4
Estudo de Caso

Este capítulo apresenta um estudo de caso, descrevendo de forma detalhada as etapas, procedimentos e ferramentas utilizadas para a geração da federação de ontologias.

4.1 Contextualização do problema

Uma empresa possui duas bases de dados BD_1 e BD_2 que precisam ser integradas para se obter informações sobre os produtos criados e oferecidos para consumo aos clientes da empresa.

A empresa em questão comercializa produtos de telefonia básica, ou seja, é uma empresa de Telecom. Ela possui um departamento de "*Marketing*" que é responsável pela criação de produtos de telefonia a serem oferecidos aos clientes. Existem vários tipos de funcionários neste departamento. Uma equipe alocada neste departamento realiza estudos para poder conhecer melhor o público e poder saber da viabilidade de se lançar um novo produto tendo em vista as necessidades do cliente.

Cada produto ao ser criado recebe um código que o identifica de maneira única e é direcionado a um público específico (residencial, pequenas empresas ou corporativo). Produtos equivalentes ou complementares podem ser agrupados em um único pacote para ser oferecido ao cliente. Um produto pode ou não pertencer a um pacote.

Pode-se saber nesta base de dados a quais produtos cada cliente adere e mais particularmente medir o seu comportamento diante de cada produto, como por exemplo, qual é o horário em que ele mais usa o produto e de que forma. Fica evidente que ao se saber qual produto de telefonia o cliente usa e como usa, pode-se ter acesso a dados confidenciais que configuram o seu "tráfego".

O tráfego de cada cliente, assim como o mapeamento do uso dos produtos adquiridos por ele não pode ser acessado por qualquer usuário da base de dados. Desta forma é necessário definir mecanismos de proteção para acesso aos dados, ou seja, um controle de acesso.

Um usuário comum desta base de dados é dito como um usuário *público*. Este tipo de usuário não tem uma visão completa da base de dados, ele não pode enxergar alguns relacionamentos em que fique configurado o tráfego do cliente.

Todos os usuários têm permissão somente de leitura destes dados e somente o administrador de dados tem permissão de escrita. O administrador é o usuário responsável para dar acesso aos outros usuários. Na hierarquia de acesso ele é quem está no topo. Ele pode dar ou tirar a permissão de acesso de cada usuário e decide quais objetos do banco de dados cada um pode ver.

O processo que envolve a criação e geração de um produto está representado no modelo Telefonia que está associado à base de dados BD_1 conforme mostra a Figura 3.

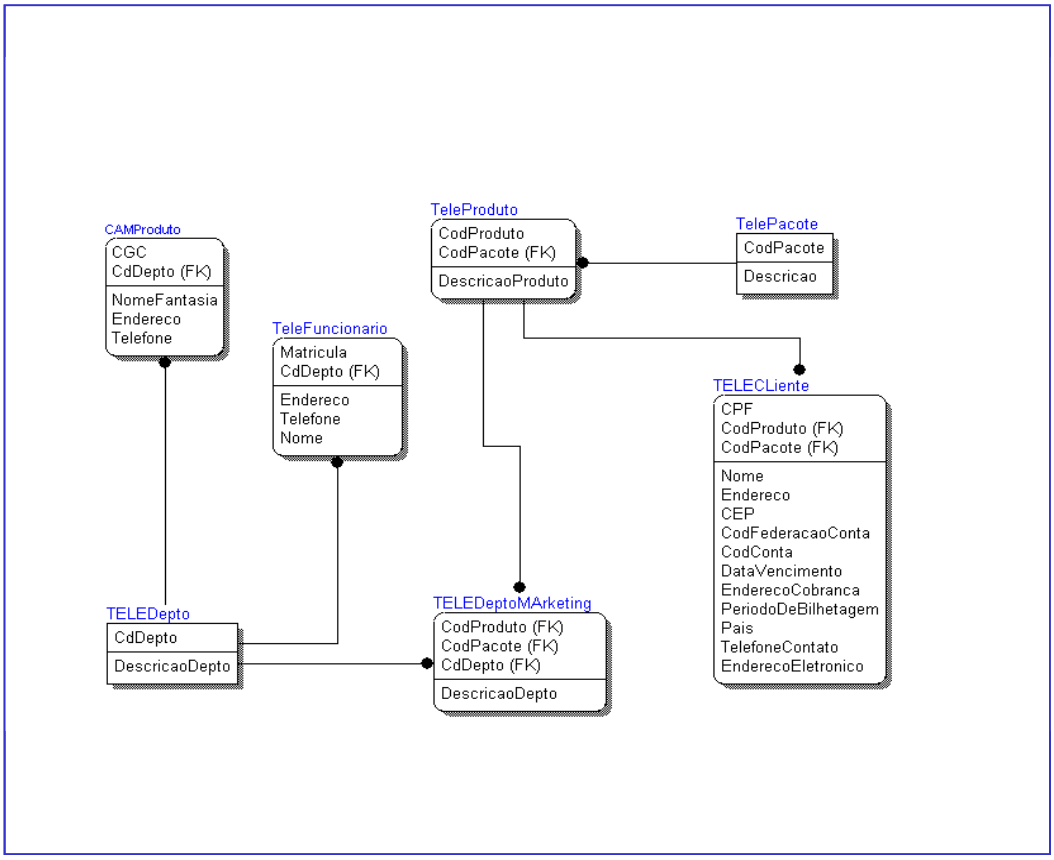


Figura 3: Modelo de dados Telefonia que representa a base de dados BD1.

O modelo Campanha mostrado na Figura 4 representa todo o processo de campanha que envolve o lançamento de um produto.

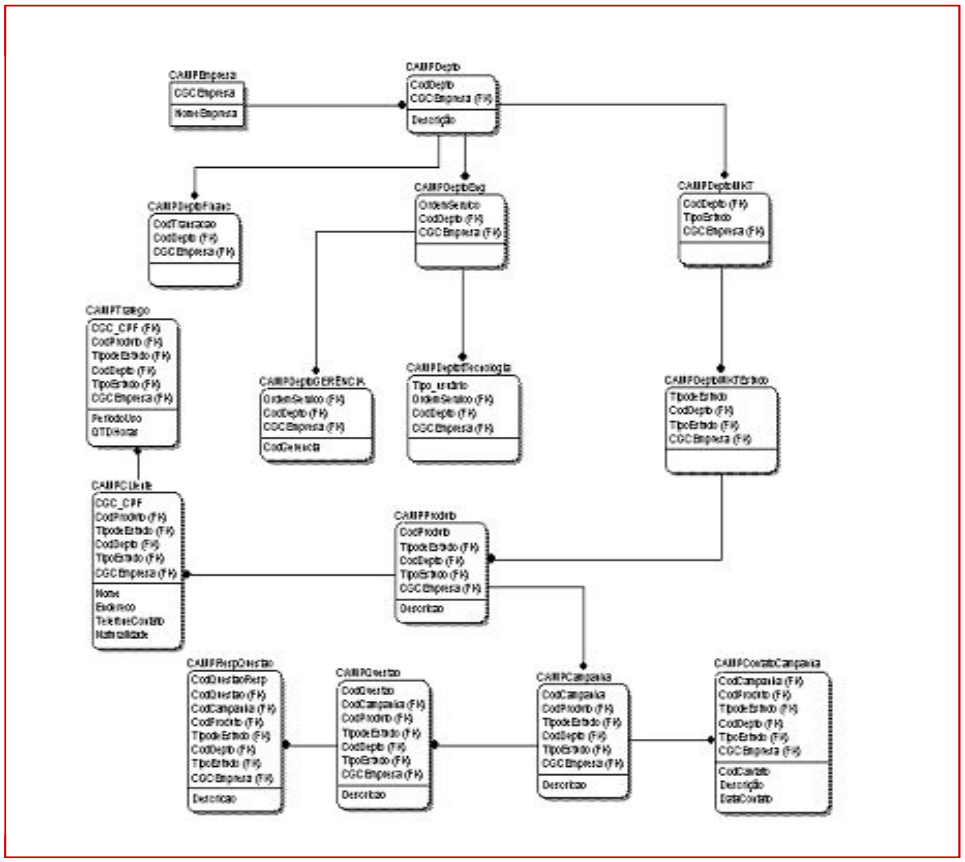


Figura 4: Modelo de dados Campanha que representa a base de dados DB2.

Um produto é criado pela equipe de estudo e lançado no mercado através de uma campanha promocional. Os clientes potenciais para adquirir o novo produto são contatados pelos funcionários. Para que a campanha seja padronizada é gerada uma "questão" (um script de venda) que deve ser seguido por todo funcionário que estiver envolvido na tarefa de venda do produto. A resposta a esta "questão" é anotada e armazenada para estudos posteriores. Também é armazenado o código da pessoa que faz o contato. Fica registrado o cliente que aderiu a este novo produto e desta forma é possível saber como ele usa este produto assim como todos os outros que ele possui configurando o seu "tráfego".

Da mesma forma que o modelo anterior os dados que configuram o tráfego de cliente são confidenciais. Logo, da mesma forma, vai ser preciso definir restrições de acesso dos usuários a esta base de dados. Como no modelo anterior

(Telefonia) o usuário administrador tem plenos poderes (leitura e gravação) sobre as bases de dados e é ele quem define o acesso de cada usuário.

4.2 Da federação de bancos de dados para a federação de ontologias

Na arquitetura de federação de dados para integração de fontes distribuídas heterogêneas, os esquemas locais são os esquemas nativos existentes nas fontes de dados. No nosso estudo de caso partimos de duas bases de dados que chamamos de BD1 - a base de dados de Telefonia, onde está registrado o processo de criação de um produto e seu relacionamento com o cliente e a base de dados BD2 a base de dados de Campanha que envolve o lançamento de um novo produto e o controle da campanha do qual este produto participa. Estas bases de dados são independentes, mas tem informações complementares e em alguns casos sobrepostas. A empresa em questão precisa gerar informação integrada dos dados que envolvem o processo de criação e campanha de um novo produto lançado pelo departamento de marketing, da adesão dos clientes a este produto e do consumo gerado por eles. Estas fontes de dados precisam ser integradas, ou seja, precisam compartilhar informação entre si. Esta integração é necessária para que se tenha uma visão mais completa do perfil do cliente da empresa de Telecom frente ao consumo de seus produtos para que as campanhas de venda de produtos sejam direcionadas a estes clientes de forma mais eficiente.

De acordo com a arquitetura e terminologia adotadas neste trabalho para a construção da federação de bancos de dados (Sheth & Larson, 1990) cada uma destas fontes de dados local é uma **fonte de dados componente** da federação de bancos de dados e, no nosso estudo de caso são bancos de dados que têm seus esquemas locais que representam modelos conceituais relacionais, ou seja, todos os esquemas locais componentes da federação serão representados em modelos de dados relacionais.

O **esquema componente** de cada base é então gerado a partir da tradução de cada esquema local para o modelo de dados canônico adotado pela federação de dados que aqui no nosso estudo de caso é também representado por um modelo de dados relacional. A decisão de escolher o Modelo Relacional para a federação de dados está baseada em facilidades de implementação de projeto. Lembramos o

fato de que nosso exemplo está fortemente influenciado por um caso real de projeto de integração de dados para o departamento de Marketing de uma empresa de Telecom que adota o Modelo Relacional como base.

Segundo (Sheth & Larson, 1990), cada **esquema de exportação** representa o subconjunto do esquema componente que será compartilhado na federação. Cada banco de dados componente não precisa disponibilizar para a federação de bancos de dados e seus usuários a sua coleção completa de dados, e nem todos os dados das fontes podem ser disponibilizados (esbarra-se aqui no problema da privacidade dos dados). Podemos citar como exemplo, dentro desta nossa empresa, uma base de dados que um determinado departamento poderia liberar para estudo e prospecção de cliente em que se pudesse saber o consumo de produtos (poderia ser minutos utilizados no período que compreendesse o horário comercial, configurando o tráfego do cliente que por ser uma empresa é dito corporativo). Este mesmo departamento não poderia tornar disponíveis os mesmos dados de consumo de produtos de um cliente residencial, pessoa física, uma vez que seu tráfego poderia dar acesso a confidencialidade de suas ligações telefônicas, (seu sigilo telefônico). Desta forma, os dados relativos ao tráfego de cliente residencial não estariam disponíveis para qualquer usuário do sistema integrador. O esquema de exportação dará, portanto uma visão homogênea, no que se refere aos esquemas locais do BD1 e BD2. Para o sistema integrador é como se as fontes de dados possuíssem o mesmo modelo de dados permitindo a utilização de uma mesma linguagem de consulta.

Segundo (Silvestre, 2005), os esquemas de exportação exercem um papel fundamental no momento da re-escrita de consultas, ou seja, quando o sistema integrador transforma a consulta global do usuário, submetida ao sistema integrador, em sub consultas, escritas de acordo com os termos (atributos e *datasets*) das fontes de dados. Além disso, é a partir da integração dos esquemas de exportação que é gerado o esquema sobre o qual os usuários efetuarão as consultas, o **Esquema Global ou Federado**.

Não é nossa intenção mostrar o esquema global da federação de bancos de dados. Vamos implementar a arquitetura de federação de ontologias e promover desta forma a integração dos dados. As heterogeneidades semânticas provenientes da tentativa de integração serão resolvidas através do uso de ontologias. A partir das bases de dados que formam os sistemas componentes, vamos criar as

ontologias que representam estas bases, ou seja, estes modelos servirão de base para a criação das ontologias. Em seguida será feita a integração das bases de dados representadas pelas ontologias através do *merging* das ontologias. O modelo global da federação de ontologias será então gerado, numa analogia ao modelo global gerado pela integração das bases de dados quando mapeadas para formação do modelo global da federação de bancos de dados.

4.3 Descrição do estudo de caso

Neste parágrafo nós apresentaremos o *merging* de ontologias que será utilizado para a criação de uma federação de ontologias para integração semântica de dados. As ontologias utilizadas são do domínio de Telecom mais especificamente de Marketing de Telecom. Elas estão baseadas nos modelos de dados existentes; modelo de dados Telefonia e Campanha que são bases de dados da empresa e que foram apresentadas nos parágrafos anteriores.

Foi eleito o ambiente de Telecom para a geração do nosso estudo de caso, devido à nossa experiência real de geração de modelos e integração de dados em um ambiente como este. Ressaltamos que apesar da experiência ter sido real os modelos são fictícios, ou seja, foram adaptados para garantia de sigilo sem deixar, no entanto, de representar muito propriamente um caso geral de uma empresa deste tipo.

A primeira ontologia contém conceitos e relações que descrevem o domínio que envolve o processo de criação de produtos de telefonia pelo departamento de Marketing da empresa bem como informações de clientes que usam ou aderem a estes produtos, a segunda ontologia contém conceitos e relações que descrevem o processo de Campanha para comercialização de um produto. Para a edição das ontologias foi utilizado o editor de ontologias Protégé.

4.3.1 O editor de ontologias Protégé

O Protégé (Noy et al., 2001) é uma ferramenta construída pela Seção de Informática Médica da Universidade de Stanford. Ele possui um ambiente de edição de bases de conhecimento e uma arquitetura extensível para a criação de outras ferramentas, ou seja, ele também tem uma API. Esta API é dedicada a desenvolvedores de software que desejem implementar novas linguagens e

características que eles gostariam de suportar em suas aplicações. Está disponível em diferentes plataformas, como *Linux*, *Unix, Solaris*, *Mac*, *OS* e *Windows*.

A ferramenta tem ganhado muitos adeptos que a utilizam para modelar um largo escopo de domínio. Protégé está disponível gratuitamente através da licença *Mozilla* de fonte aberta.

O Protégé conta com um ambiente gráfico e interativo para o projeto de ontologias e bases de conhecimento. Isto ajuda os especialistas do domínio a realizarem suas tarefas. Os desenvolvedores de ontologias podem acessar informações importantes quando necessário, e pode-se usar manipulação direta para se administrar uma ontologia. Controles em árvore permitem uma navegação simples e rápida através da hierarquia de classes.

O Protégé usa formulários como sua interface para preenchimento dos valores das aberturas (*slots*). Ele inclui suporte para classes e hierarquia de classes, com herança múltipla, modelos e aberturas (*slots*) próprias, especificações de facetas (*facets*) pré-definidas ou arbitrárias para aberturas, que incluem valores permitidos (*allowed values*), restrições de cardinalidade, abertura inversas (*inverse slots*), meta-classes e hierarquia de meta-classes.

Em adição à interface com boa usabilidade, podem-se destacar ainda duas qualidades: escalabilidade e extensibilidade.

Uma das maiores vantagens da arquitetura do Protégé é que o sistema é construído em uma forma aberta e modular. Esta arquitetura baseada em componentes permite aos construtores do sistema a adição de novas funcionalidades com a criação de *plugins*.

A maioria dos *plugins* pode ser classificada em uma destas três categorias: *Backends* que permitem ao usuário a importação e a exportação das ontologias em vários formatos diferentes, como esquemas RDF, arquivos XML com um DTD, arquivos de esquemas XML, OIL, OWL e DAML+OIL; *Slot widgets* que são usados para mostrar e editar valores das aberturas ou suas combinações em tarefas específicas de domínios específicos. Alguns incluem interfaces com imagens, vídeo e áudio e outros permitem que se criem os elementos da base de conhecimento através da manipulação de um diagrama colorido; *Tab plugins* que são aplicações baseadas em conhecimento estritamente conectadas com as bases de conhecimento Protégé.

Estes últimos são os tipos mais populares e incluem *plugins* de visualização, combinação e administração de versões como os *plugins OntoViz* e *Jambalaya* que são exemplos de visualizadores gráficos da base de conhecimento. O *Jambalaya* é um *plugin* que está incluído na versão completa do Protegé e que usa *ShriMP*. *ShriMP* é uma técnica de visualização independente de domínio desenhada para ajudar pessoas que exploram e trabalham com informações espaciais complexas. Já os *plugins Flora* e o *Jess* provêm acesso a vários motores de raciocínio (*reasoning engines*).

O *plugin Prompt* provê um ambiente para a administração de múltiplas ontologias. Seus componentes incluem ferramentas para combinação de ontologias que auxiliam o usuário a encontrar similaridades entre as fontes das ontologias para combiná-las, ferramentas para controle de versões, que automaticamente encontram diferenças estruturais entre versões de uma mesma ontologia e ferramentas para extração semântica de partes de uma ontologia e rearranjo dos quadros entre diferentes ontologias ligadas. Este *plugin, Prompt* será utilizado na integração das ontologias que serão definidas no nosso estudo de caso.

4.3.2 Definição das ontologias

Diversas metodologias têm sido desenvolvidas no intuito de sistematizar a construção de ontologias. O nosso trabalho seguirá as etapas do modelo proposto por (Noy et al., 2001). Para a implementação da ontologia será utilizada a linguagem OWL.

4.3.2.1 Definição da ontologia Telefonia

O domínio utilizado para esta ontologia refere-se ao Modelo de Telefonia que trata do relacionamento dos produtos criados pelo departamento de marketing da empresa de Telecom com os clientes que os adquirem. A identificação dos termos relevantes para representar este domínio teve por base o modelo de dados equivalente - Telefonia.

Neste domínio uma empresa é constituída de departamentos. Dentre todos os departamentos relacionados aos produtos oferecidos pela empresa o departamento de marketing tem um papel fundamental, pois este departamento é

responsável por criar produtos oferecidos pela empresa, ou seja, alguns dos funcionários deste departamento são responsáveis pelo estudo da viabilidade de criação de um novo produto. Eles formam a equipe de estudos. O processo de criação de um novo produto tem início nos estudos realizados por esta equipe de especialistas. Os produtos criados por eles e que apresentam características comuns são agrupados em pacotes, desta forma um produto pode ou não pertencer a um pacote. Os clientes por sua vez utilizam estes diversos produtos de diversas formas (horários diferentes). A Figura 5 mostra a hierarquia das classes da ontologia Telefonia editada no Protégé.

Figura 5: Hierarquia de classes da ontologia Telefonia.

Os termos que fazem parte da definição Funcionário como Administrador, por exemplo, são definidos na ontologia como subclasses desta superclasse. Outros elementos que não possuem a característica de subclasse como faz_parte são representados como propriedades. Na Figura 6 são apresentadas as classes e respectivas propriedades da ontologia Telefonia.

Termos do Domínio				
Cliente	**Empresa**	**Funcionário**	**Pacote**	**Produto**
Cod_Conta	CGC	trabalha	Descrição_Pacote	CodProduto
Cod_Estado_Conta	compoe_se_de		é_formado_por	Descrição_Produto
CodProduto	Endereço			faz_parte
contrata	Nome_fantasia			
Data_Vencimento	Telefone			
Endereco				
Endereco_Cobranca				
Nome_Cliente				
Pais				
Endereco_Eletrônico				
Período_de_Bilhetagem				
Telefone_de_Contato				
Corporativo	**Depto_marketin**	**Administrador**		**Ativo**
CGC	CodProduto	trabalha		
Individual		**Esp_Informática**		**Inativo**
CPF		trabalha		
		Esp_Marketing		
		trabalha		
		cria		

Figura 6: Classes e propriedades do domínio telefonia.

4.3.2.2 Formalização da ontologia

Nesta fase da ontologia são definidos os axiomas, ou seja, as classes de inferência. A partir delas é possível estabelecer as restrições do domínio. Essas classes estarão mais bem evidenciadas no parágrafo em que trataremos da camada de aplicação do nosso modelo.

4.3.2.3 Construção das classes e subclasses

A fase final no desenvolvimento de ontologias é realizada por meio de uma representação específica que permite o processamento e a abrangência do conhecimento pela máquina. Isso é possível através de uma linguagem específica

para a criação de ontologias e de uma ferramenta que permita sistematizar e integrar as especificações definidas à linguagem utilizada.

Como dito anteriormente, a linguagem utilizada neste trabalho foi a OWL e para desenvolver as ontologias através da utilização desta linguagem foi usada a ferramenta Protégé. Para definição das classes da ontologia Telefonia, foram utilizadas algumas propriedades oferecidas pela linguagem OWL, como: *minCardinality*, para enumeração da cardinalidade; *equivalentClass* para apresentar características semelhantes à outra classe e *comment*, para atribuir comentários da classe definida. Para exemplificar apresentamos a definição de uma das classes da ontologia, a classe Produto como mostra a Figura 7.

```
<owl:Class rdf:ID="Pacote" />
  <owl:Class rdf:ID="Produto" />
   <owl:Class rdf:ID="Ativo">
    <rdfs:subClassOf rdf:resource="#Produto" />
    </owl:Class>
   <owl:Class rdf:ID="Inativo">
    <rdfs:subClassOf rdf:resource="#Produto" />
    </owl:Class>
   <owl:ObjectProperty rdf:ID="cria">
    <rdfs:domain rdf:resource="#Esp_Marketing" />
    <rdfs:range rdf:resource="#Produto" />
    </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="contrata">
    <rdfs:domain rdf:resource="#Cliente" />
    <rdfs:range rdf:resource="#Produto" />
    </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="faz_parte">
   <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="é_formado_por" />
    </owl:inverseOf>
    <rdfs:range rdf:resource="#Pacote" />
    <rdfs:domain rdf:resource="#Produto" />
    </owl:ObjectProperty>
```

Figura 7: Definição da classe Produto.

4.3.2.4 Construção das propriedades

As propriedades capturam as diferentes variáveis relativas ao domínio da ontologia e em OWL são caracterizadas como *objectProperty* ou *Datatype*. Em ambas, é necessária a definição do *Domain* (domínio) e o *Range* (valor) da propriedade.

Uma propriedade é declarada como do *objectProperty* quando tem o papel de relacionar uma classe à outra classe. Este tipo de propriedade foi identificado no domínio Telefonia e pode ser visualizado na Figura 8.

```
</owl:ObjectProperty>
 <owl:ObjectProperty rdf:ID="faz_parte">
 <owl:inverseOf>
 <owl:ObjectProperty rdf:ID="é_formado_por" />
 </owl:inverseOf>
 <rdfs:range rdf:resource="#Pacote" />
 <rdfs:domain rdf:resource="#Produto" />
 </owl:ObjectProperty>
 <owl:ObjectProperty rdf:about="#é_formado_por">
 <rdfs:domain rdf:resource="#Pacote" />
 <owl:inverseOf rdf:resource="#faz_parte" />
 <rdfs:range rdf:resource="#Produto" />
 </owl:ObjectProperty>
```

Figura 8: Exemplo de propriedade declarada como *objectProperty*.

A propriedade do tipo *Datatype* se diferencia da *objectProperty* por utilizar uma variável para representar qualquer coisa no domínio abordado. Neste tipo de propriedade, também é necessário definir o domínio ao qual ela pertence e o seu valor que não mais será uma classe, mas um elemento do tipo *string*, *boolean*, *int* entre outros. Na Figura 9, a propriedade Data_vencimento possui como domínio a classe Cliente e como valor, o tipo *int.*

```
                    </owl:DatatypeProperty>
          - <owl:DatatypeProperty rdf:ID="Data_Vencimento">
                <rdfs:domain rdf:resource="#Cliente" />
<rdfs:rangerdf:resource="http://www.w3.org/2001/XMLSchema#int" />
                    </owl:DatatypeProperty>
```

Figura 9: Definição da propriedade Data_vencimento.

4.3.2.5 Checagem da consistência da ontologia Telefonia

O Protegé oferece a possibilidade de consultar um raciocinador (provador de teoremas) para *Description Logic* (DL) com sintaxe OWL-DL. Isto é feito via um protocolo chamado DIG que é responsável pela comunicação entre o Protegé e o provador. O provador utilizado como servidor DIG neste trabalho é o *Pellet* (versão 1.3beta2). O *Pellet* implementa um algoritmo de prova como o definido

originalmente para o Tableau de *Description Logic* (Baader et al., 2003). O *Pellet* também permite consultas via RDQL, sendo esta sua única possibilidade de dedução sobre a A-Box. Quanto à dedução na T-Box, o procedimento utilizado é a criação de uma classe de consulta expressando a propriedade que se deseja verificar ser ou não conseqüência lógica. Através então de um procedimento de classificação esta consulta, caso seja conseqüência lógica, é classificada no conceito adequado da T-Box. Este procedimento foi o utilizado nos testes de autorização de usuários. O resultado do teste de consistência da ontologia está mostrado na Figura 10.

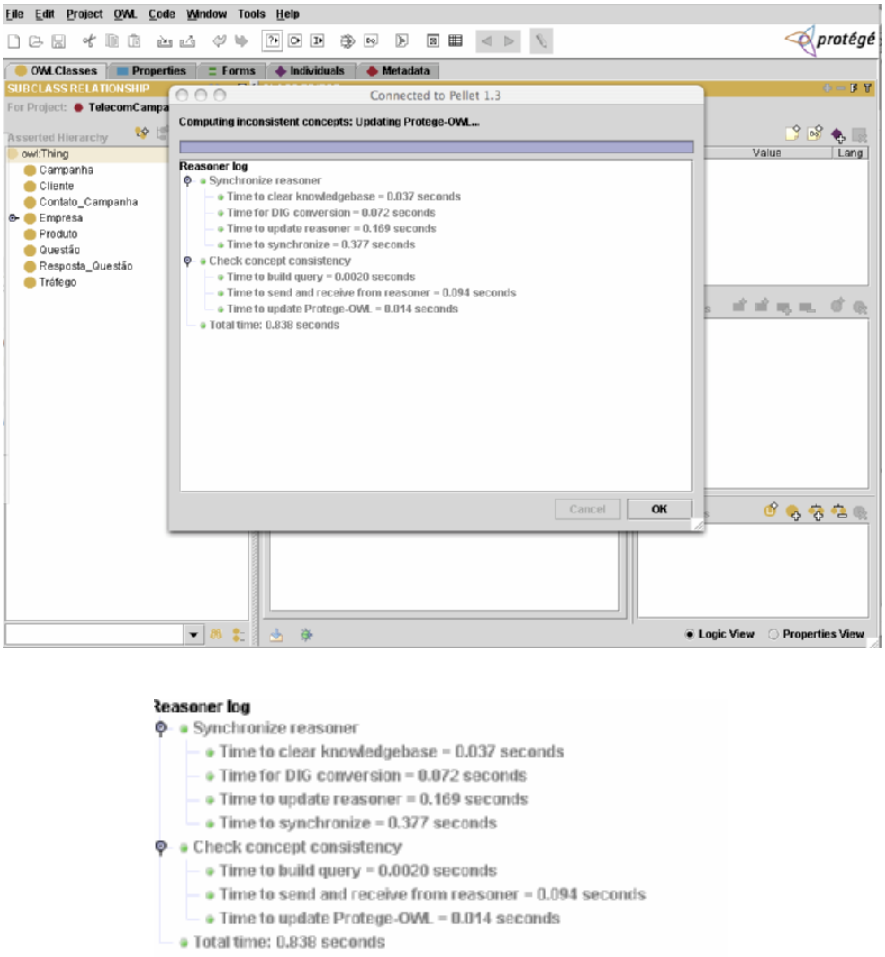


Figura 10: Resultado do teste de consistência da ontologia Telefonia usando o provador *Pellet*.

4.3.2.6 Construção dos axiomas

Na nossa ontologia Telefonia a construção dos axiomas é a construção das propriedades das classes e subclasses.

4.3.2.7 Definição da ontologia Campanha

O domínio utilizado para esta ontologia refere-se ao Modelo de Campanha que trata da campanha que envolve o lançamento de um novo produto. Um produto é criado pelo departamento de marketing e mobiliza uma campanha para seu lançamento. Esta campanha, como já foi dito, gera um *script* a ser seguido para divulgação. Este *script* possui uma questão que deve ser respondida pelo cliente quando contatado. A resposta desta questão é relevante para poder traçar-se o perfil do cliente e deve ser anotada. Quando faz uso do produto contratado por ele, o cliente gera o seu "tráfego".

A identificação dos termos relevantes para representar a ontologia gerou a hierarquia de classes que está representada na Figura 11 e que foi obtida usando o editor de ontologias Protégé.

Figura 11: Hierarquia das classes da ontologia Campanha.

4.3.2.8 Propriedades do domínio Campanha

Na Figura 12, são apresentadas as classes e propriedades do domínio Campanha.

Termos do Domínio				
Campanha	**Cliente**	**Contato_Campanh**	**Empresa**	**Departamento**
CodCampanha	adquire	Cod_Contato	CGC	CodDepto
CodProduto	Cod_Conta	CodCampanha	Endereço	Descrição_Depto
Descrição_Campanha	CodProduto	CodProduto	Nome_fantasia	CGC
realiza	CPF_CGC	Data_Contato	Telefone	Endereço
tem	Endereço	Tipo_de_Estudo		Nome_fantasia
Tipo_de_Estudo	Nome_Cliente	Área		Telefone
Área	Pais			
	produz			
	Telefone_de_Contat			
Produto	**Questão**	**Resposta_Questão**	**Tráfego**	**Dep_Marketing**
CodProduto	CodCampanha	CodCampanha	Período_de_Bil	CodProduto
demanda	CodProduto	CodProduto		CGC
Descrição_Produto	CodQuestão	CodQuestão		CodDepto
Tipo_de_Estudo	contêm	Tipo_de_Estudo		Descrição_Depto
Área	Descrição_Questão	Área		Endereço
	Tipo_de_Estudo			Nome_fantasia
	Área			Telefone
			Engenharia	**Financeiro**
			CGC	CGC
			CodDepto	CodDepto
			Descrição_Dept	Descrição_Depto
			Endereço	Endereço
			Nome_fantasia	Nome_fantasia
			Telefone	Telefone

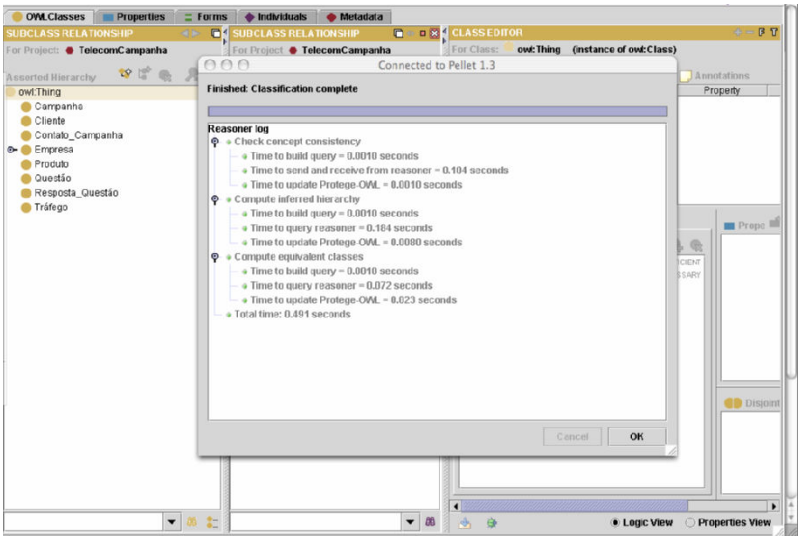
Figura 12: Classes e propriedades do domínio Campanha.

4.3.2.9 Construção dos axiomas

Na nossa ontologia Campanha a construção dos axiomas é a construção das propriedades das classes e subclasses.

4.3.2.10 Checagem da consistência da ontologia Campanha

O procedimento realizado para a checagem de consistência da ontologia Campanha foi o mesmo utilizado quando da checagem da ontologia Telefonia. O raciocinador utilizado foi o *Pellet* e o resultado está mostrado na Figura 13.



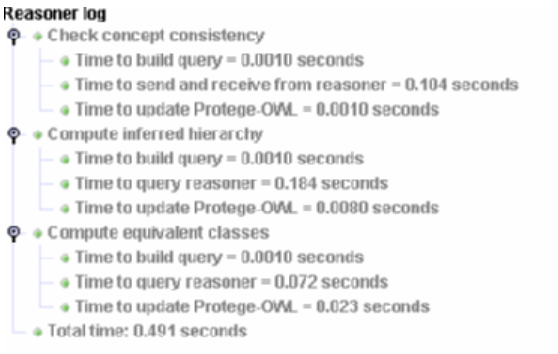


Figura 13: Resultado de teste de consistência da ontologia Campanha usando o provador Pellet.

4.3.3 A ontologia Resultante do *Merging* de Ontologias – O Modelo Global

De maneira similar à federação de bancos de dados onde o modelo global é a integração dos múltiplos esquemas de exportação, também chamado de Esquema Conceitual Global, construiremos o nosso modelo de integração das ontologias. Ele será o resultado do *merging* das ontologias Telefonia e Campanha, que serão integradas usando a ferramenta *iPrompt.*

4.3.3.1 O algoritmo *Prompt*

Pesquisadores em diversas áreas da ciência da computação têm trabalhado no *merging* de ontologias automático ou suportado por ferramenta, no entanto, *merging* automático de ontologias e criação de ferramentas que possam guiar o usuário através do processo e focar sua atenção nos pontos chave para ação são ainda tentativas recentes.

Prompt é um algoritmo independente de formalismo para alinhamento e *merging* de ontologias que automatiza o processo tanto quanto possível. Onde não é possível uma decisão automática o algoritmo guia o usuário para os lugares na ontologia onde a sua intervenção é necessária, sugere possíveis ações e determina os conflitos na ontologia e propõe soluções para esses conflitos.

O algoritmo *Prompt* está desenvolvido em cima de um modelo de conhecimento baseado em frame e está desenhado para ser compatível com OKBC (Chaudhri et al, 1998). No nível mais alto existem classes, *slots, facets* e instâncias.

Classes: são coleções de objetos que têm propriedades similares. Classes são organizadas em hierarquia de subclasses com herança múltipla. Cada classe tem *slots* atachados a ela. *Slots* são herdados pelas subclasses.

Slots: são relações binárias entre classes ou objetos primitivos (como uma string ou um número). *Slots* atachados a uma classe pode ser utilizado por outra classe.

Facets: são relações ternárias entre uma classe, um *slot* e outra classe ou um objeto primitivo. *Facets* pode impor regras adicionais a um *slot* atachado a uma classe, como cardinalidade ou tipo.

Instâncias: são membros individuais de uma classe.

Estas definições são apenas restrições para o *input* de ontologias para o algoritmo *Prompt,* mas, seu modelo de conhecimento é muito geral e as soluções de *merging* e alinhamentos, produzidas por ele podem ser aplicadas em modelos de conhecimento compatíveis com o modelo no qual ele se apóia.

O algoritmo foi implementado em um conjunto de ferramentas iterativas baseado no Protégé-2000 (Noy & Musen, 2000), um ambiente de modelo de conhecimento que foi apresentado no parágrafo 4.3.1.

Segundo (Noy & Musen, 2003), o conjunto de ferramentas *Prompt* é formado por:

- ***iPromp***, uma ferramenta interativa para *merging* de ontologias;
- ***AnchorPrompt***, uma ferramenta automática baseada em grafos para alinhamento de ontologias.
- ***PromptFactor***, uma ferramenta para extração de partes de ontologias.
- ***PromptDiff***, uma ferramenta para identificação de diferenças entre duas versões da mesma ontologia.

O conjunto de ferramentas *Prompt* e como estas estão relacionadas são ilustrados na Figura 14.

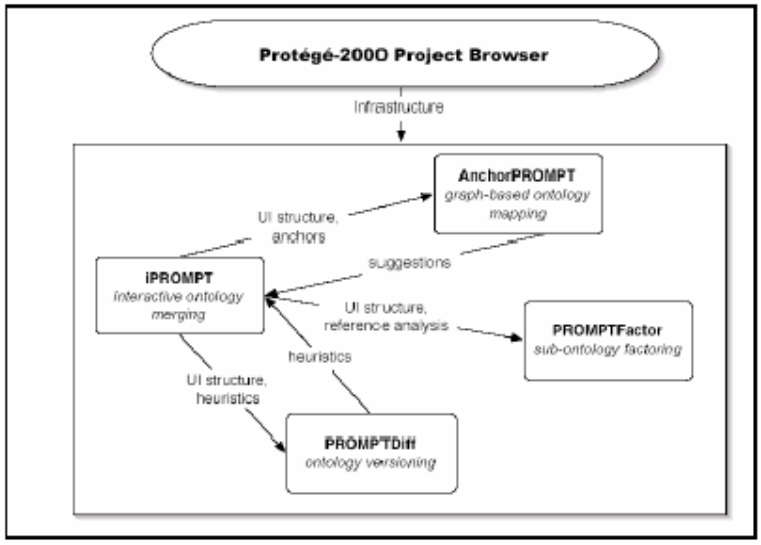


Figura 14: O conjunto de ferramentas Prompt (retirado de Noy & Musen, 2003).

A ferramenta *iPrompt* (Noy & Musen, 2003), é uma ferramenta semi-automática de *merging* de ontologias. Esta ferramenta guia o usuário, apresentando sugestões para os termos das ontologias que devem ser combinados, e identifica inconsistências e problemas potenciais, sugerindo estratégias para resolvê-los. Seu algoritmo faz uso tanto da informação da estrutura dos conceitos na ontologia e relações entre eles quanto da informação obtida do usuário (Felicíssimo, 2004). No entanto, as informações analisadas entre os conceitos são limitadas ao contexto local, ou seja, apenas são analisadas as informações das

relações entre os conceitos ligados diretamente e aqueles que são referenciados nas restrições destas relações.

A ferramenta *AnchorPrompt* (Noy & Musen, 2003), é uma ferramenta para o alinhamento de ontologias que encontra automaticamente termos semanticamente similares. Esta ferramenta tem como entrada um conjunto de âncoras, ou seja, pares de termos relacionados, definidos pelo usuário ou por identificação automática de combinação lexical, e trata uma ontologia como um grafo. Neste grafo, os conceitos das ontologias são seus nós e suas relações são seus *links*, suas ligações.

Os caminhos do sub-grafo limitado pelas âncoras são analisados e são determinados quais os conceitos que freqüentemente aparecem nas mesmas posições dos caminhos similares (Felicíssimo, 2004). Com estas análises e determinações, além do contexto local, o contexto não-local também é analisado. Além disto, utiliza na detecção de termos similares, a relação estrutural dos termos de ontologias comparadas, medidas de similaridades pré-definidas e grupos de equivalência.

A ferramenta *PromptFactor* (Noy & Musen, 2003), é uma ferramenta para separação de sub ontologias, semanticamente independentes de ontologias extensas. Esta ferramenta é de grande auxílio para autores de ontologias que desejam criá-las reutilizando partes de ontologias extensas já criadas. Isto porque, tal ferramenta trata-se de uma possível solução para o problema de reutilização de termos de ontologias extensas sem ter que importá-las em sua totalidade. Segundo (Felicíssimo, 2004) é conhecido que, ao importar uma ontologia, há a adição de todos os termos da ontologia importada na que faz uso da importação. Para as ontologias que utilizam apenas um conjunto de termos de ontologias importadas, o modelo fica desnecessariamente robusto com o excesso de informação acrescido. Este problema é agravado quando a ontologia importada é extensa.

A ferramenta *PromptDiff* (Noy & Musen, 2003), é uma ferramenta para identificação automática de diferenças entre duas versões da mesma ontologia. Nesta ferramenta, a comparação não é feita apenas por comparação de textos, como tradicionalmente é realizado em comparação de versões de programas, mas também, por comparação estrutural. O *PromptDiff* compara as estruturas de duas versões da mesma ontologia identificando as partes que não tiveram alteração alguma, aquelas que tiveram alterações apenas em suas propriedades e aquelas

que tiveram alterações nos nomes e/ou estruturas tanto de seus conceitos quanto de suas propriedades, entre outras partes.

4.3.3.2 O modelo global da federação de ontologias

De acordo com a arquitetura mostrada na , as bases de dados BD1 e BD2 com os seus respectivos modelos de dados que foram mapeadas nas ontologias Telefonia e Campanha serão as bases para a geração do modelo global, que será resultante do *merging* das mesmas.. Estas ontologias são as **ontologias normalizadas** que representam um modelo de dados comum, equivalente ao esquema componente na construção de federação de banco de dados, ou seja, são a tradução dos esquemas locais para um modelo de domínio a ser adotado na federação de ontologias.

As **ontologias de exportação** são visões das ontologias normalizadas que de maneira similar aos esquemas de exportação na federação de dados, descrevem as partes relevantes da ontologia que estarão disponíveis na federação. Foi decisão nossa de projeto considerar como ontologias de exportação as mesmas ontologias normalizadas, ou seja, todos os termos do domínio de cada ontologia normalizada estarão disponíveis para exportação, são candidatos a fazer parte do esquema global. Isto se deu porque com uma quantidade maior de objetos fazendo parte do *merging*, poderíamos fazer análises mais completas sobre o mesmo, poderíamos avaliar melhor o comportamento do algoritmo na realização da integração, principalmente na solução de conflitos semânticos.

4.3.4 O processo de integração das ontologias

De maneira similar ao modelo global gerado para a federação de dados, a **ontologia global** para a federação de ontologias será a integração das múltiplas ontologias de exportação. No nosso estudo de caso ela será a integração das ontologias Telefonia e Campanha.

O processo de geração do *merging* das ontologias teve início com a escolha das ontologias Telefonia e Campanha para serem processadas pela ferramenta *iPrompt*. O processo para realizar o *merging* constitui-se dos seguintes passos:

Configurar o Protégé para que a ferramenta *iPrompt* seja incluída. O procedimento é o seguinte: acessar o *menu project* e escolher a opção configure e em seguida marcar nas opções disponíveis que aparecem o *PromptTab*.

O *PromptTab* apresenta uma tela onde é possível interagir e escolher que tipo de operação vai ser realizado nas ontologias, dentre elas; *compare* que faz o controle de versões, *move* que faz o rearranjo dos quadros entre diferentes ontologias ligadas, *merge* que faz o *merging* das ontologias - auxilia o usuário a encontrar similaridades entre as ontologias para integrá-las, e *extract* que faz a extração semântica de partes de uma ontologia, conforme mostrado na Figura 15. A opção escolhida foi o *merge* e a ferramenta utilizada para realizá-lo, associado a esta opção foi o *iPrompt*.

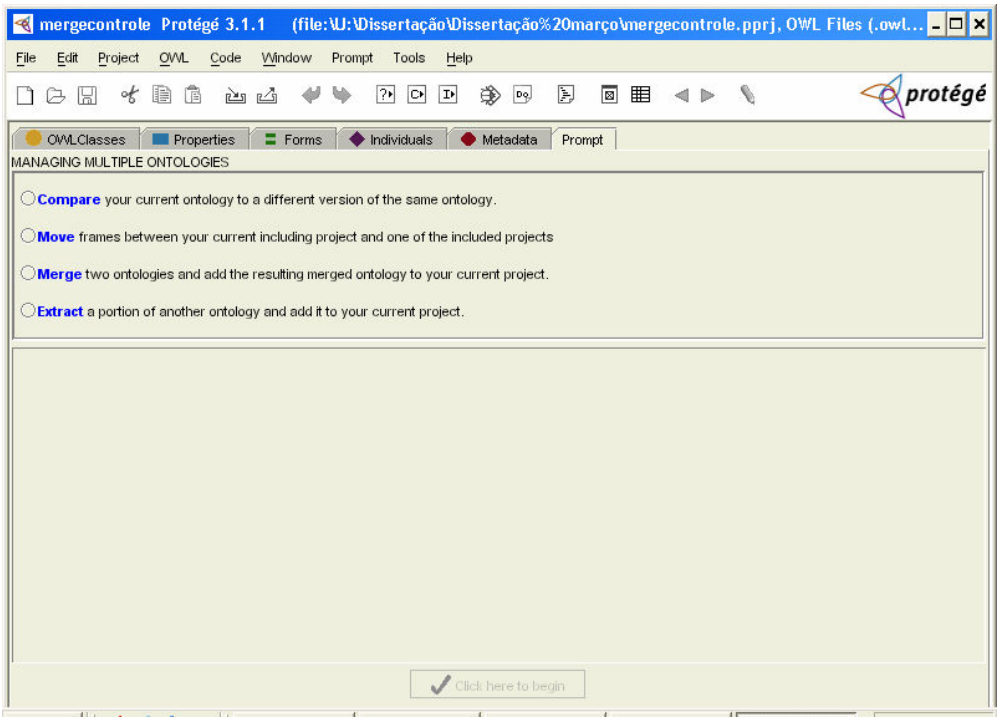


Figura 15: Tela do PromptTab.

Após a seleção da opção *merge*, o processo de *merging* acontece em seguida e o *Prompt* fornece como resposta uma lista de sugestões para realização deste. Estas sugestões são resultados do processamento do algoritmo e se resumem em termos que podem ser combinados (*merging*) sendo identificadas aqui as inconsistências e problemas. O algoritmo sugere estratégias para que os problemas encontrados sejam resolvidos. Os termos não comuns são sugeridos para serem

copiados para a ontologia resultante do *merging*. O algoritmo como dissemos, para produzir a ontologia resultante, faz uso tanto da informação da estrutura dos conceitos na ontologia e relações entre eles, quanto da informação obtida do usuário quando da sua interação na escolha das respostas.

Na realização do *merging* das ontologias Telefonia e Campanha, foi apresentada como situação de conflito a ser resolvida pelo usuário, a definição de Esp_marketing e Dep_Marketing, subclasses respectivamente de Funcionário na ontologia Telefonia e de Departamento e Empresa na ontologia Campanha, esta situação está ilustrada na Figura 16. Aparentemente, esta não deveria ser uma situação de conflito uma vez que as subclasses têm classes diferentes como origem e são sintaticamente diferentes. Esses nomes foram identificados como sinônimo pelo algoritmo e isto se deve à sua estrutura de funcionamento. O algoritmo usa medidas de similaridades simples para encontrar classes com nomes similares.

O início do processo, ou seja, a comparação inicial se dá por busca de similaridades lingüísticas, mas a decisão final se concentra em buscar regras baseadas na semântica da ontologia e nas ações do usuário. Embora o algoritmo tenha sido implementado usando medidas simples de similaridade lingüística, segundo (Noy & Musen, 2003), a ferramenta *iPrompt* foi desenvolvida de tal forma que outros algoritmos de comparação de termos possam ser *plugados* nela. O uso de WorNet, por exemplo, para encontrar sinônimos pode ser uma extensão natural.

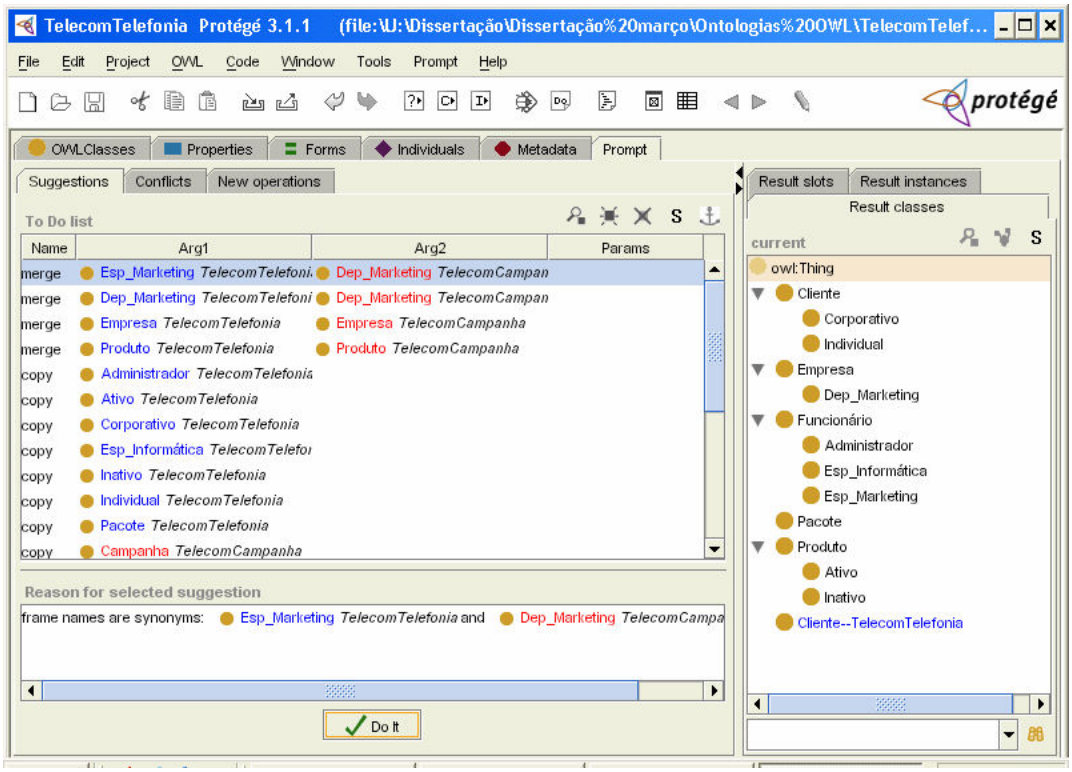


Figura 16: Situação de conflito identificada pelo iPrompt.

4.3.4.1 Hierarquia da ontologia resultante do *merging*

Após a identificação dos conflitos apresentados na tela pela ferramenta i*Prompt* foi necessária a interação humana para solução destes conflitos. Para solucioná-los, tomamos por base nosso conhecimento dos conceitos e regras do negócio de Marketing de Telecom. Nota-se aqui a presença do especialista do domínio possibilitando a geração da ontologia resultante do merging. O resultado final está mostrado na Figura 17. Esta ontologia representa a integração dos domínios Telefonia e Campanha, é o modelo global da federação de ontologias.

Figura 17: Hierarquia da ontologia resultante do *merging*.

4.3.4.2 Análise de consistência da ontologia resultante do *merging*

Foi realizada a análise de consistência da ontologia resultante do *merging*. O processo foi similar ao realizado para as ontologias Telefonia e Campanha, usando o provador Pellet. O resultado do teste está mostrado na Figura 18.

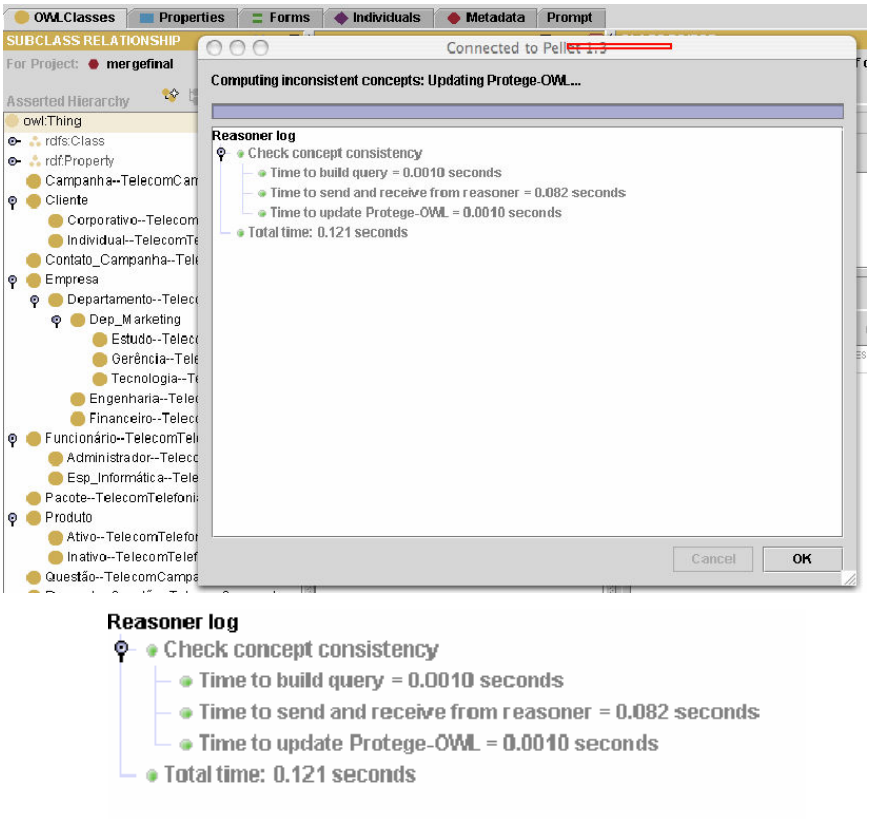


Figura 18: Resultado do teste de consistência da ontologia global.

4.3.5 A ontologia para a camada de aplicação

De maneira similar ao esquema externo para uma federação de bancos de dados, vamos definir uma camada de aplicação para a ontologia integrada. Esta camada define uma visão que um usuário ou uma aplicação terão da federação. O uso de visões pode viabilizar a personalização das informações disponibilizadas aos usuários.

A nossa camada de aplicação vai estar centrada no controle de acesso às informações disponíveis na ontologia integrada. A ontologia integrada representa um domínio de Telecom e num domínio deste tipo é necessário prever o controle

de acesso a informações que são confidenciais como, por exemplo, de dados de clientes.

A segurança de dados é uma função importante em um sistema de banco de dados, pois protege os dados contra acessos não autorizados. A solução para este problema tem sido objeto de vasta pesquisa pela comunidade de banco de dados dentre essas soluções podemos destacar as apresentadas por (Ozsu, 1999), (Wilms & Lindsay, 1981) e (Stonebraker, 1975).

Segundo (Ozsu, 1999) a segurança de dados inclui dois aspectos: *proteção dos dados* e *controle de autorização*. A *proteção dos dados* é necessária para evitar que usuários não autorizados reconheçam o conteúdo físico dos dados, sendo a criptografia de dados a principal abordagem de proteção dos dados, nela dados criptografados podem ser descriptografados somente por usuários autorizados que conheçam o código. Os dois principais sistemas de criptografia são o Data Encryption Standard (NBS, 1997) e os esquemas de criptografia de chave pública (Rivest et al., 1998). O Segundo aspecto de segurança, o *controle de autorização*, deve garantir que apenas usuários autorizados executem operações que eles têm permissão para executar sobre o banco de dados.

Para que pudéssemos atender a um modelo mais completo de controle de autorização, ou seja, para uma melhor ilustração do nosso estudo de caso, a ontologia global resultante do processo de *merging* foi estendida. Novos usuários foram criados e a cada um foi dada autorização diferente de manipulação e acesso às informações disponíveis na ontologia. Essas autorizações se resumem em permissão para leitura, (permissão de seleção ou *Select* na linguagem SQL, numa analogia ao que acontece em banco de dados) e permissão para atualização (permissão para *Update* em SQL) em cada classe da ontologia. Cada classe aqui pode ser vista como um objeto de uma base de dados, uma tabela no caso relacional.

Partindo desta analogia, foi criada uma tabela relacionando os diferentes usuários e respectivos acessos a cada classe da ontologia global. Esta tabela está ilustrada na Figura 19.

Tipo de Funcionário (Usuário)	Cliente	Depto	Produto	Pacote	Questão	RespQuest	Tráfego	Empresa
Adm. Gerente	Select	Select	Select	Select	Select	Select	Select	Select
Esp Inf Adm_Dados	Select	Update where Cod Depto <> Ger	Select	Select	Select	Select	Select	Select
Esp Inf Analista Inf	Select	Select	Select	Select	Select	Select		
EspInf Analist_Inf Produção(DBA)	Update	Update	Update	Update	Update	Update	Update	
Esp Makt Analista Neg	Select	Select where cod <> Ger	Update	Update	Update	Select	Select	
Esp Mkt Analist Produto	Select where cliente <> Coorporativo	Select where cod <> Ger	Select	Select	Select	Update		
Esp Makt Gerente Produto	Select	Select	Update	Update	Update	Select	Select	
Esp. Mkt Contato com Usuário	Select Where cod cliente <> Coorporativo			Select				

Figura 19: Mapeamento do acesso dos usuários.

O modelo de controle de acesso descrito na tabela foi implementado na ontologia global junto com os novos usuários. Ele foi traduzido na linguagem OWL-DL, linguagem disponibilizada pelo *Protégé*. Na Figura 20 nós podemos ver a hierarquia da ontologia de controle de acesso já com os novos usuários definidos e com as respectivas permissões.

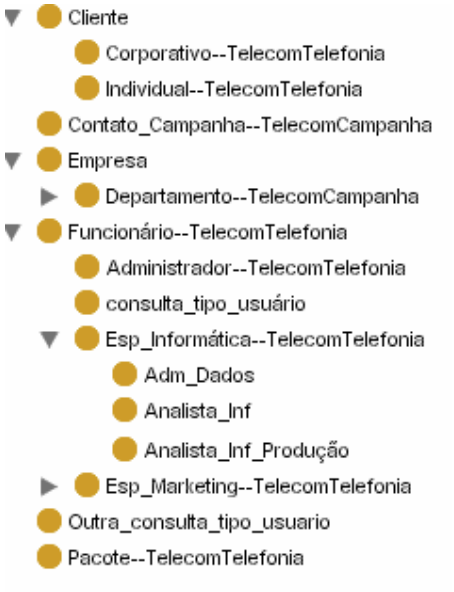


Figura 20: Hierarquia da ontologia de controle de acesso.

Para que o modelo de controle de autorização fosse testado foi criado um usuário de nome *consulta_tipo_usuário.* Este usuário é criado como fazendo parte (subclasse) da classe geral *Thing.* Sobre este usuário foram realizadas inferências com o propósito de poder reclassificá-lo no modelo dando a ele algumas regras a serem atendidas.

A pergunta a ser respondida foi a seguinte:

Como classificar o usuário (classe) de nome consulta_tipo_usuário que tem permissão de seleção nas diversas classes (Select), mas que não pode fazê-lo na classe Cliente e que pode fazer atualização (Update) nas classes Cliente, Pacote, Questão e Resposta_Questão?

Este usuário (classe) está mostrado na Figura 21.

Figura 21: Tela mostrando o usuário criado para realizar inferência.

O raciocinador Pellet é chamado e é pedido que ele classifique a ontologia. Figura 22.

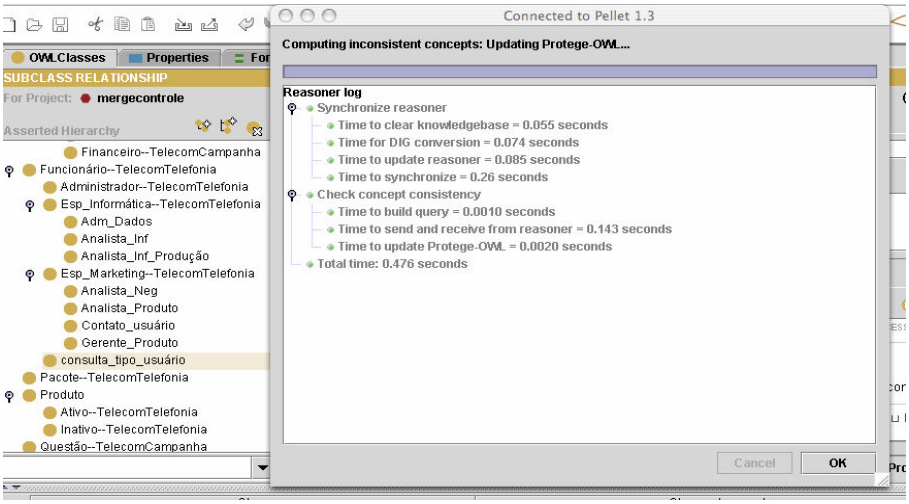


Figura 22: Classificação do usuário criado para inferência.

Ele move a classe que em princípio está definida como uma subclasse de *Thing* e a coloca como pertencendo à classe *Esp_Marketing*, ou seja, ele será uma subclasse de *Esp_Marketing* que por sua vez é uma subclasse de *Funcionário*. Assim concluímos que a classe *usuário_tipo_consulta* é uma subclasse de

Esp_Marketing que é uma subclasse de *Funcionário*. O resultado está mostrado na Figura 23.

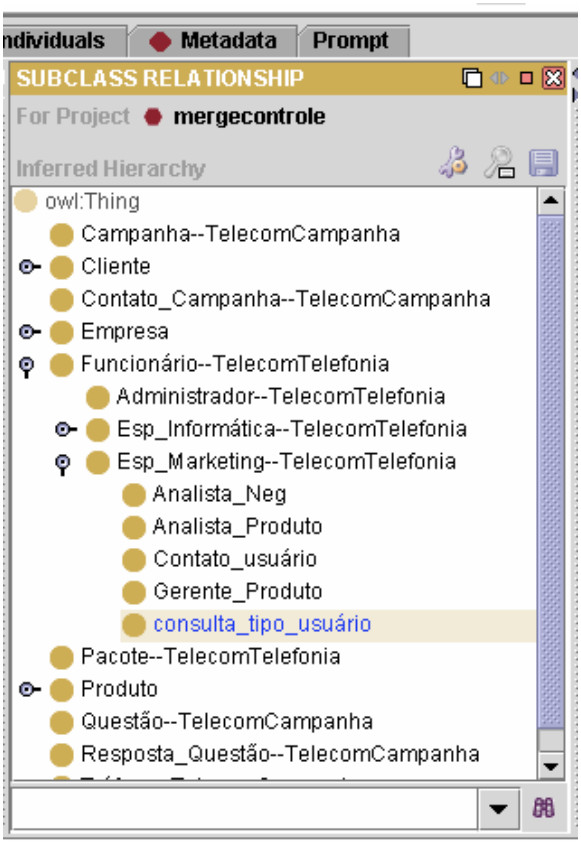


Figura 23: Tela mostrando o resultado da classificação realizada pelo raciocinador *Pellet.*

O critério de classificação utilizado pelo raciocinador é verificar classe a classe as propriedades de cada uma e verificar em qual delas a *consulta_tipo_usuário* está contida, pegando a mais específica.

Esta classificação é realizada de acordo coma sintaxe da *Description Logic* (DL) que foi utilizada para especificar as propriedades de cada classe, ou seja, cada usuário (classe) teve implementado as suas propriedades de permissão usando a linguagem de DL. Figura 24.

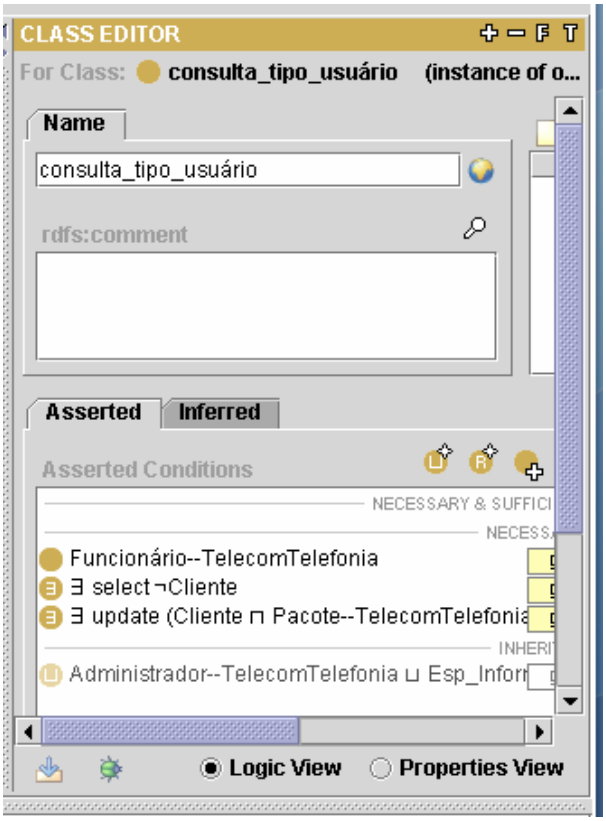


Figura 24: Propriedade de inferência em DL.