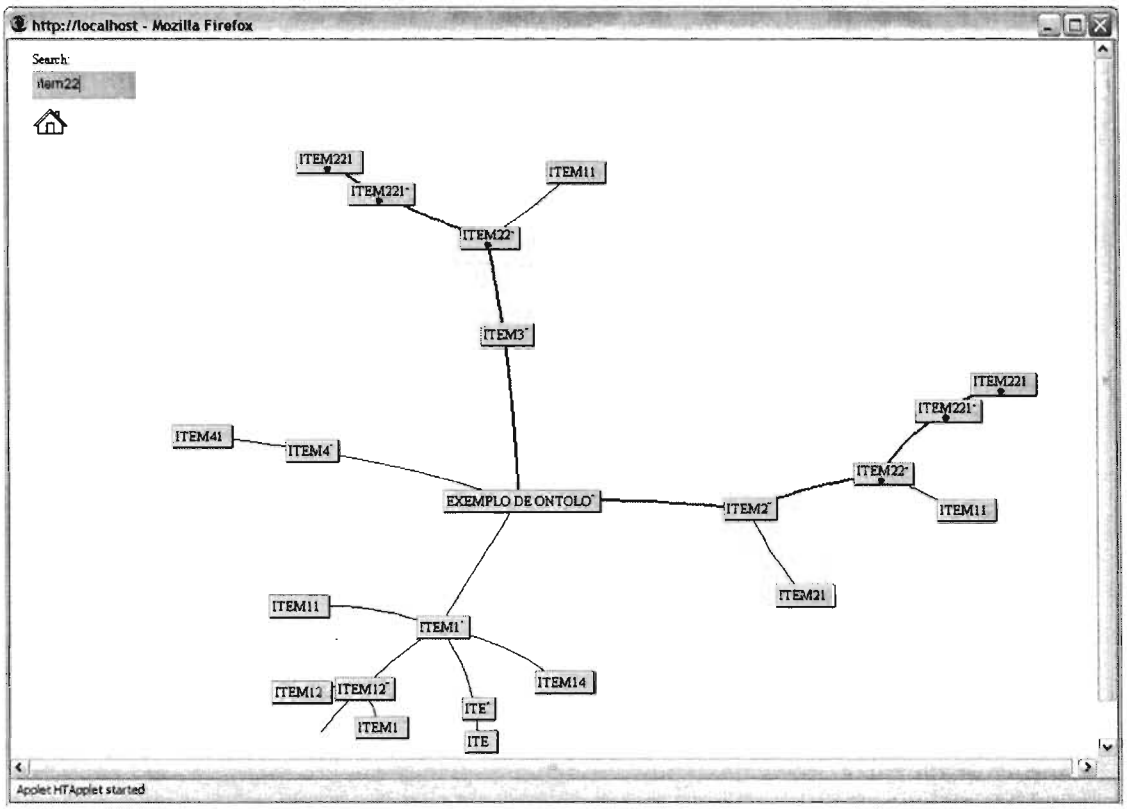

Figura 17 – Resultado da busca pelo ITEM22 na visualização hiperbólica

2.1.4.3 – A Visualização em Grafos

A visualização em grafos disponibilizada no OntoEditor implementa o conjunto de bibliotecas denominado de *TouchGraph*. Este conjunto de bibliotecas forma uma API (*Application Programming Interface*) que permite a manipulação, edição e visualização de grafos ou redes de informação dispostas e hierarquicamente relacionadas. As ontologias que podem se representadas dessa maneira, são apresentadas em grafos interativos e direcionados possibilitando uma grande variedade de manipulações visuais e permitindo os usuários navegar em grandes estruturas ontológicas e explorar diferentes características das ontologias na própria visualização.

Atualmente existem várias implementações do *TouchGraph*, como a *TouchGraph GoogleBrowser*[7] e *TouchGraph WikiBrowser*[8], que respectivamente, disponibilizam uma forma visual de navegabilidade do *Google* e da *Wikipédia*. Tais implementações fornecem, em sua visualização, uma série de opções e ferramentas adaptadas para cada caso e podem ser vistas como uma especialização da API genérica do *TouchGraph*. Na versão atual do OntoEditor foi utilizada a implementação *TouchGraph LinkBrowser*[9], uma versão mais "leve" do *TouchGraph*.

No OntoEditor vários motivos nos levaram a utilizar o *LinkBrowser* a saber:

[7] http://www.touchgraph.com/TGGoogleBrowser.html
[8] http://www.usemod.com/cgi-bin/mb.pl?TouchGraphWikiBrowser
[9] http://sourceforge.net/project/showfiles.php?group_id=30469&package_id=22440

1) **Tamanho do Código Compilado**: é pequeno e leve o que é de muita relevância para as aplicações Web e execução de *Applets* que diminuem o tempo de carga na execução no lado cliente;

2) **Hints em Texto (TXT) e HTML**: possibilidade de disponibilizar os *hints* de cada nó em texto puro ou no formato HTML (tal característica pode ser observada na Figura 18 logo adiante);

3) **Estrutura da Ontologia em XML**: leitura e armazenamento da estrutura da ontologia em formato XML, permitindo o intercâmbio e exportação da ontologia para outros sistemas que utilizam o mesmo formato;

4) **Facilidade de Manipulação**: através dos controles do *mouse* é possível expandir e contrair as estruturas da ontologia, além das possibilidades de visualização da mesma que permitem as operações de *Zoom*, *Rotate* (rotação da estrutura da ontologia) e *Locality* (que permite definir o nível de expansão da ontologia).

Do mesmo modo que a visualização hiperbólica, para cada nó da ontologia nesta visualização está previsto cinco campos: *Nome Nó, Cor do Texto, Cor do Nó, URL* e *Hint*. Além destas facilidades inerentes ao código do *TouchGraph*, novas funcionalidades podem ser adicionadas, como por exemplo a ferramenta de busca (*search*) de nós, edição de *link* e nós na própria interface de visualização.

A Figura 18 mostra o EXEMPLO DE ONTOLOGIA na visualização em grafos. Como pode ser visto, a principal vantagem desta visualização é a possibilidade de ver a ontologia em formato de grafos direcionados, que representa a estrutura formal na ontologia. Observe que neste caso o "ITEM22" aparece uma única vez na estrutura e mantém os mesmos *links* com os nós "ITEM2" e "ITEM3", visto nas outras visualizações. Note também que os controles da visualização podem ser utilizados através das barras de rolagem superior e lateral direita, bem como na lista de operações (*Zoom*, *Rotate* e *Locality*) dispostos no controle *Select* localizado no canto superior esquerdo.

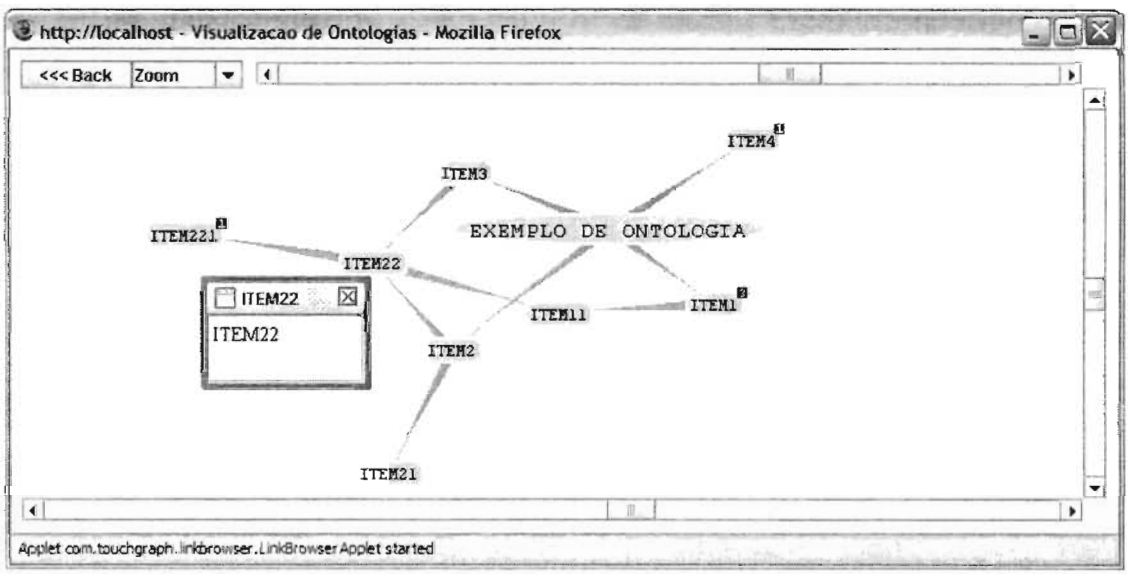

Figura 18 – Visualização em Grafos do EXEMPLO DE ONTOLOGIA

Ainda na Figura 18, é possível notar a presença de pequenos quadradinhos vermelhos (nós ITEM1, ITEM221, ITEM4), preenchidos com um número inteiro. Estes números indicam a quantidade de nós "filhos" existentes "abaixo" de um determinado nó. Com um simples clique do *mouse* sobre estes nós, expande-se a visualização da ontologia. Além disso, como pode ser visto na Figura 18, cada nó possui um *hint* cujo conteúdo pode ser tanto em texto puro ou originário de um arquivo HTML. Adicionalmente, a edição da ontologia, e conseqüentemente de cada nó de sua estrutura, pode ser realizada dando um duplo clique do *mouse* sobre o nó, na qual é disparada uma ação que possibilita a edição nó, semelhante à edição ocorrida na visualização *folder-tree*. A Figura 19 mostra a tela da edição do nó "ITEM1", após o duplo clique.

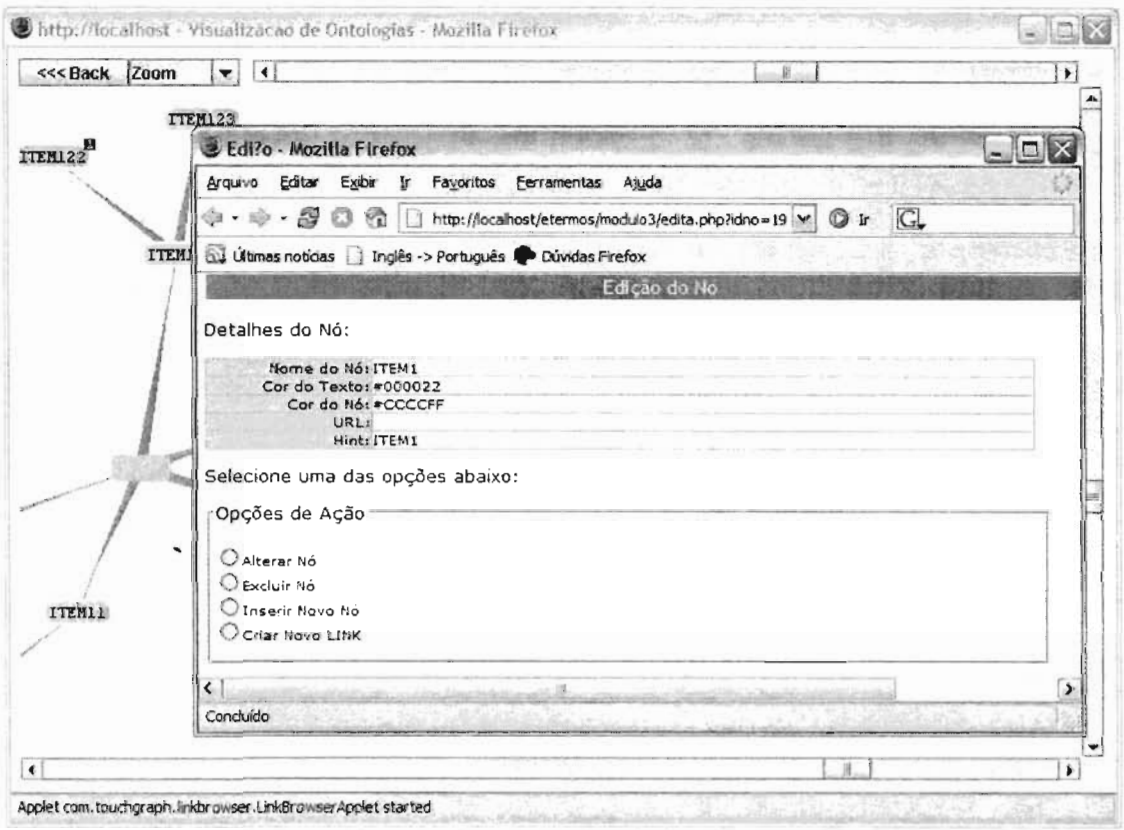

Figura 19 – Tela de edição do nó ITEM1 após o duplo clique do *mouse* sobre este nó

2.2 – Recursos e Padrões Utilizados

Devido à característica de aplicação Web do OntoEditor, vários recursos de desenvolvimento deste tipo de aplicação foram utilizados. Tais recursos são: 1) banco de dados relacional (MySQL), 2) servidor web (Apache), 3) linguagem de hipertexto (HTML), 4) padrão de estilos em HTML (CSS – *Cascade Style Sheet*) e 4) três linguagens de programação, sendo uma delas voltada para lado *servidor* (PHP) e duas voltadas para o lado cliente (JavaScript e Applets Java).

Todas esses recursos foram integrados de forma transparente ao usuário, visto que não há uma delimitação clara das fronteiras entre uma e outra. Por outro lado, para o

um programador, técnico ou analista, essa fronteira é visível, tornando a manutenção mais do OntoEditor mais fácil.

Entretanto, com intuito de dividir os recursos entre cliente e servidor, respeitando o modo de como a Internet trabalha, podemos dividir e agrupar esses recursos da seguinte forma:

a) **Grupo de recursos do lado *Servidor***: Servidor Web (Apache), Banco de Dados (MySQL) e Linguagem PHP;
b) **Grupo de recursos do lado *Cliente***: páginas HTML em conjunto com o CSS, JavaScript e Applets Java.

2.3 – O Modelo de Dados

Além de armazenar os dados referentes às ontologias, o banco de dados (MySQL) utilizado na implementação do OntoEditor suporta e gerencia as tarefas de edição, conforme descritas na Seção 2, que os usuários executam.

Apenas três tabelas compõem o modelo de dados: a tabela *ontologia*, a tabela *no* e a tabela *relacaonos*. A tabela *ontologia* armazena as informações genéricas de uma ontologia qualquer, como seu nome, a descrição e seu identificador (representado pelo atributo *id*). A tabela *no* armazena a estrutura de determinada ontologia e representa todos os nós que a pertencem. Todos os dados que identificam e caracterizam um determinado nó pertencente a uma ontologia são armazenados nessa tabela. A tabela *relacaonos* representa as relações (ou links), entre os nós da ontologia. O Anexo 1 apresenta o modelo físico, e as Tabela 1, 2 e 3 abaixo mostram o modelo conceitual dessas tabelas.

Tabela 1 – Modelo Conceitual da Tabela *Ontologia*

Tabela Ontologia	
Atributo	**Tipo**
id	numerico(5)
nome	varchar(255)
descricao	varchar(255)

Tabela 2 – Modelo Conceitual da Tabela *No*

Tabela No	
Atributo	**Tipo**
id	numerico (11)
ontologia_id	numerico (5)
nome	varchar(255)
cortexto	varchar(8)
corno	varchar(8)
url	varchar(255)
hint	varchar(255)

Tabela 3 – Modelo Conceitual da Tabela *Relacaonos*

Tabela Relacaonos	
Atributo	**Tipo**
id	numerico (11)
id_origem	numerico (11)

id_destino	numerico (11)
id_ontologia	numerico (5)

Conforme pode ser observado no modelo conceitual, as tabelas *ontologia* e *no* possuem um relacionamento com granularidade *1* para *n* por meio do atributo *id* da tabela *ontologia*, cujo nome é alterado para *ontologia_id* na tabela *no*. Este relacionamento representa o mapeamento lógico da realidade onde uma determinada ontologia possui um ou vários nós. Além disso, esse relacionamento é importante pois permite separar logicamente as informações referentes à ontologia das informações referentes aos nós, evitando assim a duplicação dos atributos *nome* e *descrição*.

Por sua vez, a tabela *relacaonos* possui dois relacionamentos com a tabelo *no*. Cada tupla desta tabela guarda a identificação única do link (atributo *id*), o identificador do nó de origem do link (atributo *id_origem*, que representa o primeiro relacionamento com a tabela *no*), o identificador do nó de destino do link (atributo *id_destino*, que representa o segundo relacionamento com a tabela *no*) e a ontologia este link pertence (atributo *id_ontologia*). Apresentando granularidade *1* pra *n*, esses relacionamentos permitem a implementação da relação de dependência hierárquica que um determinado nó possui com o outro, podendo representar a possibilidade de um determinado nó possuir *n* filhos, independentemente da quantidade e dentro de uma mesma tabela.

O Modelo Entidade-Relacionamento (MER) das tabelas *ontologia*, *no* e *relacaonos* pode ser vista na Figura 20 abaixo.

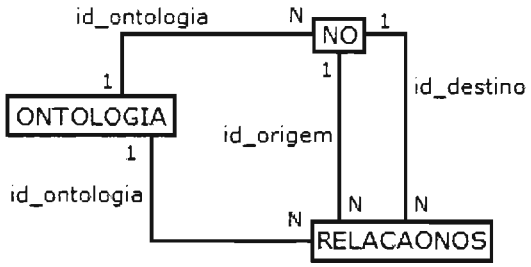

Figura 20 – Modelo de Entidade-Relacionamento

As tabelas implementadas no MySQL são do tipo *InnoDB*. Este tipo de tabela suporta as restrições de integridade dos relacionamentos definidos nas tarefas de inserção (*insert*), exclusão (*delete*) e atualização (*update*). As restrições de integridade do banco de dados servem para assegurar que as regras dos relacionamentos estabelecidos entre as tabelas sejam mantidas durante as manipulações dos dados.

2.4 – Conjunto de Arquivos

Considerando todos os recursos utilizados na implementação do OntoEditor, e, conseqüentemente, suas diferentes extensões, a Tabela 3 mostra os arquivos que fazem parte de sua codificação, o seu tipo e sua função.

Tabela 3 – Lista de Arquivos do Código-Fonte do OntoEditor

Nome do Arquivo	Tipo de Arquivo	Conteúdo/Função
abre.php	PHP	Arquivo que abre uma determinada ontologia para edição.

ajuda.htm	HTML	Arquivo que contém o conteúdo de ajuda da ferramenta.
altera.php	PHP	Arquivo que executa as alterações no(s) nó(s) de uma determinada ontologia.
AppletGrafoHTM_template.txt	Texto	Arquivo de template que representa o arquivo de visualização em grafos do AppetJava.
cores.js	JavaScript	Arquivo fonte em JavaScript que cria a tabela de cores a serem aplicados na ontologia.
cria.htm	HTML	Arquivo HTML que mostra as opções de criação de ontologias.
cria.php	PHP	Arquivo que cria uma determinada ontologia. A criação compreende não inclusão no banco de dados e criação de arquivos para visualização.
crialink.php	PHP	Arquivo que cria os links entre os nós da ontologia.
edita.php	PHP	Arquivo que oferece as opções de edição de um determinado nó de uma ontologia.
exclui.php	PHP	Arquivo que oferece a seleção de uma determinada ontologia para exclusão.
excluino.php	PHP	Arquivo que faz a exclusão de determinado nó de uma ontologia.
fazcriacao.php	PHP	Arquivo que cria uma ontologia e suas visualizações.
fazedicao.php	PHP	Arquivo que executa as edições realizadas nas ontologias.
fazexclusao.php	PHP	Arquivo que exclui um determinado nó de uma ontologia.
funções.php	PHP	Arquivo que contem todas as funções que são utilizadas pelos arquivos PHP. Funciona como uma "library".
incluino.php	PHP	Arquivo que executa a inclusão de um novo nó em uma ontologia. As informações do Novo Nó devem ser informadas através do arquivo edita.php.
index.htm	HTML	Arquivo HTML principal (main) da ferramenta OntoEditor.
inicializa.inc	ARQUIVO DE INICIALIZAÇÃO DE VARIÁVEIS	Arquivo que contém as variáveis e inicializações que são usadas pelos arquivos PHP.
impmenusuptedit.php	PHP	Arquivo que imprime o menu de opções superior da interface.
javahtm_template.txt	TEXTO	Arquivo de template que representa o arquivo de visualização do AppetJava.
jshtm_template.txt	TEXTO	Arquivo de template que representa o arquivo de visualização do JavaScript.
linear.htm	HTML	Arquivo HTML que apresenta a visualização linear (JavaScript) vazio.
menu.htm	HTML	Arquivo HTML que apresenta o Menu da aplicação do OntoEditor.
nenhuma.htm	HTML	Arquivo HTML que é mostrado quando nenhuma opção do menu (menu.htm) é selecionado.
tabcores.html	HTML	Arquivo que mostra as cores geradas pelo arquivo 'cores.js'.
vizualiza.php	PHP	Arquivo que oferece a seleção de uma

		determinada ontologia para visualização.
hypertreeN.jar	JavaApplet	Arquivo fonte do AppletJava que permite a visualização hiperbólica.
hypertreeE.jar	JavaApplet	Arquivo fonte do AppletJava que permite a visualização hiperbólica.
BrowserLaucher.jar, nanoxml-2.1.1.jar TGLinkBrowser.jar	JavaApplet	Arquivos fonte do AppletJava que permite a visualização em grafos da ontologia.
tree.js	JavaScript	Arquivo fonte do JavaScript que permite a visualização *folder-tree*.
tree_tpl.js	JavaScript	Arquivo fonte do JavaScript que permite a visualização *folder-tree*.
XMLGrafo_template.txt	Texto	Arquivo de template que representa a estrutura em XML da ontologia em grafo.
*.htz	HTZ (compilado Java)	Arquivo compilado que contem a estrutura de visualização hiperbólica de uma determinada ontologia. O "*" representa o numero identificador da ontologia.
*.js	JavaScript	Arquivo fonte em JavaScript que contem a estrutura de navegação *folder-tree*. O "*" representa o numero identificador da ontologia.
*.html	HTML	Arquivo HTML que acopla os códigos das visualizações *folder-tree*, hiperbólica e em grafos para apresentação no browser. O "*" representa o numero identificador da ontologia.

Para um melhor entendimento do funcionamento da execução do OntoEditor, a Figura 21 mostra o Diagrama Chama/Chamado do conjunto de arquivos que fazem parte da codificação. O arquivo que inicia a execução do OntoEditor é o *index.htm.*

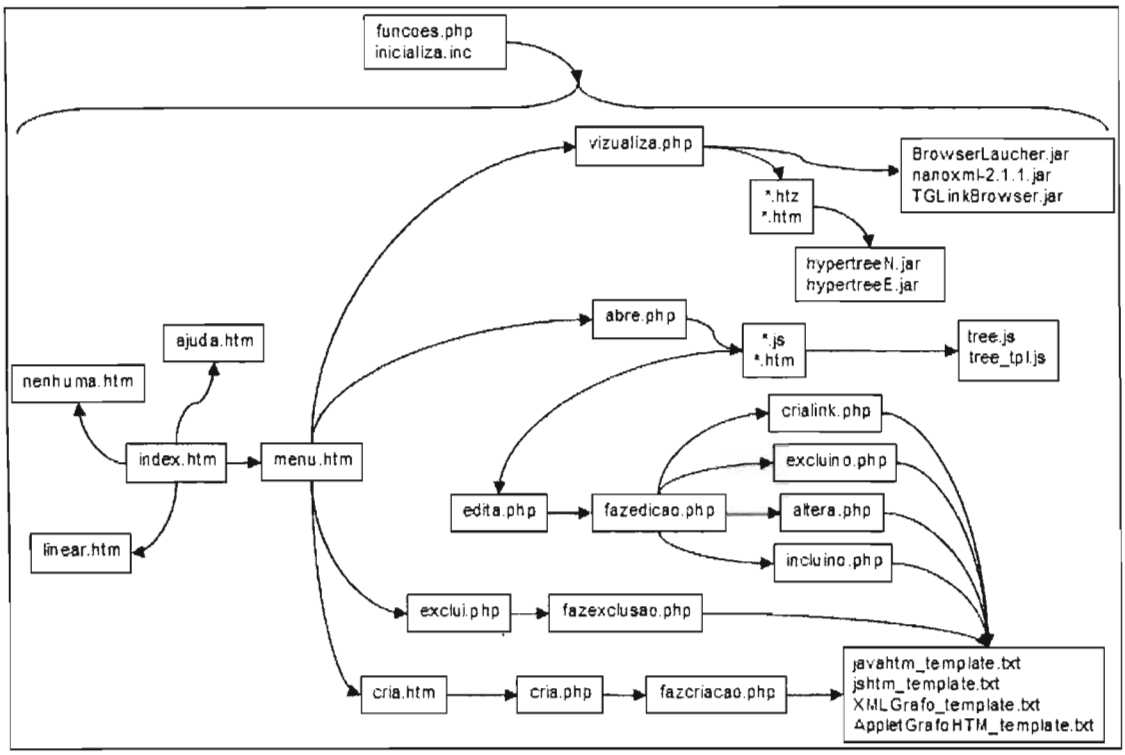

Figura 21 – Diagrama Chama/Chamado dos arquivos de codificação

3 – Conclusão e Melhoramentos

Desde que a principal motivação do OntoEditor foi a criação de um editor de ontologias para o ambiente Web, sua implementação exigiu um estudo da viabilidade e uso conjunto de diversas tecnologias de desenvolvimento Web, que juntos pudessem trabalhar de maneira transparente e útil para o usuário. Contudo, o ajuste fino da junção dessas tecnologias não foi realizado, de maneira que a tarefa específica da edição de ontologias ficou limitada. Isso porque sabemos que ainda existem diversas atividades pertencentes ao processo de edição de ontologias que o OntoEditor não contempla, e que, outras ferramentas e/ou software de mesmo propósito, como: *Protégé, Inxight StarTree, HyperEditor e TreeBolic Generator* já fornecem.

A única diferença, porém, é que tais ferramentas trabalham fora do ambiente Web, sendo na maioria das vezes programas *stand-alone* que não suportam a criação compartilhada e paralela de ontologias via Internet. Entretanto, uma possível solução, de caminho mais curto, seria a adaptação de algum software já existente, e de comprovada qualidade, para o ambiente Web. Alguns exemplos como o *TreeBolic Generator* e *HyperEditor* são construídos sobre a plataforma Java e podem, dependendo da implementação, ser adaptados para o ambiente Web como Applets.

Por outro lado, como primeira versão, salientamos que as principais vantagens do OntoEditor são: 1) a possibilidade de visualizar a estrutura de uma ontologia no formato de árvore hiperbólica e de grafos, 2) a capacidade de converter a estrutura de uma ontologia a partir de um formato texto (TXT) em estruturas de visualização (*folder-tree*, hiperbólica e de grafos) a partir de uma operação de *upload*, e 3) a flexibilidade de estar acessível via Internet para o público geral e especializado, possibilitando qualquer

usuário criar e visualizar suas ontologias a qualquer tempo. Contudo, ocorre a nítida impressão que o OntoEditor deverá passar por uma fase de maturação, até alcançar os níveis profissionais e de atendimento às necessidades dos usuários.

Assim, entendemos que apesar das tarefas de edição estarem aquém do que poderão vir a ser, acreditamos que o OntoEditor é uma valiosa ferramenta Web de visualização de ontologias. Dessa forma, reconhecidas as limitações e vantagens, sugerimos os seguintes melhoramentos para as próximas versões:

1) Implementar o controle de versões de uma mesma ontologia, de maneira que o acompanhamento das versões e/ou estágios de desenvolvimento de uma ontologia possam ser identificados;
2) Implementar um item de *Check Box* ao lado de cada nó da estrutura visualização *folder-tree* para o caso de verificação e/ou validação de uma determinada ontologia;
3) Incluir as opções de edição de um nó (Alterar, Excluir e Incluir) de maneira "*on-line*" nas próprias visualizações, como função do botão direito do *mouse*;
4) Implementar um *parser* que possa ler como entrada de ontologia um arquivo OWL, bem como exportar as ontologias neste formato.

Referências Bibliográficas

(W3C, 2005) The World Wide Web Consortium (W3C). Extensible Markup Language (XML). http://www.w3.org/XML.

(Ceri, et. al. 2000) S. Ceri, P. Fraternali, S. Paraboschi. XML: Current Developments and Future Challenges for the Database Community. Proc. EDBT Conf., LNCS 1777, Springer- Verlag, 2000.

(Venancio, et. al. 2003) Venâncio, L. R., Fileto, R. Medeiros, C. B.) Aplicando Ontologias de Objetos Geográficos para Facilitar Navegação em GIS. GeoInfo, 2003, Disponível em: http://www.geoinfo.info/geoinfo2003/papers/geoinfo2003-45.pdf

(Lamping, et. al. 1995) J. Lamping, R. Rao, e P. Pirolli. A Focus+Context Techinique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. Proc. ACM SIGCHI Conf. on Human Factor in Computing System, 1995.

(Inxight, 2005) INXIGHT SOFTWARE INCORPORATED. Inxight Star Tree. Disponível em: http://www.inxight.com/products/oem/star_tree/

(Freitas, et. al. 2001) FREITAS, C. M. dal S.; CHUBACHI, O. M.; LUZZARDI, P. R. G.; CAVA, R. A. Introdução à visualização de informações. RITA, v. 8, n. 2, p. 1-16, 2001. Disponível em: http://www.inf.ufrgs.br/cg/publications/carla/Freitas-RITA2001.pdf

(Hao, et. al. 1999) HAO, M. C.; HSU, M.; DAYAL, U.; KRUG, A. Visual mining large web-based hyperbolic space using hidden links. Palo Alto: HP Laboratories-Software Technology Laboratory, 1999. 9 p. (HPL 1999-20). Disponível em: http://www.hpl.hp.com/techreports/1999/HPL-1999-20.pdf

(Staab and Maedche, 2001) Steefen Staab Alexander Maedche. Knowledge Portals – Ontologies at Work, Al Magazine, Summer 2001, pgs. de 63-75.

(Noy, et. al. 2002) N.F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fregerson, e M. A. Musen. Creating semantic Web contents with protégé-2000. IEEE Intelligent Systems, 16 (2):60-71, 2002.

(Furnas, 1986) G.Furnas. Generalized fisheye views. In: Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, 1986, pp 16-23.

Anexo 1 – Modelo Físico de Dados

Tabela *ontologia*

Field	Type	Null	Key	Default	Extra
id	smallint(5) unsigned		PRI	NULL	auto_increment
nome	varchar(255)				
descricao	varchar(255)				

Tabela *no*

Field	Type	Null	Key	Default	Extra
id	int(11)		PRI	NULL	auto_increment
ontologia_id	smallint(5) unsigned		MUL	0	
nome	varchar(255)				
cortexto	varchar(8)				
corno	varchar(8)				
url	varchar(255)	YES		NULL	
hint	varchar(255)	YES		NULL	

Tabela *relacaonos*

Field	Type	Null	Key	Default	Extra
id	int(11)		PRI	NULL	auto_increment
id_origem	int(11)		MUL	0	
id_destino	int(11)	YES	MUL	NULL	
id_ontologia	smallint(5) unsigned		MUL	0	