

UNIVERSIDADE FEDERAL DE SANTA CATARINA

**UMA ABORDAGEM PARA MANUTENÇÃO DE ONTOLOGIAS DE
UMA PLATAFORMA DE BUSINESS INTELLIGENCE SEMÂNTICO**

Fernando Benedet Ghisi

Florianópolis – SC

2008/2

**UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE SISTEMAS DE INFORMAÇÃO**

**UMA ABORDAGEM PARA MANUTENÇÃO DE ONTOLOGIAS DE
UMA PLATAFORMA DE BUSINESS INTELLIGENCE SEMÂNTICO**

Fernando Benedet Ghisi

Trabalho de conclusão de curso apresentado como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação.

Florianópolis - SC

2008/2

Fernando Benedet Ghisi

**UMA ABORDAGEM PARA MANUTENÇÃO DE ONTOLOGIAS DE
UMA PLATAFORMA DE BUSINESS INTELLIGENCE SEMÂNTICO**

Trabalho de conclusão de curso apresentado como parte dos requisitos para a obtenção do grau de Bacharel em Sistemas de Informação

Banca Examinadora:

Prof. Dr. José Leomar Todesco

Universidade Federal de Santa Catarina
Orientador

Prof. Dr. Denilson Sell

Universidade Federal de Santa Catarina
Co-orientador

Prof. Dr. Roberto Carlos dos Santos Pacheco

Universidade Federal de Santa Catarina

RESUMO

Segundo Sell (2006), com as soluções de BI tradicionais verificam-se desafios que permanecem não resolvidos no contexto da sociedade do conhecimento. Dentre esses desafios estão a falta de perspectiva de utilização da semântica do negócio (i.e. sua terminologia e regras) no apoio ao processamento analítico e a falta de flexibilidade para a extensão das funcionalidades exploratórias de acordo com as especificidades de cada organização. Sell propôs então uma arquitetura para BI, utilizando um novo paradigma - com poder analítico baseado na semântica e no conhecimento do negócio da organização, através do uso de ontologias e inferências semânticas. No presente trabalho, vislumbrou-se a possibilidade de auxiliar na aplicação de uma plataforma baseada nessa arquitetura, através da construção de um ambiente intuitivo e de fácil utilização para a realização do mapeamento e configuração, de forma semi-automática, das diferentes ontologias utilizadas por ela.

Palavras-chave: Business Intelligence; Ontologia; Manutenção de Ontologias; Arquitetura SBI.

ABSTRACT

For Sell (2006), there are challenges that remain unsolved in the context of the knowledge society on the traditional BI solutions, as the lack of perspective of the business semantics use (i.e., its terminology and rules) leveraging the analytical processing and the lack of flexibility to extend the exploration functionalities based on the specificities of each organization. Sell proposed a BI architecture, using a new paradigm – with the analytical power based on the semantics and on the business knowledge of the organization, through the use of ontologies and semantics inferences. In this study, is showed a possibility to help the use of a platform based on that architecture, with the development of an intuitive and easy to use tool for the mapping and the configuration, on a semi-automatic way, of the ontologies used by the platform.

KEYWORDS: Business Intelligence, Ontology, Ontologies Management, SBI Architecture.

*"Diga-me e eu esquecerei.
Mostre-me e eu lembrarei.
Envolva-me e eu aprenderei."
(Provérbio Chinês)*

LISTA DE FIGURAS

FIGURA 1 - EVOLUÇÃO DOS DADOS ATÉ A EXPERIÊNCIA	23
FIGURA 2 - ARQUITETURA DE DW/BI	27
FIGURA 3 - EXEMPLO DE UM MODELO DIMENSIONAL RELACIONADO AO PROCESSO DE PEDIDOS DE UMA ORGANIZAÇÃO	32
FIGURA 4 - LIGAÇÃO ENTRE UMA TABELA DE FATO E UMA TABELA DIMENSIONAL RELACIONADOS A UM PROCESSO DE VENDAS....	33
FIGURA 5 - A ARQUITETURA DA WEB SEMÂNTICA	41
FIGURA 6 - SEQÜÊNCIA HIERÁRQUICA DE COMPONENTES DE UMA URI	45
FIGURA 7 - EXEMPLO DE DUAS URIS, MOSTRANDO OS COMPONENTES DE SUAS PARTES	45
FIGURA 8 - EXEMPLOS DE URIS.....	46
FIGURA 9 – REPRESENTAÇÃO DA TRIPLA-RDF.....	50
FIGURA 10 - NOVA ARQUITETURA PARA A WEB SEMÂNTICA	54
FIGURA 11 - TIPOS DE ONTOLOGIA (AS SETAS REPRESENTAM RELAÇÕES DE ESPECIALIZAÇÃO)	56
FIGURA 12 - A ARQUITEURA SBI.....	60
FIGURA 13 - EXEMPLO DE UMA ONTOLOGIA DE DOMÍNIO (P&D)	62
FIGURA 14 - A ONTOLOGIA DE BI.....	64
FIGURA 15 - PARTE DA ONTOLOGIA ANALÍTICA UTILIZADA NO PRESENTE TRABALHO	72
FIGURA 16 - ARQUITETURA DO AMBIENTE PROPOSTO.....	77
FIGURA 17 – VISÃO GERAL DOS MÓDULOS UTILIZADOS NO PROTÓTIPO	82
FIGURA 18 - IMAGEM DO PROTÉGÉ EM EXECUÇÃO	89
FIGURA 19 - WebPROTÉGÉ.....	90
FIGURA 20 - PARTE DO MODELO DIMENSIONAL DA PLI UTILIZADO NA VALIDAÇÃO DO PROTÓTIPO	92
FIGURA 21 – TELA INICIAL DO PROTÓTIPO – ABA “ONTOLOGIA ISO”	93
FIGURA 22 - RESULTADO DO CLIQUE NO BOTÃO PARA GERAR ELEMENTOS DA ONTOLOGIA ISO	94
FIGURA 23 - POSSIBILIDADE DE CONFIGURAÇÃO DOS RELACIONAMENTOS ENTRE AS TABELAS.....	95
FIGURA 24 - ABA "ONTOLOGIA ANALÍTICA"	96
FIGURA 25 - TELA PARA DEFINIÇÃO DAS HEURÍSTICAS UTILIZADAS NO MAPEAMENTO	97
FIGURA 26 - POSSIBILIDADES DE CONFIGURAÇÃO DA ONTOLOGIA ANALÍTICA	98
FIGURA 27 - ILUSTRAÇÃO DA CAPACIDADE DE "DRAG AND DROP" DA APLICAÇÃO	99
FIGURA 28 - CONFIGURAÇÃO DOS DETALHAMENTOS DA UNIDADE DE TEMA SELECIONADA	100
FIGURA 29 - CONFIGURAÇÃO DE UMA HIERARQUIA DE ATRIBUTOS	101
FIGURA 30 - ABA "ONTOLOGIA DE DOMÍNIO"	102
FIGURA 31 - UTILIZAÇÃO DO ISEXTRACTA PARA VALIDAR OS MAPEAMENTOS REALIZADOS.....	103

LISTA DE TABELAS

TABELA 1 - COMPARAÇÃO DOS AMBIENTES OLTP E OLAP	36
TABELA 2 - REPRESENTAÇÃO DO MAPEAMENTO LÉXICO-VALOR PARA O TIPO DE DADO XSD:BOOLEAN DO XMLSCHEMA.....	51
TABELA 3 - TIPOS DE OWL	58
TABELA 4 - DESCRIÇÃO DAS PRINCIPAIS CLASSES DA ONTOLOGIA DE BI	66

LISTA DE ABREVIATURAS

API – Application Program Interface
BI - Business Intelligence
DBA – Data Base Administrator
DL - Description Logics
DM – Data Mining
DW - Data Warehouse
ETL - Extraction, Transformation and Loading
KDT – Knowledge Discovery in Textual Databases
MER – Modelo Entidade-Relacionamento
OIL - Ontology Inference Layer
OLAP - On-Line Analytical Processing
OLTP - On-Line Transaction Processing
OWL - Web Ontology Language
P&D – Pesquisa e Desenvolvimento
PLI – Plataforma Lattes Institucional
RDF - Resource Description Framework
RuleML - Rule Markup Language
SBI - Semantic Business Intelligence
SGBD - Sistema Gerenciador de Banco de Dados
SGML - Standard Generalized Markup Language
SQL - Structured Query Language
SWRL - Semantic Web Rule Language
TI - Tecnologia da Informação
W3C - World Wide Web Consortium
XML - eXtensible Markup Language

SUMÁRIO

RESUMO	4
ABSTRACT	5
LISTA DE FIGURAS	7
LISTA DE TABELAS	8
LISTA DE ABREVIATURAS.....	9
1. INTRODUÇÃO	13
1.2. OBJETIVOS	17
1.2.1. Objetivo geral	17
1.2.2. Objetivos específicos.....	17
1.3. DELIMITAÇÃO DO ESCOPO.....	18
1.4. JUSTIFICATIVA.....	18
1.5. METODOLOGIA	19
1.6. ORGANIZAÇÃO DO TRABALHO	20
2. FUNDAMENTAÇÃO TEÓRICA	21
2.1. GESTÃO DO CONHECIMENTO NAS ORGANIZAÇÕES	21
2.1.1. Dos dados à experiência organizacional.....	22
2.1.2. Por que gerir o conhecimento nas organizações?	23
2.1.3. Gestão do conhecimento e a tecnologia da Informação.....	24
2.2. BUSINESS INTELLIGENCE	26
2.2.1. Arquitetura de BI	27
2.2.2. Data Warehouse.....	28
2.2.3. Processo de ETL.....	29
2.2.4. Modelagem Dimensional	31
2.2.5. OLAP.....	33
2.2.6. Considerações sobre as soluções de DW/BI tradicionais	37
2.2.7. Business Intelligence 2.0.....	38
2.3. TECNOLOGIAS SEMÂNTICAS.....	40
2.3.1. A Web Semântica e suas camadas.....	40
2.3.1.1. Unicode e URI.....	41
2.3.1.2. XML, Namespace e XMLSchema.....	46
2.3.1.3. RDF e RDFSchemas	49
2.3.1.4. Ontology (Ontologias).....	52

2.3.1.5.	Logic (Lógica)	53
2.3.1.6.	Proof (Prova)	53
2.3.1.7.	Trust (Confiança)	54
2.3.2.	<i>A Nova Arquitetura Proposta para a Web semântica</i>	54
2.3.3.	<i>Ontologias e a Representação do Conhecimento</i>	55
2.3.3.1.	Definição de Ontologias	55
2.3.3.2.	Classificação das Ontologias	56
2.3.3.3.	OWL	57
2.3.3.4.	Considerações sobre o uso de Ontologias	59
2.3.4.	<i>Aplicação de Tecnologias Semânticas no Domínio de BI</i>	59
2.4.	PLATAFORMA SBI	60
2.4.1.	<i>As Ontologias SBI</i>	61
2.4.2.	<i>Módulos funcionais do SBI</i>	67
2.4.3.	<i>Reasoning</i>	68
2.4.4.	<i>Considerações sobre o uso das ontologias da plataforma SBI</i>	68
2.5.	INTEGRAÇÃO E MAPEAMENTO DE ONTOLOGIAS	69
3.	DESENVOLVIMENTO DO TRABALHO	71
3.1.	ONTOLOGIAS UTILIZADAS	71
3.2.	ETAPAS NECESSÁRIAS NO MAPEAMENTO DAS ONTOLOGIAS	74
3.3.	FERRAMENTA PROPOSTA	76
3.3.1.	<i>Arquitetura</i>	77
3.3.1.1.	Servidor ("Back-end")	77
3.3.1.2.	Cliente ("Front-end")	79
3.3.1.3.	Fluxo de execução do processo de mapeamento	80
3.3.2.	<i>Implementação do Protótipo</i>	82
3.3.2.1.	Descrição dos componentes, ferramentas e tecnologias	83
3.3.2.2.	Considerações sobre os componentes, ferramentas e tecnologias	85
3.3.2.3.	Considerações gerais sobre o desenvolvimento do protótipo	86
3.4.	COMPARAÇÃO COM INICIATIVAS ACADÊMICAS E SOLUÇÕES COMERCIAIS	87
3.4.1.	<i>Protégé</i>	87
3.4.2.	<i>WebProtégé</i>	90
4.	APLICAÇÃO DO PROTÓTIPO	91
4.1.	BASE DE DADOS UTILIZADA	91
4.2.	FERRAMENTA ANALÍTICA UTILIZADA	92
4.3.	MANIPULAÇÃO DAS ONTOLOGIAS UTILIZANDO O PROTÓTIPO	93
4.3.1.	<i>Manipulação da Ontologia ISO</i>	93
4.3.2.	<i>Manipulação da Ontologia de Análise</i>	96

4.3.3. Manipulação da Ontologia de Domínio	102
4.4. FERRAMENTA ANALÍTICA UTILIZANDO AS ONTOLOGIAS CONFIGURADAS COM O AUXÍLIO DO PROTÓTIPO	103
4.5. CONSIDERAÇÕES SOBRE A APLICAÇÃO DO AMBIENTE	104
5. CONSIDERAÇÕES FINAIS	104
5.1. CONTRIBUIÇÕES	104
5.2. LIMITAÇÕES E TRABALHOS FUTUROS	105
6. REFERÊNCIAS BIBLIOGRÁFICAS	106
ANEXO A - CÓDIGO-FONTE DO PROTÓTIPO	111

1. INTRODUÇÃO

O conhecimento sempre foi o coração do crescimento econômico e do aumento gradual dos níveis de bem-estar social da humanidade. A habilidade de inventar e inovar, ou seja, criar novos conhecimentos e novas idéias - que podem ser incorporadas em produtos, processos e organizações - sempre serviu de base para o desenvolvimento mundial. A “economia baseada no conhecimento”, no entanto, é um termo cunhado recentemente – seu uso significa uma mudança significativa em relação à economia dos períodos anteriores (DAVID e FORAY, 2003). Essas mudanças, em estágio avançado nos países industrializados, constituem uma tendência dominante mesmo para economias menos industrializadas e definem um novo paradigma, que expressa a essência da presente transformação tecnológica em suas relações com a economia e a sociedade (WERTHEIN, 2000).

Segundo David e Foray (2003), essas transformações podem ser observadas de diferentes perspectivas:

- a) **Aceleração na produção do conhecimento:** está relacionada à velocidade acelerada e sem precedentes com a qual o conhecimento é criado, acumulado e, mais provavelmente, depreciado em termos de valor e relevância econômica.
- b) **O crescimento do capital intangível no nível de macroeconomia:** nos dias de hoje, a disparidade na produtividade e crescimento dos diferentes países está menos relacionada com a quantidade de recursos naturais do que com a capacidade de aperfeiçoar a qualidade do capital humano e dos fatores de produção – em outras palavras, gerar novos conhecimentos e incorporá-los em equipamentos e pessoas.
- c) **A inovação está se tornando uma atividade dominante e suas fontes ainda mais variadas:** nota-se um crescimento intensivo nas inovações, fato não só demonstrado pelo maior número de patentes requisitadas e aprovadas, mas também pela proliferação de novas variedades de bens e serviços que sinalizam as tendências da customização em massa. É importante salientar que a pesquisa formal permanece sendo a pedra fundamental na produção de conhecimento em muitos setores; contudo, os sistemas produtores de

conhecimento estão se tornando amplamente distribuídos, envolvendo muitos novos lugares e atores – pessoas “amadoras” estão se tornando produtoras de conhecimento científico.

- d) **A revolução dos instrumentos de conhecimento:** está relacionado à grande revolução tecnológica que está se estabelecendo desde a entrada na era digital, envolvendo tecnologias para produção e disseminação do conhecimento e tendo como grandes destaques o advento da internet e da Web – possibilitando o acesso remoto à informação e a meios de adquirir conhecimento – como bibliotecas digitais, ambientes de experimentação remota e de tele-educação, dentre outros.

Dentre os instrumentos que se propõem a criar e distribuir conhecimento dentro das organizações estão as soluções de *Business Intelligence*. Essas soluções visam oferecer os meios necessários para a transformação de dados em informação e para a sua divulgação através de ferramentas analíticas, a fim de gerar conhecimento, suportando o processo decisório nas organizações. Contudo, as funcionalidades exploratórias oferecidas por essas aplicações – pelo menos na primeira e atual geração de BI - limitam-se normalmente a um conjunto de funcionalidades genéricas de navegação e filtragem dos dados disponíveis (INMON, 2005; KIMBALL et al, 2008). Para próxima geração de BI - chamada de BI 2.0 - novos conceitos estão sendo introduzidos, como a utilização da semântica do negócio (HAJNYSZ; RADEN, 2007).

Segundo Sell (2006), com as soluções de BI tradicionais verificam-se desafios que permanecem não resolvidos no contexto da sociedade do conhecimento. Entre as principais deficiências das soluções atuais verificam-se a falta de perspectiva de utilização da semântica do negócio (i.e. sua terminologia e regras) no apoio ao processamento analítico e a falta de flexibilidade para a extensão das funcionalidades exploratórias de acordo com as especificidades de cada organização. Sell propôs então uma arquitetura para BI baseada em um novo paradigma que prevê a utilização intensiva do conhecimento do negócio para guiar o processamento analítico. Esta arquitetura serviu como base para o desenvolvimento do presente trabalho.

1.1. Caracterização do problema

De acordo com Marshall (2008), os analistas do grupo Gartner prevêem um crescimento lento do mercado de BI em 2008. Eles esperam que o BI tenha uma taxa de crescimento de 12,5% e que movimente em torno de 7 bilhões de dólares neste ano. Para eles, esse crescimento - relativamente mais lento que nos anos anteriores - é devido à maturidade desse mercado, que está diretamente relacionada à falta de diferenciação entre as soluções de BI.

Muitas organizações já constataram a complexidade em se traduzir os dados providos por essas soluções de BI em conhecimento e resultados positivos para o negócio. Entre as causas enumeradas pelos institutos de pesquisa encontra-se a falta de aderência das soluções implantadas com os requisitos analíticos das organizações. Isso ocorre porque as soluções de BI foram concebidas para facilitar a integração de dados e o processamento rápido de consultas, mas subjugando os valores e as necessidades de conhecimento específicas de cada organização (SELL, 2006).

Segundo Pinheiro (2008), a criação de um ambiente de inteligência de negócios, ou inteligência analítica, tem como principal objetivo criar uma infraestrutura tecnológica capaz de atender às necessidades de negócio das empresas. Assim, a orientação para os desenvolvimentos no ambiente de BI deve sempre partir da área de negócio, que demanda necessidades de visões e contextos relacionados com assuntos que auxiliem o processo de tomada de decisão. Para Sell (2006), as organizações modernas, ante os desafios e as oportunidades da era do conhecimento, almejam soluções que permitam a utilização da terminologia e das regras do seu negócio de forma efetiva no processamento analítico. Nesse contexto, uma das maiores deficiências das soluções atuais de BI é a impossibilidade de utilização da semântica do negócio para apoiar a localização, a seleção e a transformação de informação para formar e divulgar conhecimento aos seus colaboradores e parceiros estratégicos. Atualmente, algumas empresas já possuem um repositório de conhecimento constituído e agora buscam maneiras de extraírem todo o potencial desses repositórios em busca de diferencial competitivo.

Na tese de Sell (2006) é oferecido um novo paradigma para as arquiteturas de BI, com poder analítico baseado na semântica e no conhecimento do negócio da organização, através do uso de ontologias e inferências semânticas. Segundo Sell et al (2008), a manutenção dessas ontologias, utilizadas por uma plataforma baseada nessa arquitetura, poderia ser realizada por um engenheiro de conhecimento, um papel que seria criado dentro da equipe de BI. Esse papel exigiria novas competências, como o conhecimento de ferramentas e técnicas de aquisição e representação de conhecimento. Nas abordagens tradicionais, o conhecimento do negócio é obtido pelos engenheiros de BI que, por sua vez, definem o modelo do DW e as rotinas de ETL de acordo com as suas interpretações dos requisitos do negócio. Na nova abordagem proposta, o conhecimento do negócio também representado nas ontologias e integrado nos módulos do BI. Para atingir esse objetivo, o engenheiro de conhecimento precisaria trabalhar mais perto dos analistas de negócio e do arquiteto de BI.

Uma das possibilidades vislumbradas para auxiliar na aplicação prática dessa arquitetura - detalhada em uma seção posterior do presente trabalho - é a construção de um ambiente intuitivo e de fácil utilização para a realização do mapeamento e configuração, de forma semi-automática, das diferentes ontologias utilizadas. Essa ferramenta poderia facilitar a manutenção dessas ontologias pelo engenheiro de conhecimento ou até viabilizar a realização dessa função diretamente pelo próprio analista de negócios da organização - já que se espera deixar esse processo o mais simples possível, não requerendo conhecimentos computacionais avançados.

Embora existam ferramentas que possibilitem a manipulação dessas ontologias, como o Protégé (STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH, 2008), elas costumam ser genéricas e complexas demais, exigindo do usuário um bom conhecimento de engenharia de ontologias e da própria ferramenta, além de tornarem esse processo mais lento e trabalhoso.

1.2. Objetivos

A seguir serão apresentados o objetivo geral e os objetivos específicos dessa proposta de trabalho de conclusão de curso.

1.2.1. Objetivo geral

Construir um ambiente para manutenção de ontologias para apoiar a aplicação de uma solução de business intelligence semântico.

1.2.2. Objetivos específicos

- Identificar as etapas necessárias na manutenção de ontologias para uso em uma solução de BI semântico.
- Possibilitar o mapeamento semi-automático dos recursos de diferentes tipos de repositório de dados com os conceitos de uma ontologia de fontes de informação, para suportar a criação e configuração de instâncias dessa ontologia (descrição semântica padrão capaz de representar variados tipos de repositório, estruturados ou semi-estruturados).
- Possibilitar o mapeamento semi-automático das instâncias da ontologia de fontes de informação com os conceitos de uma ontologia analítica, para suportar a criação e configuração de instâncias dessa última ontologia - que podem ser utilizadas como suporte a análises semânticas na área de Business Intelligence.
- Possibilitar o mapeamento das instâncias da ontologia de fontes de informação com os conceitos de ontologias de domínio de aplicação, possibilitando ao usuário final navegar nas informações através da taxonomia de seu negócio.

1.3. Delimitação do escopo

Para o propósito inicial deste trabalho, para a realização do mapeamento semi-automático dos recursos de diferentes repositórios de dados com os conceitos da ontologia de fontes de informação, será suportado apenas o uso de um repositório estruturado, do tipo relacional. Ainda para essa etapa, serão suportados somente mapeamentos diretos entre os diferentes recursos, ou seja, do tipo “um-para-um”.

Em futuras versões da ferramenta, além do suporte ao uso conjunto de múltiplos repositórios – estruturados e semi-estruturados - estão previstos mapeamentos mais complexos, como “muitos-para-um” ou “um-para-muitos”. Assim, estão previstas operações como agregação de mais de um recurso dos repositórios de dados e o mapeamento desse agregado com apenas um conceito da ontologia de fontes de informação, ou ainda cortes (filtros) sobre um único recurso e o mapeamento dos mesmos em diferentes conceitos da ontologia.

Com relação ao mapeamento das instâncias da ontologia de fontes de informação com os conceitos de ontologias de domínio de aplicação, será suportada somente uma ontologia de domínio. A utilização de mais de uma ontologia de domínio simultaneamente, assim como a possibilidade de realizar um mapeamento entre os conceitos dessas ontologias, estão também previstos para futuras versões da ferramenta.

1.4. Justificativa

Através da ferramenta proposta, o usuário final - possivelmente um analista de negócios ou engenheiro do conhecimento - poderá manipular as diferentes ontologias de uma plataforma de business intelligence semântico, utilizando recursos visuais, de uma maneira simples e intuitiva, sem necessidade de conhecimentos computacionais avançados. Dessa forma, serão facilitadas a implantação e manutenção da plataforma nas organizações.

1.5. Metodologia

Para a realização do presente trabalho, estipulam-se as seguintes etapas a serem realizadas:

1. levantamento na literatura sobre as informações e tecnologias disponíveis relacionadas ao tema proposto;
2. adoção de uma plataforma baseada na arquitetura SBI, para apoiar o uso do ambiente para manutenção de ontologias;
3. definição das ontologias que serão utilizadas para representar as fontes de informação e os elementos analíticos de BI;
4. definição das etapas necessárias para a utilização e manutenção dessas ontologias;
5. definição de uma arquitetura para especificar o ambiente que será desenvolvido para a manutenção dessas ontologias;
6. definição de tecnologias e ferramentas serão utilizadas na construção do ambiente proposto – possibilitando a aplicação da arquitetura definida e o suporte em algumas operações necessárias, como o acesso às fontes de informação (com a extração e carga de dados) e o auxílio na manipulação e persistência das ontologias.
7. construção de um protótipo funcional e aplicação do mesmo em uma situação real – utilizando uma base de dados real e uma ferramenta OLAP compatível com o framework utilizado (possibilitando a realização de buscas semânticas sobre os dados mapeados nas ontologias).
8. comparação do ambiente desenvolvido com outras soluções acadêmicas e comerciais que poderiam ser utilizadas com o mesmo propósito.

1.6. Organização do Trabalho

O presente capítulo faz uma contextualização do assunto relacionado a este trabalho e apresenta a problemática que serviu de subsídio à pesquisa realizada, bem como os objetivos estabelecidos, a delimitação do escopo, as justificativas e a metodologia empregada.

No capítulo dois, é apresentada uma fundamentação teórica sobre os principais conceitos envolvidos neste estudo, dentre eles: gestão do conhecimento nas organizações, *business Intelligence*, tecnologias semânticas, arquitetura SBI e mapeamento e integração de ontologias.

O capítulo três especifica a proposta deste trabalho, apresentando uma possível solução para alguns dos problemas caracterizados anteriormente. Dessa forma, são definidas as ontologias que foram utilizadas - tendo-se como base uma plataforma que apoiou a construção do ambiente aqui apresentado. Além disso, são definidas as etapas necessárias para a utilização e manutenção dessas ontologias e a arquitetura que especifica o ambiente que será desenvolvido com esta finalidade, bem como as tecnologias e ferramentas utilizadas na construção do mesmo. Por fim, são feitas comparações com algumas ferramentas comerciais que poderiam ser utilizadas para o mesmo propósito.

No capítulo quatro, é feita uma validação da ferramenta proposta no capítulo anterior, através da apresentação de um protótipo funcional aplicado na simulação de uma situação real. Para essa simulação, estarão disponíveis uma base de dados relacional e uma ferramenta analítica, compatível com a plataforma adotada no presente trabalho. Essa ferramenta deverá fazer pesquisas sobre a base disponibilizada, utilizando as ontologias configuradas pelo ambiente aqui proposto.

Já no quinto e último capítulo, são apresentadas algumas considerações finais e alguns possíveis trabalhos que poderiam ser desenvolvidos a partir da proposta aqui apresentada.

2. Fundamentação Teórica

Este capítulo oferece o embasamento teórico para a compreensão dos assuntos relacionados a este trabalho. A seguir, é feita uma explanação geral sobre a gestão do conhecimento nas organizações. Na seqüência, sucedem-se as seções que abordam os conceitos relacionados ao *business intelligence* tradicional, às tecnologias semânticas disponíveis, à plataforma SBI (*Semantic Business Intelligence*) e a estudos realizados na área de mapeamento e integração de ontologias.

2.1. Gestão do Conhecimento nas Organizações

A produção de conhecimento constitui-se num processo dialético de confrontação permanente de idéias e conceitos e de construção de novas sínteses de conhecimento. Os conhecimentos tácitos e explícitos, conjugados com idéias, valores e emoções, possibilitam a visualização de novas realidades e cenários que permitem a concepção de maneiras diferentes de se interpretar e de se fazer as coisas, gerando contradições e interrogações que conduzem a novos conhecimentos (CARVALHO; MENDES; VERAS, 2006).

Assim como o próprio termo “conhecimento”, é difícil definir “gestão do conhecimento”. Do ponto de vista mercadológico, o conhecimento poderia ser visto como sendo a resposta para os desafios competitivos encontrados pelas empresas nos dias de hoje. Dessa forma, gestão do conhecimento incluiria os sistemas de informações e de criação de conhecimento, assim como a gestão estratégica e a inovação (LLORIA, 2008).

Segundo Lloria (2008), nos recentes anos, a criação e gestão do conhecimento tem sido uma das questões que tem atraído mais interesse, não apenas no campo acadêmico, mas também no mundo dos negócios.

2.1.1. Dos dados à experiência organizacional

Segundo Pinheiro (2008), com o aumento da capacidade de armazenamento por parte das empresas e com a crescente automação dos processos por meio de mecanismos sistêmicos, o volume de informações disponíveis está cada vez maior. Contudo, os dados operacionais provenientes dos processos transacionais das empresas contribuem pouco para a tomada de decisão – podendo tornar o processo decisório confuso e volátil.

Para que os dados gerados pelos processos operacionais possam ser utilizados de forma estratégica pelas corporações, se tornando subsídio para o processo decisório, é fundamental que exista uma transformação natural em seu conteúdo e forma. Esses dados operacionais devem ser transformados em **informações**, se tornando persistentes em um ambiente adequado de coleta, armazenamento e publicação. Essas informações - que podem se apresentar na forma de indicadores de desempenho empresarial - permitem que uma empresa possa reconhecer suas necessidades e fraquezas, assim como seus pontos fortes, tornando as ações estratégicas mais substanciais e eficientes. O conjunto de indicadores analíticos permite uma visão ampla sobre determinado cenário, mas não fornecem conhecimento necessário para uma tomada de decisão diferenciada.

Dessa forma, é necessária a transformação dessas informações em conhecimento, por meio de um processo de modelagem analítica, envolvendo modelos de agrupamento e preditivos, no qual padrões e comportamentos são reconhecidos e identificados de forma automática, bem como previsões podem ser realizadas de maneira consistente. Esses modelos dão origem a indicadores que identificam grupos com características semelhantes, indicam as probabilidades de ocorrência de determinados eventos, otimizam determinado cenário de negócio, dentre outras possibilidades - ou seja, propiciam a geração de **conhecimento** sobre oportunidades e ameaças relacionados ao negócio da empresa.

Por conseguinte, esse conhecimento deve ser transformado em ações factíveis e exequíveis, fazendo com que o fluxo de conteúdo, desde os dados

operacionais, passando pelas informações analíticas e pelo conhecimento inferido, se torne **inteligência** na prática. Isto significa dizer que o conhecimento não aplicado não gera inteligência e, por consequência, não trás benefícios tangíveis para as corporações.

E por fim, tem-se a **experiência** - a persistência de todo conhecimento e inteligência gerados na corporação, permitindo não apenas a disseminação de capital intelectual pela empresa, mas também a reutilização e aprimoramento dos mesmos em um ambiente competitivo, o qual demanda velocidade e precisão nas ações de negócio.

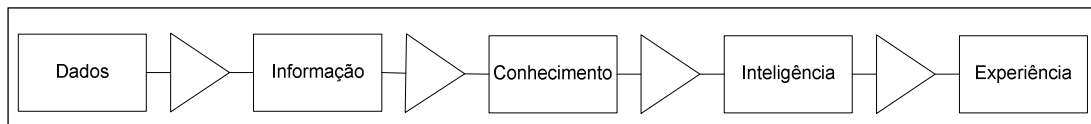


Figura 1 - Evolução dos dados até a experiência

(ADAPTADO DE: Pinheiro, 2008)

2.1.2. Por que gerir o conhecimento nas organizações?

As tradicionais visões de negócio e estruturas organizacionais vêm sofrendo diversas mudanças ao longo do tempo, sobretudo devido às pressões de negócios e novos mercados nas economias mundiais. A concorrência e a globalização alteraram o quadro geral dessas visões tradicionais. A perspectiva da informação tornou-se cada vez mais importante, sendo um bem tangível para as corporações atuais e, de forma natural, o conhecimento passou a ser primordial para o sucesso competitivo. Assim, se uma organização deseja crescer e tornar-se mais rentável, deve saber gerenciar e explorar esse bem de forma produtiva, criando a partir do conhecimento gerado, inteligência competitiva como um diferencial de negócios (PINHEIRO, 2008).

Na atual era do conhecimento, sua busca alavanca o modo de pensar nas organizações e influencia a redefinição de conceitos e métodos de agregação de

valor. De fundamental importância, o conhecimento é especialmente útil nas organizações para o desenvolvimento de estratégias que visam propiciar o cumprimento de sua missão. Permeado nas pessoas, nos processos, nos sistemas, nos produtos e serviços, nos clientes, fornecedores, parceiros, o conhecimento é o elemento que mantém a vitalidade de uma organização que precisa sempre recriar novas oportunidades e perspectivas (CARVALHO; MENDES; VERAS, 2006).

A gestão do conhecimento implica em uma série de políticas e diretrizes que possibilitam a criação, difusão e institucionalização do conhecimento, visando alcançar os objetivos da empresa (LLORIA, 2008). Com isso deve ser focado fundamentalmente o processo de criação de uma estrutura capaz de relacionar o capital, os recursos humanos, os sistemas computacionais, as culturas regionais e globais e o conhecimento sobre as diversas entidades relativas ao negócio de uma corporação. Adicionalmente, um processo de gestão do conhecimento deve poder manipular e gerenciar todos esses relacionamentos mencionados anteriormente de forma efetiva e eficiente (PINHEIRO, 2008). Por causa dessa complexidade inerente ao processo de gestão do conhecimento, a tecnologia da informação deve exercer um papel fundamental no seu desenvolvimento.

2.1.3. Gestão do conhecimento e a tecnologia da Informação

Segundo Cody et al (2002), os executivos das organizações entendem que o conhecimento preciso e no tempo certo pode significar um aumento da performance dos negócios. Para ele, algumas tecnologias têm sido centrais no melhoramento qualitativo e quantitativo dos conhecimentos disponíveis para os tomadores de decisão – dentre elas estão as soluções de business intelligence e as soluções de gestão do conhecimento. Business intelligence tem utilizado as funcionalidades, a escalabilidade e a confiança dos SGBDs modernos para construir data warehouses cada vez maiores e utilizar técnicas de data mining para extrair vantagens competitivas da vasta quantidade de dados organizacionais disponíveis. Já as tecnologias de gestão do conhecimento, mesmo que ainda menos maduras que as tecnologias de BI, são capazes de

combinar os sistemas de gerenciamento de conteúdo e a Web com as capacidades amplamente desenvolvidas de busca e mineração de texto para produzir mais valor através da enorme quantidade de informação textual.

De acordo com Pinheiro (2008), no aspecto tecnológico, existem cinco frentes que servem de base para o gerenciamento do conhecimento dentro de uma organização: business Intelligence, colaboração, transferência de conhecimento, descoberta de conhecimento e locação da experiência. O contexto de Business Intelligence é contemplado através de um ambiente de data warehouse, através de consultas OLAP, processos de data mining e outras técnicas para analisar dados estruturados.

2.2. Business Intelligence

Segundo Pinheiro (2008), os objetivos de um ambiente de Business Intelligence estão presentes desde o início da utilização dos sistemas computacionais: auxiliar as organizações a controlar e otimizar melhor seus processos. Porém, os sistemas de banco de dados e as diversas aplicações que manipulam os dados de uma determinada organização, otimizam e controlam tais processos com uma granularidade pequena, ou seja, com o foco totalmente voltado para as transações referentes aos processos operacionais e aos usuários que controlam esses processos. Dessa forma, uma classe de usuários não é atendida adequadamente por um ambiente desse tipo - esta classe é composta pelos usuários que são responsáveis pelas tomadas de decisões nas corporações, estabelecendo estratégias de negócio, planos de marketing, campanhas promocionais, etc. Para superar essa barreira, foram idealizados e projetados os ambientes de “data warehouse”, no qual o foco é o armazenamento das informações de uma determinada organização pertinentes ao processo de tomada de decisões.

O conceito de Business Intelligence vai ainda um pouco além disso - um ambiente desse tipo irá possibilitar explorações analíticas e gerenciais por parte de usuários de negócio, no sentido de aperfeiçoar e otimizar o processo de tomada de decisão. A criação de um ambiente de inteligência de negócios, ou inteligência analítica, tem como principal objetivo criar uma infra-estrutura tecnológica capaz de atender às necessidades de negócio das empresas. Entende-se que a orientação para os desenvolvimentos no ambiente de BI deva sempre partir da área de negócio, que demanda necessidades de visões e contextos relacionados com assuntos que auxiliem o processo de tomada de decisão (PINHEIRO, 2008). Essa é também a visão de Kimball et al. (2008), que afirma que os requisitos de negócio devem sempre orientar a concepção de uma arquitetura de BI.

2.2.1. Arquitetura de BI

Para Kimball et al. (2008), uma arquitetura de Business Intelligence deve descrever o fluxo de dados dos sistemas-fonte até os tomadores de decisão e as transformações e armazenamentos dos dados que ocorrem longo desse caminho. Ela também deve especificar as ferramentas, técnicas, serviços ou plataformas necessários para fazer com que esse fluxo ocorra.

A figura abaixo apresenta em alto-nível uma arquitetura técnica de BI, mostrando uma natural separação da mesma em duas partes: o “Back-Room” (também chamado de “Back-End”) e o “Front-Room” (também chamado de “Front-End”).

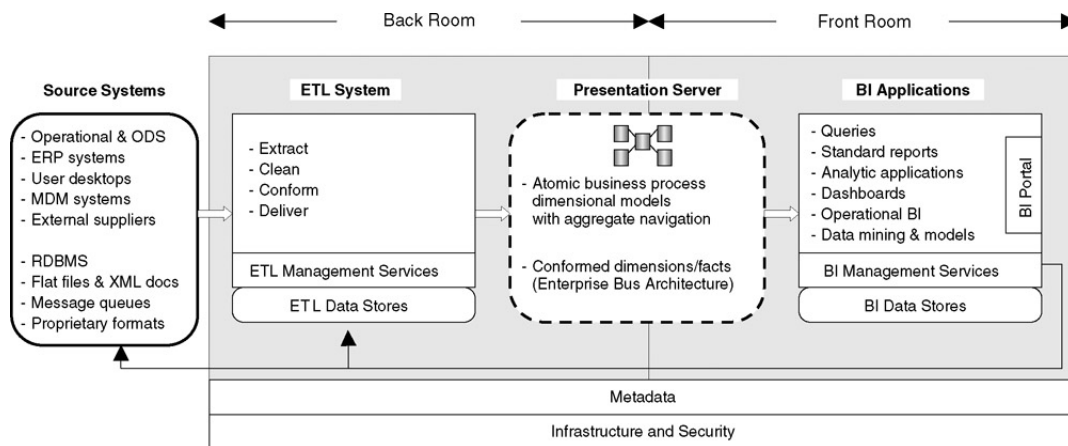


Figura 2 - Arquitetura de DW/BI

(FONTE: KIMBALL et al., 2008)

Na camada de “Back-Room”, do lado esquerdo, pode-se observar a movimentação dos dados oriundos dos sistemas-fonte através do processo de ETL (visto em uma seção posterior). Depois dos dados limpos e alinhados, o processo de ETL seleciona, agrega e reestrutura os dados dentro dos modelos dimensionais dos processos de negócio (também chamados de “data marts”). Estes são carregados em servidores de apresentação, sendo unidos através de dimensões ou fatos em comum ou em conformidade - utilizando a arquitetura

“enterprise bus”. Já a camada de “Front-Room”, no lado direito, mostra como os usuários podem acessar os dados disponíveis nos servidores de apresentação através de variadas aplicações e ferramentas de BI (KIMBALL et al., 2008).

Os servidores de apresentação são definidos por Kimball et al. (2008) como sendo as plataformas de bases de dados onde os dados são armazenados para consulta direta pelos usuários do negócio, por sistemas de relatórios, ou por outras aplicações de BI. Além disso, esses autores utilizam o termo “data warehouse empresarial” como sendo a estrutura que, incluindo um ou mais servidores de apresentação, fornece acesso aos dados dos diferentes processos de negócio de uma maneira atômica ou sumarizada, servindo como uma fonte única para todos os dados analíticos da organização. Os data warehouses e alguns dos processos a eles associados serão vistos com mais detalhes a seguir.

2.2.2. Data Warehouse

Segundo Pinheiro (2008), o termo data warehouse foi utilizado pela primeira vez por W. H. Inmon para descrever um banco de dados projetado e construído para auxiliar as empresas no processo de tomada de decisões. Para o primeiro autor, data warehouse pode ser definido como um processo de organização dos dados de forma a criar novos conhecimentos de negócios e permitir novos *insights* no processo decisório. Para Inmon (2005), data warehouse é uma coleção de bases de dados integradas e orientadas por assunto, projetadas para apoiar a função dos sistemas de apoio à decisão, onde cada unidade dos dados é relevante em algum momento no tempo. Além disso, o data warehouse pode conter dados atômicos e dados ligeiramente sumarizados.

Segundo Pinheiro (2008), não se deve entender ou visualizar um data warehouse como simplesmente um sistema de banco de dados ou uma ferramenta estática e isolada de auxílio no processo de tomada de decisões, mas sim um conjunto de tecnologias de suporte à decisão, um processo contínuo e integrado que objetiva subsidiar os analistas e executivos de negócios nas tomadas de decisão.

Essa é também a visão estabelecida por Kimball et al. (2008), que preferem criar uma dependência entre o sistema de data warehouse e o de business intelligence como um todo, relacionando e utilizando os dois acrônimos em conjunto, e utilizando o termo “DW/BI” (de uma forma independente, fazem referência ao repositório unificado de dados como “data warehouse empresarial” e aos instrumentos analíticos como “aplicações BI”). Dessa forma, eles afirmam que embora alguns possam argumentar que se pode, teoricamente, fazer BI sem um data warehouse, e vice-versa, isto não está de acordo o ponto-de-vista deles. Data warehouse é uma plataforma necessária para business intelligence - ele faz o trabalho pesado de extrair os dados dos sistemas-fonte, limpá-los e organizá-los de uma forma que os usuários comuns com foco em negócio possam entendê-los. Em contrapartida, um data warehouse sem uma solução integrada de business intelligence tende a falhar completamente. Para eles, essa mudança de foco coloca a iniciativa nas mãos dos usuários do negócio, e não mais no departamento de TI.

Em um ambiente de data warehouse, os dados são extraídos dos sistemas transacionais da empresa, adequados e carregados em um repositório corporativo, para que possam ser geradas as diversas visões analíticas de informações para apoio à tomada de decisão. Para se referenciar a esse processo de extração, transformação e carga dos dados, utiliza-se freqüentemente o acrônimo ETL.

2.2.3. Processo de ETL

Para Kimball et al. (2008), um sistema de extração, transformação e carga consiste em um grupo de processos através dos quais os dados operacionais fonte são preparados para o data warehouse. Consiste em extrair dados operacionais de aplicações fonte, limpá-los, adequá-los e entregá-los para os servidores de apresentação. Já Inmon (2005) define o ETL como sendo o processo de pegar os dados de aplicações legadas e integrá-los dentro do data warehouse.

Apesar do acrônimo ETL popularizar a divisão desse tipo de processo em três etapas (“Extraction, Transformation and Loading” - ou “Extração, Transformação e Carga”), segundo Kimball et al. (2008), um sistema de ETL pode ser dividido em quatro grandes componentes:

- I. **Extração:** recolhimento dos dados brutos dos sistemas-fonte (algumas vezes sistemas legados) e, na maioria das vezes, gravação em um disco dentro do ambiente do ETL (ou área de estagiamento) antes de fazer qualquer reestruturação significativa nesses dados.
- II. **Limpeza e Adequação:** estabelecimento de uma série de passos de processamento sobre os dados-fonte, para melhorar a qualidade dos dados recolhidos, e junção dos dados de duas ou mais fontes para criação de dimensões e métricas adequadas.
- III. **Entrega:** Estruturação e carga física dos dados dentro dos modelos dimensionais desejados do servidor de apresentação.
- IV. **Gerenciamento:** gestão de forma coerente dos processos e sistemas relacionados ao ambiente ETL.

Estima-se que um sistema de ETL consome em geral 70% do tempo e esforço na construção de um ambiente de DW/BI. Construir um sistema ETL é um sempre um desafio, pois muitas forças externas influenciam em seu projeto: os requisitos do negócio, as fontes de dados, o orçamento, as capacidades de processamento disponíveis e as habilidades da equipe disponível. Um sistema ETL é geralmente complexo e demanda muitos recursos das organizações; contudo, ele é o alicerce para um projeto de DW/BI – seu sucesso é fundamental para o sucesso do data warehouse (KIMBALL et al., 2008).

A finalidade de um processo de ETL, dentro de uma arquitetura de DW/BI, é de entregar os dados extraídos e transformados para um repositório adequado, que propicie o melhor suporte possível para consultas analíticas, favorecendo o processo decisório. Esse repositório deve ser modelado de forma que possibilite visões multidimensionais das informações e alta performance nas consultas.

2.2.4. Modelagem Dimensional

Conforme já mencionado, uma das grandes diferenças entre os sistemas de bancos de dados operacionais e os data warehouses é a forma como os dados são armazenados - o que reflete diretamente o processo de modelagem de dados. Segundo Kimball et al (2008), o tradicional modelo entidade-relacionamento (MER), não é adequado para a construção de um data warehouse - o baixo desempenho em consultas e a falta de uma navegabilidade adequada entre as tabelas para a apresentação das informações são alguns dos motivos citados. Para eles, a técnica mais viável (e amplamente aceita) para disponibilização de dados em soluções de business intelligence é a modelagem dimensional.

Segundo Kimball et al (2008), um modelo dimensional é um modelo de dados estruturado para atingir a máxima performance em consultas e facilidade de uso. Ele divide o mundo em duas partes: medidas e contexto. As medidas são capturadas pelos processos da organização e pelos sistemas operacionais que dão suporte a eles - elas são geralmente valores numéricos e são chamadas também de “fatos”. Esses fatos são envolvidos por um amplo contexto textual no momento em que eles são armazenados – esse contexto é intuitivamente dividido em grupos lógicos chamados “dimensões”. As dimensões descrevem contextos como: “quem?”, “o que?”, “quando?”, “onde?”, “por que?” e “como?”. No ambiente dos SGBD relacionais, uma tabela de fato é baseada em um evento de medição, geralmente com um registro para cada medição distinta. Essa tabela de fato possui uma chave composta, fazendo junção com tabelas dimensionais, cada uma com uma única chave primária, que descrevem precisamente o que é conhecido dentro do contexto de cada registro de medição.

Cada processo de negócio de uma organização pode ser representado por um modelo dimensional, que consiste em uma tabela de fato, contendo as medidas numéricas, envolvida por um conjunto de tabelas dimensionais. Devido a essa estrutura característica de um modelo dimensional, em que a tabela de fato é rodeada por tabelas de dimensão, ele é geralmente chamado de esquema-estrela (KIMBALL et al, 2008). Para Inmom (2005), um esquema estrela é a base

para o projeto multidimensional de um “data mart” – que ele define como sendo uma estrutura de dados departamentalizada que compõe o data warehouse, onde os dados são denormalizados baseando-se nas necessidades de informação do departamento. Kimball et al. (2008) relatam que, embora também utilizasse freqüentemente o termo “data mart”, não mais o fazem, pois esse termo está caindo em desuso por representar conjuntos de dados sumarizados por departamento, independentes e não integrados com o restante da arquitetura – por isso eles preferiram adotar o termo “business process dimensional models” (ou “modelos dimensionais dos processos de negócio”).

Nas figuras abaixo, podemos observar respectivamente um modelo dimensional de um processo de negócios relacionado aos pedidos de organização e parte de outro modelo dimensional relacionado ao processo de vendas, onde é mostrado a ligação entre uma tabela de fatos e uma dimensão relativa aos produtos associados às vendas.

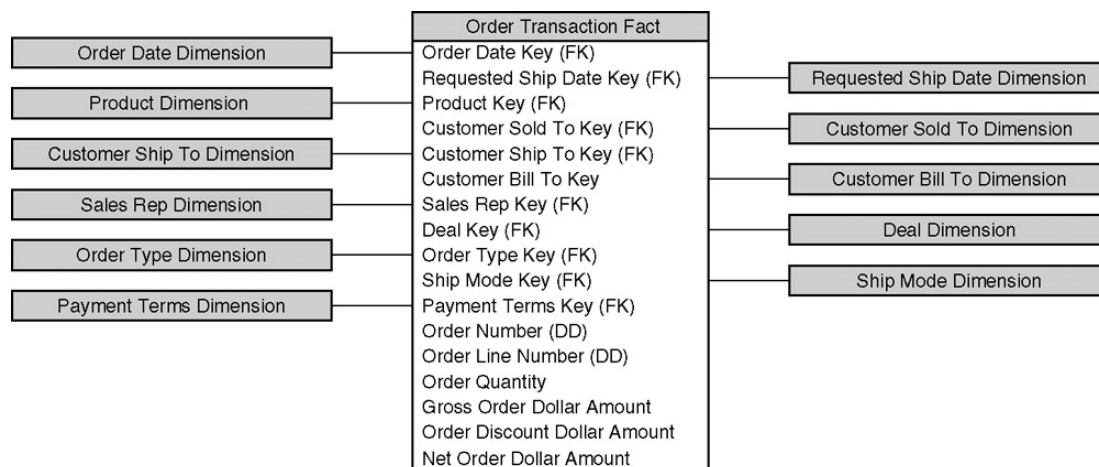


Figura 3 - Exemplo de um modelo dimensional relacionado ao processo de pedidos de uma organização

(FONTE: Kimball et al, 2008)

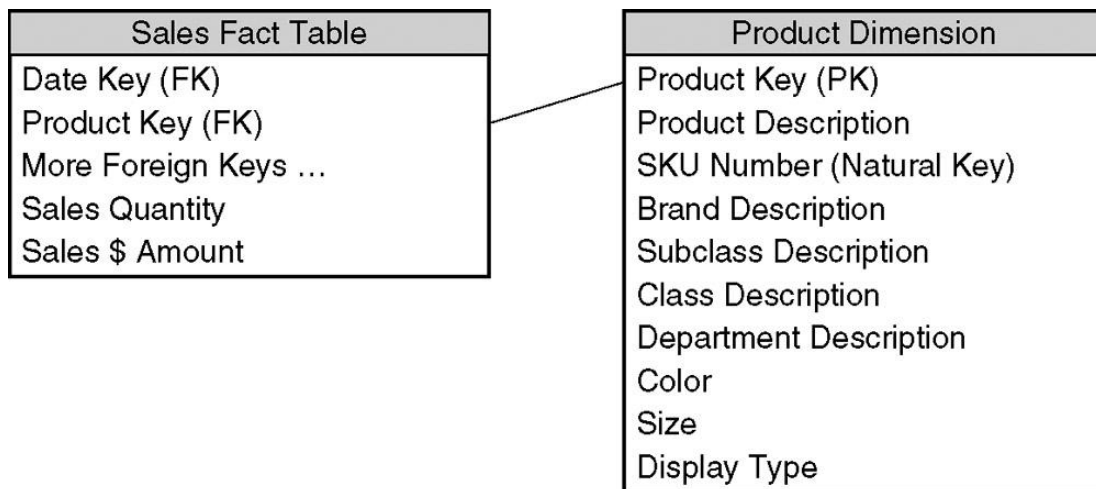


Figura 4 - Ligação entre uma tabela de fato e uma tabela dimensional relacionados a um processo de vendas
(ADAPTADO DE: Kimball et al, 2008)

Para Kimball et al (2008), modelos dimensionais demonstraram serem fáceis de entender, previsíveis, estendíveis e altamente responsivos a demandas “ad hoc” devido à sua natureza simétrica previsível. Além disso, eles são a base de muitas melhorias de performance dos SGBDs, incluindo potentes abordagens de indexação e agregação, e a base do desenvolvimento incremental distribuído de data warehouses através do uso de dimensões e fatos adequados. Os modelos dimensionais são também a fundamentação lógica de todos os sistemas OLAP.

2.2.5. OLAP

Segundo Anzanello (2002) e Pinheiro (2008), o termo OLAP foi citado pela primeira vez por E.F. Codd, quando ele definiu doze regras que estas aplicações deveriam atender. De acordo com Anzanello (2002), a visão conceitual multidimensional dos negócios de uma empresa foi umas das regras citadas, a qual se tornou a característica fundamental no desenvolvimento destas aplicações. A visão multidimensional consiste de consultas que fornecem dados a respeito de medidas de desempenho, decompostas por uma ou mais dimensões dessas medidas. Esses dados podem também ser filtrados pela dimensão e/ou pelo valor da medida.

As visões multidimensionais fornecem as técnicas básicas para as análises requeridas pelas aplicações de BI. Sua idéia básica é que os gestores possam manipular visões de dados através de muitas dimensões, para que possam entender as mudanças que estão ocorrendo nas suas organizações.

Segundo Pinheiro (2008), a melhor forma de entender um ambiente OLAP (On Line Analytical Processing), proposto através da implementação de um data warehouse, é fazendo uma comparação direta com seu antecessor, o ambiente OLTP (On Line Transaction Processing). Desse modo, segue abaixo uma tabela comparativa entre as duas abordagens:

OLTP	OLAP
<p>Controle do Processo</p> <p>Em um ambiente operacional, as aplicações de banco de dados devem controlar o processo como um todo, envolvendo as diversas etapas de uma entrada ou atualização de dados.</p>	<p>Não controla o processo</p> <p>Em um ambiente de data warehouse não existe controle do processo, pois eles são todos controlados no ambiente operacional, onde já deveriam ter sido validados e consistidos.</p>
<p>Grande volume de pequenas transações realizado diariamente</p>	<p>Pequeno volume de grandes transações realizado diariamente</p>
<p>O foco é na transação</p> <p>Como essas aplicações devem controlar os processos, as transações pertinentes a esses processos devem ser atômicas (todas as operações de banco de dados associados a uma transação devem ser efetuadas como um todo).</p>	<p>O foco é no conjunto dos dados</p> <p>Os dados oriundos do ambiente operacional podem ser sumarizados e agregados. Sendo assim, o que interessa não é o dado isolado, mas a sumarização deles.</p>
<p>Um registro por transação</p> <p>Cada transação insere, atualiza ou remove apenas um registro da base de dados por vez. Mesmo existindo diversas operações em uma mesma transação, os registros das diversas tabelas da base de dados serão manipulados isoladamente, um de</p>	<p>Milhares de registros por transação</p> <p>A carga do ambiente operacional para um DW é caracterizada por pequenas quantidades de transações com grandes volumes de dados. Assim, dentro de uma transação no ambiente de DW, que é a carga do operacional, um grande número</p>

<p>cada vez.</p>	<p>de registros é passado de um ambiente para o outro.</p>
<p>Consistência microscópica dos dados</p> <p>Devido ao conceito de atomicidade da transação, uma operação de banco de dados deve ser verificada completamente. Isto implica que todos os campos do registro que está sendo inserido, atualizado ou removido podem ser consistidos de forma isolada e, em caso de falha de apenas um deles, toda a transação pode falhar.</p>	<p>Não há consistência, apenas consultas</p> <p>A grande finalidade de um DW é proporcionar uma maneira eficaz de se realizar consultas estratégicas para o processo de tomada de decisões. Como o ambiente é apenas para consulta, as informações nele contidas não precisam ser validadas – elas já deveriam ter sido, quando estavam no ambiente operacional.</p>
<p>Escopo temporal é momentâneo</p> <p>Todas as informações que estão armazenadas no banco de dados possuem um escopo temporal limitado, ou seja, em algum momento elas não terão mais importância ou validade e, conseqüentemente, serão deslocadas para uma área de armazenamento secundária (como um “arquivo morto”, p. ex.). Um sistema de banco de dados não poderia realmente armazenar indefinidamente todas as informações que lhe são inseridas – isso poderia implicar em uma queda de performance considerável do sistema com o passar do tempo, podendo comprometer as novas operações.</p>	<p>Escopo temporal é histórico</p> <p>Conforme já mencionado, a finalidade do ambiente de DW é proporcionar um meio eficiente de consultar os dados de interesse, para auxílio no processo de tomada de decisões. Estas consultas normalmente são complexas e envolvem comparações históricas. Estas comparações, no decorrer do tempo, determinam as estratégias a serem adotadas em determinado segmento dos negócios. Portanto, o tempo é uma dimensão fundamental em um ambiente de DW e, por isso, as informações nele contidas são armazenadas por um longo período de tempo.</p>
<p>Consultas pré-concebidas</p> <p>Em um ambiente operacional, as consultas à base de dados são pré-concebidas e implementadas. Essa implementação pode demandar um tempo considerável, já que as consultas devem ser especificadas pelo usuário e, posteriormente, codificadas, implementadas e testadas por um analista, programador ou DBA. O tempo de implementação de novas consultas pode ser proibitivo com relação à expectativa de resposta do usuário.</p>	<p>Consultas “ad-hoc”</p> <p>As consultas em um ambiente de DW podem ser realizadas de forma instantânea, ou seja, na medida em que elas vão sendo necessárias, o usuário pode ir compondo as diversas dimensões do DW para efetuar as consultas desejadas. Isso é possível graças ao modelo multidimensional fornecido pelo ambiente OLAP.</p>

Ambiente estático	Ambiente dinâmico
<p>Pelo fato das consultas serem pré-concebidas ou demandarem um tempo considerável para sua implementação, o ambiente operacional torna-se um ambiente estático, ou seja, ele é uma fotografia do estado corrente da aplicação de banco de dados utilizada. Qualquer alteração nessa aplicação, ou em relatórios e consultas nela inseridos, necessitam ser implementados como um novo processo sistêmico.</p>	<p>Como as consultas podem ser montadas na hora em que se fazem necessárias, o ambiente de DW torna-se um ambiente dinâmico. Isto implica num ganho substancial de tempo no processo de tomada de decisão. Algumas análises podem até continuar sendo pré-concebidas (as mais solicitadas, por exemplo), mas sempre se terá disponível a opção das consultas “ad-hoc”.</p>

Tabela 1 - Comparação dos ambientes OLTP e OLAP
(COMPILADO DE: Pinheiro, 2008)

De uma forma geral, em um sistema OLAP, os usuários podem executar algumas operações para “navegar” nas informações, utilizando para isso as diversas dimensões disponibilizadas pelo modelo. Abaixo estão relacionadas algumas dessas operações, que foram conceituadas por Kimball et al. (2008).

- **Slice and dice:** é a habilidade em acessar igualmente um data warehouse através de qualquer uma de suas dimensões. São processos para separar ou unir os dados do data warehouse através de variadas combinações.
- **Drill down:** o ato de adicionar mais detalhamento a uma visão dos dados, freqüentemente descendo um nível de hierarquia (por exemplo, de ano para mês) ou, alternativamente, através da inclusão de atributos não-hierárquicos na lista de seleção das consultas (“SELECTs”).
- **Drill up:** é o processo contrário ao “drill down”. É o ato de sumarizar informações em uma visão dos dados, usualmente removendo atributos das consultas ou subindo um nível na hierarquia (por exemplo, de mês pra ano).
- **Drill across:** O ato de requisitar dados identicamente rotulados, oriundos de duas ou mais tabelas de fato, dentro de uma mesma visão de dados.

Esse processo só funciona se os valores dimensionais utilizados como rótulos dos registros estiverem em conformidade.

O ambiente OLAP provê a capacidade de visualização das informações a partir de muitas perspectivas diferentes, utilizando uma estrutura de dados adequada e eficiente. Através dele é possível extrair muito do potencial analítico fornecido pelos data warehouses. Em adição às consultas normais às bases de dados, essas soluções podem ainda conter poderosos algoritmos de análise (como os de mineração de dados), baseados em “expertise” de domínio.

2.2.6. Considerações sobre as soluções de DW/BI tradicionais

Segundo Kimball et al (2008), notáveis transformações ocorreram nos últimos anos. A indústria do data warehouse atingiu a sua maturidade plena e aceitação através do mundo dos negócios. Hardware e software avançaram muito durante esses anos. Começou-se a falar em “terabytes”, ao invés de “gigabytes”. Mas ainda, por alguma razão, o data warehouse permaneceu inalterado em seus fundamentos.

Para Kimball et al e Pinheiro (2008), um processo de data warehouse é um processo contínuo e dinâmico, que possui um ciclo de vida que não termina com a sua implementação - ele segue indefinidamente enquanto ocorrerem mudanças de rumo ou objetivos no negócio ou até de estratégias dentro de uma organização¹. Gestores colocam demandas inesperadas para os sistemas e novas fontes de dados se tornam disponíveis. Dessa forma, segundo Kimball et al (2008), os sistemas DW/BI precisam evoluir tão rapidamente quanto as organizações que o utilizam (organizações estáveis irão colocar demandas evolutivas modestas para o sistema; já as organizações dinâmicas e turbulentas irão tornar essa tarefa mais desafiadora). Sendo assim, é necessário o

¹ Por essa característica, costuma-se referenciar o processo de construção de um data warehouse como data warehousing (PINHEIRO, 2008).

desenvolvimento de novas técnicas, mais flexíveis e adaptáveis, para o desenvolvimento e a evolução desse tipo de sistema.

Recentemente surgiu o termo BI 2.0, que representaria uma evolução das soluções de business intelligence tradicionais, utilizando novas abordagens e tecnologias disponíveis.

2.2.7. Business Intelligence 2.0

Segundo logiXML (2007), o conceito de BI 2.0, que se refere à próxima geração de Business Intelligence, segue a idéia da Web 2.0, que se refere à próxima geração da Web. Seu foco, assim como o da Web 2.0, é nas pessoas - possibilitando aos usuários expressar a sua criatividade, permitindo que eles acessem a informação livremente, produzindo alguma coisa significativa a partir dela, ao mesmo tempo em que é focado o compartilhamento de informações, a comunicação e a colaboração. Ele representa uma coleção de abordagens tecnológicas, funcionalidades de BI e modelos de licenciamento inovadores, dinâmicos, adaptativos e colaborativos, que irá realmente levar o BI para as massas, tornando a informação disponível a qualquer hora, em qualquer lugar, para qualquer pessoa.

O objetivo do BI 2.0 é auxiliar ainda mais no processo de tomada de decisão, dando à pessoa certa exatamente a informação de que ela precisa - e essa informação tem que ser completa. Além disso, um foco muito forte do BI 2.0 é em relação à colaboração. Muitas decisões, especialmente as mais importantes - envolvendo toda a empresa e a sua estratégia geral - não são tomadas em uma fração de tempo por uma única pessoa. Essas decisões dependem de muitos fatores e pessoas diferentes dentro da empresa precisam participar delas. Assim, a idéia é melhorar a comunicação e o compartilhamento de informações através das ferramentas de BI - assim todos os envolvidos no processo decisório poderão incluir seus conhecimentos e pareceres em cada situação. Tem que haver um ambiente que propicie uma colaboração ativa e um compartilhamento de informações imediato (HAJNYSZ, 2007)

Para Hajnysz (2007), outra idéia por trás do BI 2.0 é utilizar dados de múltiplas fontes, não importando se elas estão localizadas em um servidor interno ou disponibilizadas na Web - isso implica em uma mudança fundamental em relação às abordagens tradicionais: o suporte à utilização de dados não estruturados. Além disso, um conceito um pouco mais abstrato está sendo introduzido no BI 2.0: a semântica - visto que a Web 2.0 inclui semântica, o BI 2.0 também deve seguir nessa linha. Para Raden (2007), localizar a informação adequada para a resolução de um determinado problema tem que ser um processo semântico, sem que se tenha a necessidade de conhecer estruturas de dados ou formas canônicas. A existência de uma camada semântica possibilita também a divulgação das informações constantes nos ambientes analíticos com uniformização dos conceitos e com uma linguagem muito mais próxima dos usuários, além do conteúdo muito mais completo e preciso. É aí que entram as tecnologias semânticas, dando suporte às ferramentas de BI.

2.3. Tecnologias Semânticas

Na abordagem utilizada no presente trabalho, utilizam-se tecnologias que vêm sendo desenvolvidas no contexto da Web semântica para tornar as soluções de BI mais flexíveis e alinhadas às necessidades dos tomadores de decisão. Este capítulo apresenta então uma introdução sobre as tecnologias da Web semântica.

2.3.1. A Web Semântica e suas camadas

Segundo Berners-Lee, Handler e Lassila (2001), a maioria do conteúdo da Web hoje é projetada para humanos lerem, não para programas de computadores manipulá-los expressivamente. Computadores podem “parsear” eficientemente as páginas Web para processamento de layout e de rotina - localizando cabeçalhos ou links para outras páginas, por exemplo -, mas eles não têm um modo confiável para processar a semântica presente nos documentos. A informação varia em muitos eixos e um deles é a diferença entre a informação produzida primariamente para consumo humano e a produzida principalmente para máquinas. De um lado da escala, nós temos tudo desde comerciais de TV a poesia; do outro lado, nós temos base de dados, programas e saídas de sensores. Até hoje, a Web se desenvolveu mais rapidamente como um repositório de documentos para pessoas do que para dados e informações que possam ser processados automaticamente. A Web semântica tem por objetivo completar isso.

A Web semântica, proposta por Berners-Lee, Handler e Lassila (2001), traria estrutura para o conteúdo semântico de páginas Web, criando um ambiente no qual agentes de software vagando de página a página poderiam prontamente realizar tarefas sofisticadas para os usuários. A Web semântica não seria uma Web separada, mas uma extensão da atual, na qual é dado um significado bem definido à informação, possibilitando que os computadores e as pessoas trabalhem em cooperação de uma melhor forma.

Berners-Lee, Handler e Lassila (2001) propuseram uma estrutura em camadas para a Web semântica, conforme ilustrado abaixo. Essa divisão

permitiria o desenvolvimento em etapas, em que cada camada é apoiada pelas camadas inferiores.

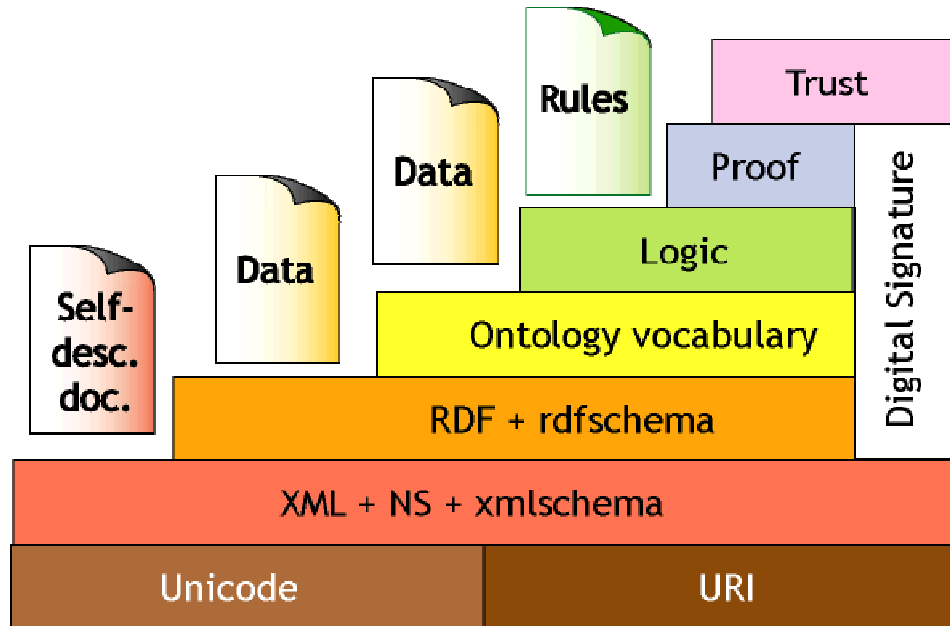


Figura 5 - A Arquitetura da Web Semântica

As camadas que fazem parte da arquitetura proposta para a Web semântica são apresentadas com mais detalhes a seguir.

2.3.1.1. Unicode e URI

Na camada inferior da arquitetura, encontra-se a referência para troca de conteúdos (Unicode), que estabelece a “forma de escrita” na Web Semântica.

Segundo UNICODE (2008), linguagens escritas são representadas por elementos textuais que são utilizados para criar palavras e sentenças. Esses elementos podem ser letras como “w” ou “M”, caracteres (como os utilizados em japonês Hiragana para representar sílabas), ou ideogramas (como os utilizados em chinês para representar palavras e conceitos). A definição de elementos de texto muda com frequência, dependendo do processo que está tratando o texto. Por exemplo, na classificação histórica do espanhol (do espanhol histórico), “ll”

conta como um único elemento de texto. No entanto, quando palavras em espanhol são digitadas, “ll” são dois elementos de texto separados: “l” e “l”. Para evitar definir o que é e o que não é um elemento de texto nos diferentes processos, o padrão Unicode utiliza o termo “elementos de codificação” (comumente chamados de caracteres). Um elemento de codificação é fundamental para o processamento computacional de textos. Para a maioria, os elementos de código correspondem aos elementos de textos mais comumente utilizados. No caso do “ll” espanhol, o Unicode define cada “l” como sendo um elemento de codificação separado. A tarefa de combinar dois “l” juntos para ordenação alfabética é deixada para o sistema que processa o texto.

O Unicode[®] é o padrão universal de codificação de caracteres utilizado para codificação de texto para processamento computacional. As versões desse padrão são totalmente compatíveis e sincronizadas com as versões correspondentes do padrão internacional ISO/IEC 10646. O Unicode[®] fornece um meio consistente de codificar textos planos multilíngües e de trazer ordem para um estado caótico de acontecimentos que tornaram difícil a troca internacional de textos.

O projeto do Unicode é baseado na simplicidade e consistência do ASCII, mas vai muito além da habilidade limitada do ASCII em codificar somente caracteres do alfabeto latino. Ele define códigos para os caracteres utilizados em todas as principais linguagens escritas de hoje. Para manter a codificação de caracteres simples e eficiente, o padrão Unicode atribui a cada caractere um único nome e valor numérico. Ao todo, o padrão Unicode versão 5.1 fornece códigos para 100.713 caracteres dos alfabetos mundiais, conjuntos de ideogramas e coleções de símbolos.

Os padrões de codificação de caracteres definem não apenas a identidade de cada caractere e seu valor numérico, ou “ponto de codificação”, mas também como esse número é representado em bits. O padrão Unicode define três formas de codificação, que permitem o mesmo dado ser transmitido em um formato orientado em *byte*, *word* ou *double word* (isto é, em 8, 16 ou 32 bits por unidade de código). Todas as três formas de codificação codificam o mesmo repertório de

caracteres e podem ser eficientemente transformadas de uma para outra, sem perda de dados. O UTF-8 é popular para HTML e protocolos similares. UTF-8 é um modo de transformar todos os caracteres Unicode em uma codificação em bytes de tamanho variável. Ele tem como vantagem o fato de que os caracteres Unicode correspondentes ao grupo familiar do ASCII têm os mesmos valores em bytes que o ASCII, e assim, os caracteres Unicode transformados em UTF-8 podem ser utilizados com muitos sistemas existentes, sem ter que recodificá-los.

Ainda na camada inferior da arquitetura da Web semântica, encontra-se a referência para a localização dos recursos – a URI.

Segundo Berners-Lee (2005), o “*Uniform Resource Identifier*”, ou “Identificador Uniforme de Recursos” (mais conhecida pelo seu acrônimo em inglês “URI”), é uma seqüência compacta de caracteres que identifica um recurso físico ou abstrato. Uma boa forma de caracterizar as URIs é caracterizando individualmente cada termo que compõe o seu nome. Dessa forma tem-se:

1. “**Uniform**”: a uniformidade fornece diversos benefícios. Ela permite que diferentes tipos de identificadores de recursos possam ser utilizados no mesmo contexto, mesmo que os mecanismos utilizados para acessá-los sejam diferentes. Ela permite uma identificação semântica uniforme de algumas convenções sintáticas comuns entre diferentes tipos de identificadores de recurso. Ela permite ainda a introdução de novos tipos de identificadores de recursos sem interferir na forma como os atuais são utilizados. Ela permite que os identificadores possam ser utilizados em diferentes contextos, permitindo assim que as novas aplicações e protocolos utilizem o grande e amplamente utilizado grupo de identificadores pré-existentes.
2. “**Resource**”: a especificação da URI não limita o escopo do que pode ser um recurso – assim, o termo “recurso” pode ser utilizado em um sentido geral para qualquer coisa que possa ser identificada por uma URI (pode ser, por exemplo, um documento eletrônico, uma imagem, uma fonte de informações, um serviço ou uma coleção de outros recursos). Um recurso

não precisa estar necessariamente acessível via internet (seres-humanos, corporações e livros impressos, por exemplo, também podem ser recursos). Além disso, conceitos abstratos também podem ser recursos (como, por exemplo, os operadores e operandos de uma equação matemática, os tipos de relações – como “pai” ou “empregado” -, ou valores numéricos – como zero, um e infinito).

3. **“Identifier”**: um identificador deve incluir a informação necessária para distinguir o que está sendo identificado de todas as outras coisas (ou recursos) dentro do escopo de identificação, não importando como isso seja feito (pelo nome, endereço, ou contexto, por exemplo). O identificador não precisa incluir a identidade do que está sendo referenciado (o que pode ocorrer em alguns casos). Nem deve-se assumir que um sistema que faz uso de URIs irá sempre acessar os recursos por eles identificados - algumas vezes as URIs são utilizadas para simbolizar recursos, sem a intenção de que eles sejam acessados. Além disso, pode-se identificar algo que varie ao longo do tempo (como um grupo de outros recursos, um mapeamento, ou ainda diferentes recursos em diferentes momentos).

A URI é um identificador que consiste em uma seqüência de caracteres que seguem uma regra de sintaxe definida. Ela possibilita a identificação uniforme de recursos através de um grupo definido e extensível de esquemas de nomeação. Como essa identificação é realizada, designada ou possibilitada fica a cargo de cada especificação de esquema.

Uma URI começa com o nome de um esquema – que se refere a uma especificação para incluir uma URI dentro desse esquema. Sendo assim, a sintaxe das URIs é um sistema de nomeação federado e extensível no qual cada especificação de esquema pode restringir ainda mais a sintaxe e a semântica dos identificadores que usam esse esquema. A sintaxe genérica de uma URI consiste em uma seqüência hierárquica de componentes referidos como: “scheme”, “authority”, “path”, “query” e “fragment”, conforme pode ser observado na figura abaixo:

```

URI      = scheme ":" hier-part [ "?" query ] [ "#" fragment ]

hier-part = "//" authority path-abempty
          / path-absolute
          / path-rootless
          / path-empty

```

Figura 6 - Seqüência hierárquica de componentes de uma URI

(FONTE: Berners-Lee, 2005)

Existem regras para a disposição dessa seqüência de componentes, como “o ‘schema’ e o ‘path’ são obrigatórios - ainda que o ‘path’ possa estar vazio (sem caracteres)” e “quando um ‘authority’ está presente, o ‘path’ deve ser vazio ou iniciar com um caractere ‘/’ ”. A figura abaixo mostra dois exemplos de URIs, mostrando os componentes de suas partes.

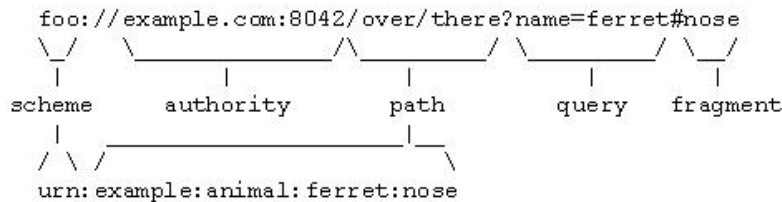


Figura 7 - Exemplo de duas URIs, mostrando os componentes de suas partes

(FONTE: Berners-Lee, 2005)

Seguem alguns exemplos de URIs, mostrando os seus esquemas e algumas variações comuns possíveis dos componentes de sua sintaxe.

```
ftp://ftp.is.co.za/rfc/rfc1808.txt
http://www.ietf.org/rfc/rfc2396.txt
ldap://[2001:db8::7]/c=GB?objectClass?one
mailto:John.Doe@example.com
news:comp.infosystems.www.servers.unix
tel:+1-816-555-1212
telnet://192.0.2.16:80/
urn:oasis:names:specification:docbook:dtd:xml:4.1.2
```

Figura 8 - Exemplos de URIs

(FONTE: Berners-Lee, 2005)

Uma URI pode ser classificada como sendo um localizador, um nome, ou os dois. O termo "Uniform Resource Locator" (URL), ou "Localizador Uniforme de Recursos", se refere a um subgrupo das URIs que, além de identificar um recurso, fornece um meio de localizá-lo através do seu mecanismo de acesso primário (por exemplo, sua "localização" em uma rede). Já o termo "Uniform Resource Name" (URN), ou "Nome Uniforme de Recurso", tem sido utilizado historicamente para se referir tanto às URIs do esquema "urn" (MOATS, 1997) – que devem permanecer únicas globalmente e persistentes (mesmos e o recurso deixar de existir ou deixa de estar disponível) -, quanto à qualquer URI que tenha propriedades de nome.

2.3.1.2. XML, Namespace e XMLSchema

A "Extensible Markup Language", ou linguagem extensível de marcação – conhecida geralmente pelo seu acrônimo "**XML**" – é uma linguagem desenvolvida pelo W3C (World Wide Web Consortium) principalmente para solucionar as limitações da HTML (MARCHAL, 2000). É uma linguagem simples e muito flexível, derivada do SGML. Originalmente projetada para suprir os desafios da publicação eletrônica em larga escala, o XML está desempenhando também um papel cada vez mais importante na troca de uma grande variedade de dados na Web e em outros lugares (BRAY, 2006).

Segundo Marchal (2000), as aplicações da XML são classificadas como pertencentes a um desses dois tipos:

a) Aplicações de documentos que manipulam informações voltadas principalmente para o consumo humano.

b) Aplicações de dados que manipulam informações voltadas principalmente para o consumo do software.

A diferença entre os dois tipos é qualitativa. O padrão XML é o mesmo, é implementado com as mesmas ferramentas, mas atende a objetivos diferentes.

A primeira aplicação da XML serve para a publicação de documentos. A vantagem principal dessa linguagem nessa área é que ela se concentra na estrutura do documento e o torna independente do meio de distribuição. Assim, é possível editar documentos XML e publicá-los automaticamente em diferentes meios. Já a segunda aplicação serve para a publicação e transferência de informações voltadas ao consumo por diferentes tipos de sistemas – o que está mais relacionado ao objetivo principal da web semântica.

Segundo Marchal (2000), o valor da XML não é conferido por ela se tratar de uma linguagem de marcação, mas por ser uma linguagem de marcação padrão. Complementando esse padrão, W3C desenvolveu diversos outros padrões - normalmente denominados “padrões acompanhantes da XML” - e entre eles está o “Namespace”.

Segundo Marchal (2000), o **namespace** associa um proprietário aos elementos. Isso ajuda na extensão, pois significa que uma organização pode aumentar os elementos existentes e rotular com clareza quem é o responsável pela extensão. Isso evita conflitos de nomes e é a única maneira de permitir a reutilização de estruturas padrão.

A XML é extensível - qualquer um pode criar tags. O problema é que essa facilidade de extensão não é gratuita. Em um ambiente distribuído, ela precisa ser gerenciada para evitar conflitos. Namespaces são uma solução para ajudar a gerenciar a facilidade de extensão da XML.

Namespaces pode ser definido como um mecanismo para identificar os elementos da XML. Dessa forma, um prefixo é incluído antes do nome de cada elemento e um sinal de dois pontos separa o nome e o prefixo.

Ex.: <qua:avaliacao>5 estrelas</qua:avaliacao>

O prefixo identifica sem ambigüidades o tipo de avaliação nesse documento. No entanto, apenas os prefixos não resolvem nada, pois qualquer um pode criar seus prefixos. Portanto, pessoas diferentes podem criar prefixos incompatíveis e com isso volta-se à estaca zero: move-se o risco de conflitos dos nomes de elementos para os prefixos. Para evitar conflitos nos prefixos, eles precisam ser declarados. A declaração de namespaces é feita por meio de atributos com o prefixo “xmlns” seguido pelo prefixo.

Ex.: <referencias xmlns:qua=”http://joker.playfield.com/star-rating/1.0” xmlns:pa=”http://penguin.xml.com/review/1.0”>

A declaração associa uma URI a um prefixo. Esse é o ponto crucial da proposta namespaces, pois os URIs, ao contrário dos nomes, são exclusivos. Os namespaces são um mecanismo para identificar de modo não ambíguo quem desenvolveu determinado elemento.

Já o **XML Schema** forma a base para a criação de vocabulários. Segundo Fallside e Walmsley (2004), a finalidade básica de um esquema é de definir classes de documentos XML – assim, “instâncias” de documentos podem ser definidas como sendo documentos XML em conformidade com um esquema em particular. Para eles, XML Schemas expressam vocabulários compartilhados e permitem que as máquinas processem regras feitas por pessoas. Eles fornecem um meio de definir estrutura, conteúdo e semântica de documentos XML mais detalhadamente, dando suporte à definição de regras de validação.

2.3.1.3. RDF e RDFSchema

Segundo Klyne, Carroll e McBride (2004), o RDF (Resource Description Framework) é um framework para representação de informações na Web. Ele possui uma sintaxe abstrata, que reflete um modelo de dados baseado em grafos simples, e uma semântica formal com uma notação rigorosamente definida para implicações lógicas, provendo a base para deduções bem fundamentadas sobre os dados RDF.

Segundo Klyne, Carroll e McBride (2004), o RDF foi projetado para ter as seguintes características:

- ser um modelo de dados simples - tornando simples o processamento e manipulação pelas aplicações;
- ter uma semântica formal e com inferências capazes de serem provadas – essa semântica fornece uma base confiável para raciocínios sobre o significado de uma expressão RDF (ela suporta notações rigorosamente definidas para implicações lógicas, que fornecem a base para a definição de regras confiáveis de inferências sobre os dados RDF);
- usar um vocabulário extensível baseados em URIs – referências URIs são utilizadas para nomear todos os tipos de coisas no RDF (com exceção dos literais);
- usar uma sintaxe baseada no XML – o RDF tem uma forma recomendada de serialização em XML (“RDF/XML Syntax Specification” – ver Beckett e McBride, 2004), que pode ser utilizada para codificar o modelo de dados na troca de informações entre as aplicações;
- suportar o uso de valores representados de acordo com os tipos de dados do XML Schema – facilitando a troca de informações entre aplicações RDF com outras que fazem uso de XML;
- permitir a qualquer um fazer sentenças sobre qualquer recurso – para facilitar as operações na escala da Internet, o RDF é um framework de “mundo-aberto”, que permite que qualquer um faça afirmações sobre qualquer recurso; assim, ele não impossibilita ninguém de criar assertivas que são incompatíveis ou inconsistentes com outras sentenças, ou com o mundo que as pessoas geralmente vêem (fica a cargo dos desenvolvedores projetar as suas aplicações para tolerar fontes de informação incompletas ou inconsistentes).

De um modo geral, o RDF usa basicamente os seguintes conceitos-chave:

1-) Modelo de dados em grafo: a estrutura fundamental sob qualquer expressão RDF é uma coleção de triplas, cada uma consistindo de um sujeito, um predicado e um objeto (um grupo dessas triplas é conhecido como grafo RDF). Essas triplas podem ser ilustradas pela ligação de um nodo com outro nodo, através de um arco direcionado, conforme observado na figura abaixo:

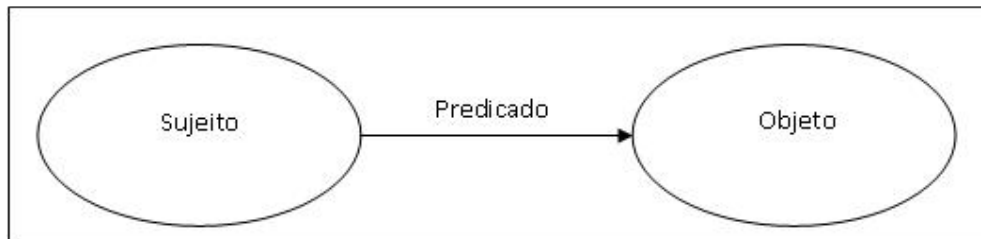


Figura 9 – Representação da Tripla-RDF
(FONTE: Klyne, Carroll e McBride, 2004)

Cada tripla representa uma sentença que relaciona dois conceitos e a seta (predicado) sempre deve apontar para o objeto. A semântica de um grafo RDF é dada pela conjunção (“E” lógico) de todas as sentenças (triplas) que ele contém (o RDF não fornece meios de expressar a negação - “NOT” – ou a disjunção – “OR”).

2-) Vocabulário baseado em URIs e identificação dos nodos:

Um nodo pode ser uma URI, um literal ou pode ser vazio. Já as propriedades só são referenciadas por URIs. Uma URI ou um literal utilizados em um nodo identificam o que esse nodo representa. Uma URI utilizada como um predicado identifica a relação entre as coisas representadas pelos nodos que ela conecta. Um predicado também pode ser um nodo em um grafo.

3-) Tipos de dados:

Os tipos de dados são utilizados no RDF como representações de valores como números inteiros, números de ponto flutuante ou datas. Por definição, o RDF utiliza os tipos de dados do XML Schema (com exceção do “rdf:XMLLiteral”,

que é pré-definido pelo próprio RDF e é utilizado para embutir XML em RDFs). Um tipo de dado é formado por um espaço léxico, um espaço de valores e um mapeamento léxico-valor – conforme pode ser observado abaixo, na representação do tipo de dado “xsd:boolean” do XML Schema, na qual cada membro do espaço de valores tem duas representações léxicas.

Value Space	{T, F}
Lexical Space	{"0", "1", "true", "false"}
Lexical-to-Value Mapping	{<"true", T>, <"1", T>, <"0", F>, <"false", F>}

Tabela 2 - Representação do mapeamento Léxico-Valor para o tipo de dado xsd:boolean do XMLSchema

(FONTE: Klyne, Carroll e McBride, 2004)

4-) Literais

Os literais podem ser o objeto de uma sentença RDF, mas não o sujeito ou o predicado. Eles podem ser do tipo “plain” (planos) ou “typed” (tipados), representando, respectivamente, textos planos em linguagem natural ou textos associados a tipos de dados (por exemplo, os literais “0” e “1”, e “true” e “false”, são associados ao tipo de dado “xsd:boolean”). Para textos que possuem marcações, deve ser utilizado o literal tipado associado ao tipo de dado “rdf:XMLLiteral”.

5-) Expressão RDF de fatos simples

Alguns simples fatos indicam a relação entre duas coisas. Esses fatos podem ser representados em uma tripla RDF, na qual o predicado nomeia a relação, e o sujeito e o objeto representam as duas coisas. Com a utilização de vocabulários baseados em URIs (que podem representar qualquer coisa que possa receber um nome), é possível expressar fatos sobre qualquer assunto.

6-) Implicações

As idéias de significado e inferência dentro de RDFs são apoiadas no conceito formal de implicação lógica, como é discutido por Hayes e McBride (2004). Em resumo, se é dito que uma expressão RDF A implica em uma expressão RDF B, significa que qualquer combinação de coisas que tornem o A

verdadeiro, tornarão também o B verdadeiro – ou seja, se a verdade da sentença A é presumida ou demonstrada, a verdade da sentença B pode ser assim inferida.

O modelo de triplas do RDF permite que sejam feitas descrições sobre conteúdo independentemente da estrutura desse conteúdo (associar propriedades a recursos, por exemplo). Em geral, mais informação é necessária para se entender, por exemplo, o que o recurso representa. Já o RDF Schema (BRICKLEY e MCBRIDE, 2004) compreende primitivas para a organização de hierarquias e para a definição de restrições sobre RDF.

O RDF Schema já pode ser considerado um formalismo para a estruturação de ontologias mais simples. No entanto, para o suporte a inferências mais poderosas, são necessários formalismos mais expressivos para a representação de ontologias, que devem incluir o suporte para axiomas e outras formas de relacionamentos entre os conceitos (GÓMEZ-PÉREZ et al, 2004).

2.3.1.4. Ontology (Ontologias)

Segundo Decker et al. (2000), a Web só é possível devido a um grupo de padrões amplamente estabelecidos e a Web semântica só seria possível se níveis adicionais de interoperabilidade fossem estabelecidos. Desse modo, os padrões deveriam ser definidos não apenas para a forma sintática dos documentos, mas também para o conteúdo semântico. Sendo assim, eles propuseram um método de codificar as linguagens de representação das ontologias em RDF/RDF Schema, aplicando esse método sobre a linguagem OIL (“Ontology Representation and Inference Language”, ou “linguagem de inferências e representação de ontologias”). O conjunto RDF/ RDF Schema seria muito mais expressivo que as propostas de representação em XML/ XML Schema que estavam sendo estudadas na época.

Mais tarde, havendo a necessidade de um poder de expressividade ainda maior, surgiu a OWL através de iniciativas européia e americana -

respectivamente, Darpa Agent Markup Language (DAML) e Ontology Inference Layer (OIL). As ontologias e a OWL serão melhores abordadas em uma seção posterior desse trabalho.

2.3.1.5. Logic (Lógica)

Para Berners-Lee, Handler e Lassila (2001), para a Web Semântica funcionar, os computadores têm que ter acesso a coleções estruturadas de informação e conjuntos de regras de inferência que eles possam utilizar para conduzir um *reasoning* (raciocínio) automatizado.

A camada de Lógica é utilizada para estender a representação ontológica, permitindo a declaração de conhecimento através de regras de produção ou lógica de predicado para suportar inferências (ANTONIOU; VAN HARMELEN, 2004; BRACHMAN; LEVESQUE, 2004).

2.3.1.6. Proof (Prova)

Segundo Berners-Lee, Handler e Lassila (2001), o real poder da Web semântica seria atingido quando as pessoas criassem e utilizassem os agentes - programas para coletar conteúdo de diversas fontes, processar as informações e trocar os resultados com outros agentes. A Web semântica promoveria essa sinergia: mesmo agentes que não foram explicitamente projetados para atuarem juntos, poderiam transferir dados entre eles quando os dados possuem semântica associada. Um ponto importante da funcionalidade dos agentes seria a troca de provas escritas em uma linguagem unificada da Web semântica – uma linguagem que expressaria as inferências lógicas realizadas através de regras e informações como as especificadas nas ontologias. Essa camada compreende então a representação de evidências e o suporte dedutivo para a validação das assertivas feitas nas camadas inferiores.

2.3.1.7. Trust (Confiança)

Segundo Berners-Lee, Handler e Lassila (2001), outra característica vital para a implementação da Web semântica seria a utilização das assinaturas digitais, que são blocos de dados criptografados que computadores e agentes podem utilizar para verificar se a informação recebida veio realmente de uma fonte específica confiável. Os agentes deveriam ser céticos em relação às assertivas que eles lêem na Web semântica até que eles tenham checado a fonte das informações.

2.3.2. A Nova Arquitetura Proposta para a Web semântica

Kifer et al (2005), propõe uma nova arquitetura para Web semântica, com algumas diferenças em relação à apresentada inicialmente por Berners-Lee, Handler e Lassila (2001). Uma dessas diferenças é a inclusão de uma camada de regras (“Rules”) no mesmo nível da camada que representa as ontologias (OWL). De qualquer modo, essa arquitetura não será aprofundada, visto que não apresenta grande relevância e impacto para o presente trabalho.

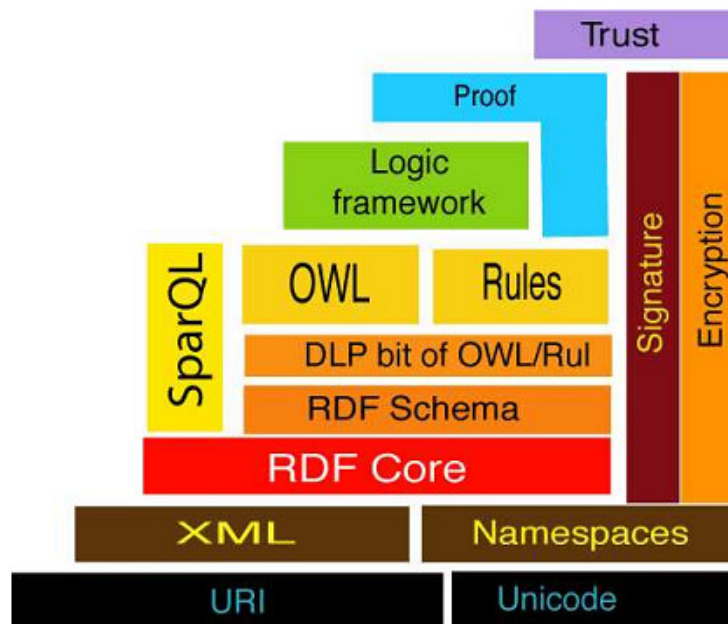


Figura 10 - Nova arquitetura para a Web Semântica

(FONTE: Kifer et al, 2005)

2.3.3. Ontologias e a Representação do Conhecimento

Pesquisadores da área de inteligência artificial têm estudado sistemas com características semelhantes às propostas da Web semântica desde muito antes do desenvolvimento da Web. Representação do conhecimento, como essa tecnologia é chamada com freqüência, está atualmente em um estado comparável ao do hipertexto antes do advento da Web: é claramente uma boa idéia, e algumas demonstrações muito interessantes existem, mas ela ainda não mudou o mundo - ela contém as sementes de importantes aplicações, mas ainda não atingiu todo o seu potencial total (BERNERS-LEE, HANDLER e LASSILA, 2001). Na arquitetura que embasa o presente trabalho, aplicam-se ontologias para a representação do conhecimento nas organizações, incluindo seus domínios (conceitos e regras de negócio) e descrevendo seus dados. Esta seção apresenta uma introdução sobre ontologias.

2.3.3.1. Definição de Ontologias

Uma das definições clássicas de ontologias foi dada por Gruber (1993), que diz que uma ontologia é uma especificação formal e explícita de uma conceitualização compartilhada. Ele ainda define o conceito de “conceitualização” como sendo uma visão abstrata e simplificada do mundo que se quer representar por algum propósito.

Segundo Guarino (1998), uma ontologia se refere a um artefato de engenharia constituído por um vocabulário específico, usado para descrever uma certa realidade, mais um conjunto de assertivas explícitas relacionados com o significado pretendido das palavras no vocabulário. Esse grupo de assertivas tem usualmente a forma de uma teoria lógica de primeira-ordem, em que palavras do vocabulário aparecem como nomes de predicados unários ou binários, chamados respectivamente de conceitos e relações. No caso mais simples, uma ontologia descreve uma hierarquia de conceitos relacionados por relações de classificação. Em casos mais sofisticados, axiomas apropriados são adicionados com o intuito de expressar outras relações entre conceitos e para restringir a sua interpretação conforme o planejado. As duas leituras de ontologia citadas possuem relação

direta, mas para evitar um impasse terminológico, usa-se com freqüência o termo “conceitualização” para se referir à ótica filosófica.

Segundo Stanford Center for Biomedical Informatics Research (2008), uma ontologia descreve os conceitos e relações que são importantes em um domínio particular, provendo um vocabulário para esse domínio assim como uma especificação formal e computável do significado dos termos utilizados nesse vocabulário. As ontologias abrangem desde taxonomias, classificações e esquemas de banco de dados, a teorias totalmente “axiomatizadas”.

2.3.3.2. Classificação das Ontologias

Segundo Guarino (1998), as ontologias podem ser classificadas em diferentes tipos, de acordo com o seu nível de generalização – ou nível de dependência em relação a uma tarefa ou ponto-de-vista.

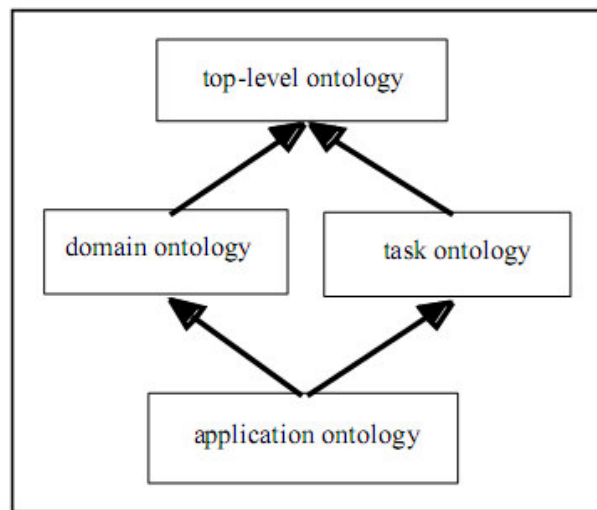


Figura 11 - Tipos de Ontologia (as setas representam relações de especialização)

(FONTE: Guarino, 2008)

As ontologias “top-level”, ou “de alto nível”, descrevem conceitos gerais - como espaço, tempo, matéria, objeto, evento e ação – que são independentes de

um problema ou domínio particular. Sendo assim, parece razoável, pelo menos na teoria, que se tivessem ontologias de alto nível unificadas, disponíveis para toda a comunidade de usuários.

As ontologias de domínio e de tarefa descrevem, respectivamente, o vocabulário relacionado a um domínio genérico (como medicina, ou automóveis) e uma tarefa genérica ou atividade (como diagnosticar, ou vender), especializando os termos introduzidos na ontologia de alto nível.

Já as ontologias de aplicação descrevem conceitos dependentes de domínios e tarefas específicos, sendo geralmente uma especialização das duas ontologias combinadas. Esses conceitos correspondem com freqüência a papéis desempenhados por entidades de domínio na execução de certas atividades.

2.3.3.3. OWL

Segundo Smith, Welty e Mcguinness (2004), a OWL (“Web Ontology Language”, ou Linguagem de Ontologia da Web) foi concebida com intuito de prover uma linguagem que pode ser utilizada para descrever as classes e as suas relações, que estão inerentes a documentos Web e aplicações.

Uma ontologia OWL pode incluir descrições de classes, propriedades e suas instâncias. Dada uma ontologia desse tipo, a semântica formal da OWL especifica como derivar suas conseqüências lógicas, ou seja, fatos não presentes literalmente dentro da ontologia, mas escondidos na semântica. Esses fatos poderiam estar baseados em um único documento, ou em múltiplos documentos distribuídos que foram combinados utilizando mecanismos da OWL.

A linguagem OWL fornece três sublinguagens, com expressividade incremental, projetadas para a utilização de comunidades específicas de usuários e desenvolvedores, conforme pode ser observado na tabela abaixo:

Tipo	Descrição
OWL Lite	<p>Suporta os usuários que precisam primariamente de uma hierarquia de classificação e de alguns mecanismos de restrições. Por exemplo, mesmo suportando restrições de cardinalidade, ela só permite os valores de cardinalidade de 0 ou 1.</p> <p>Dessa forma, ficaria mais fácil desenvolver ferramentas para suportar a OWL Lite, em relação aos tipos mais expressivos de OWL, e promover um caminho de migração rápida para tesouros e outras taxonomias.</p>
OWL DL	<p>Suporta os usuários que querem expressividade máxima, sem perder características de computabilidade, como a completude (garantia de que todas as conclusões serão tomadas) e a decidibilidade (todas as conclusões serão efetuadas em tempo finito) dos sistemas raciocinadores.</p> <p>A OWL DL inclui todas as propriedades da OWL com algumas restrições, como restrição de tipo (uma classe não pode ser também um indivíduo, ou uma propriedade; uma propriedade não pode ser também um indivíduo, ou uma classe).</p> <p>O OWL DL recebeu esse nome – usando o acrônimo DL – devido a sua correspondência com a lógica descritiva (um campo de pesquisa que tem estudado um fragmento da lógica de primeira ordem que tem a característica de poder ser sempre decidido). Por essa correspondência, a OWL DL tem as características computacionais desejáveis para sistemas raciocinadores.</p>
OWL Full	<p>Suporta os usuários que querem a máxima expressividade e liberdade sintática do RDF, sem garantias computacionais. Por exemplo, uma classe pode ser tratada ao mesmo tempo como uma coleção de indivíduos e como um único indivíduo.</p> <p>A OWL Full permite ainda que uma ontologia sobrescreva o significado dos vocabulários pré-definidos (por RDF ou OWL), dentre outras questões. Assim, é improvável que os sistemas raciocinadores sejam capazes de suportar todas as características da OWL Full, se tornando imprevisíveis.</p>

Tabela 3 - Tipos de OWL
(COMPILADO DE: Smith; Welty e McGuinness, 2008)

Além disso, cada sublinguagem é uma extensão do seu predecessor, de modo que uma ontologia OWL Lite é uma ontologia OWL DL, e uma ontologia OWL DL é uma ontologia OWL Full.

No presente trabalho, utilizaremos a linguagem OWL DL para representação de conhecimento.

2.3.3.4. Considerações sobre o uso de Ontologias

Nos anos recentes, as ontologias têm sido adotadas em muitas organizações e em comunidades científicas como um meio de compartilhar, reutilizar e processar o conhecimento do domínio. As ontologias são agora a base de muitas aplicações, como portais de conhecimento científico, sistemas de gestão da informação e de integração, comércio eletrônico e web services semânticos (STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH, 2008).

2.3.4. Aplicação de Tecnologias Semânticas no Domínio de BI

As tecnologias relacionadas à Web Semântica têm sido aplicadas de diferentes maneiras no apoio aos sistemas de informação. Guarino (1998) enumera algumas referências na literatura sobre a utilização de ontologias em áreas como engenharia do conhecimento, modelagem de banco de dados, modelagem e integração de informações, análise orientada a objetos, extração e recuperação de informações e modelagem de sistemas baseados em agentes.

Segundo Sell (2006), as tecnologias da Web semântica têm sido utilizadas de diferentes maneiras para lidar com questões relacionadas a sistemas de informação, mas não especificamente no contexto de ferramentas analíticas. O autor propôs então uma arquitetura para aplicações de business intelligence baseada na semântica do negócio. Essa arquitetura serviu como base para o desenvolvimento da plataforma SBI, vista detalhadamente a seguir.

2.4. Plataforma SBI

A plataforma SBI[®] - que foi apresentada por Sell et al (2008) e tem como base a arquitetura SBI, proposta por Sell (2006) - é desenvolvida pelo Instituto Stela (INSTITUTO STELA, 2008), onde o autor do presente trabalho atua na pesquisa e desenvolvimento de soluções de BI. Essa plataforma é utilizada no presente trabalho como base para o desenvolvimento e para testes do ambiente aqui proposto.

Segundo Sell et al. (2008), a plataforma SBI é composta por um conjunto de módulos fracamente acoplados. Parte dessa plataforma pode ser observada na figura abaixo.

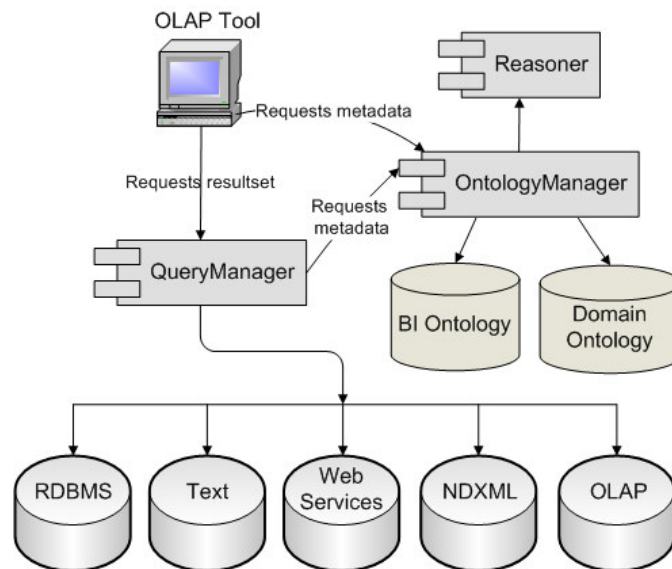


Figura 12 - A Arquiteura SBI

(FONTE: SELL et al, 2008)

As ontologias SBI incluem a semântica do negócio e descrevem as relações entre essa semântica, a terminologia de BI e as fontes de dados da organização. As ontologias são utilizadas pelo *QueryManager* para interpretar as requisições de informações realizadas pelas ferramentas analíticas e executá-las em fontes de dados heterogêneas, possibilitando a combinação de dados estruturados e não-estruturados nas mesmas análises.

O OntologyManager é o módulo que possibilita o acesso às ontologias do SBI. Este módulo conta com um *Reasoner* (interpretador) que suporta inferências “on-the-fly” ou em “batch” sobre a semântica do negócio. Essas inferências tornam possíveis operações de “slice” e “drill” guiadas pela semântica, baseando-se nas regras do negócio.

2.4.1. As Ontologias SBI

Na arquitetura SBI são utilizadas ontologias para armazenar a semântica do negócio e para definir os modelos de conhecimento necessários para gerar funcionalidades flexíveis e exploratórias dentro das ferramentas de análise. Na versão atual da plataforma SBI, essas ontologias - que estão modeladas na linguagem OWL DL - incluem as ontologias de domínio e a ontologia de BI, vistas a seguir.

2.4.1.1. A Ontologia de Domínio

Segundo Sell et al (2008), essa ontologia fornece uma terminologia formalmente especificada do domínio do negócio. Por exemplo, uma ontologia para o domínio de Pesquisa e Desenvolvimento (P&D) compreende conceitos como instituição, estudante, pesquisador, dentre outros. A figura abaixo descreve alguns conceitos relacionados a esse domínio.

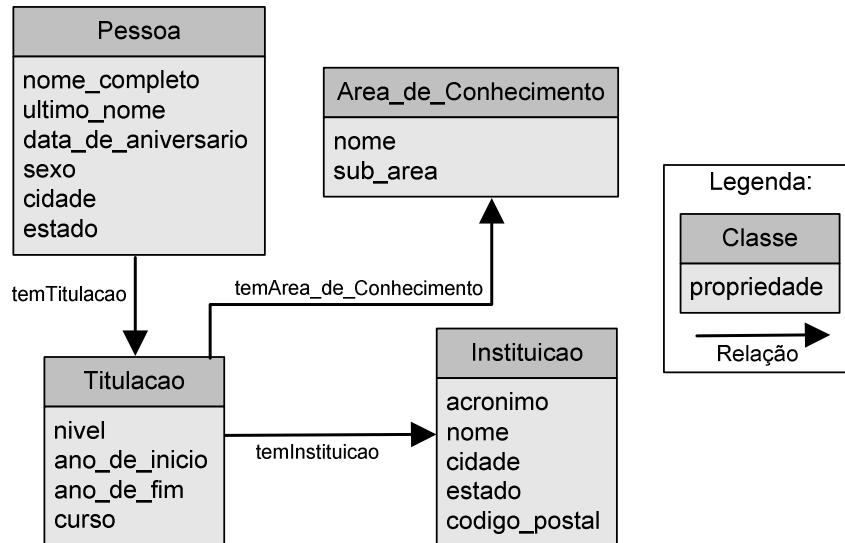


Figura 13 - Exemplo de uma Ontologia de Domínio (P&D)

(ADAPTADO DE: SELL et al, 2008)

No domínio de P&D, a classe “Pessoa” representa estudantes, pesquisadores e colaboradores das instituições. Suas titulações são representadas na classe “Titulacao”. Essa classe é ligada a duas outras classes: a classe “Instituicao”, que corresponde à universidade ou instituição onde a pessoa completou a sua titulação, e “Area_de_Conhecimento”, que descreve a área relacionada à titulação.

A ontologia de domínio permite a anotação das fontes de dados com a terminologia do negócio. Dessa forma, os usuários podem explorar as fontes de informação utilizando os conceitos do negócio, ao invés de descrições técnicas. Além disso, as relações, regras e expressões lógicas - descritas na ontologia de domínio - irão dar suporte a operações de *drill* e *slice* semânticas no cubo OLAP, à definição de consultas e à extração de detalhes adicionais dos dados apresentados pelas ferramentas de análise. As regras do negócio podem ser representadas usando SWRL (SELL et al, 2008).

2.4.1.2. A ontologia de BI

Segundo Sell et al (2008), a ontologia de BI modela os conceitos utilizados para descrever como os dados são organizados nos repositórios de dados e para mapear esses dados aos conceitos da ontologia de domínio. Essas definições são utilizadas para:

a) dar suporte às inferências, utilizando a ontologia de domínio, para estender os resultados das consultas;

b) dar suporte à apresentação dos resultados das consultas, utilizando a terminologia do negócio;

c) prover uma abstração dos repositórios de dados, para guiar a interação do tomador de decisões com as fontes de informação da organização.

Além disso, essa ontologia mapeia os conceitos OLAP, utilizados por ferramentas analíticas, para guiar os tomadores de decisão na definição das análises e para fornecer operações de *drill* e *slice* semânticas (SELL et al, 2008).

Os principais conceitos de Business Intelligence são modelados na ontologia de BI, como é mostrado na figura abaixo.

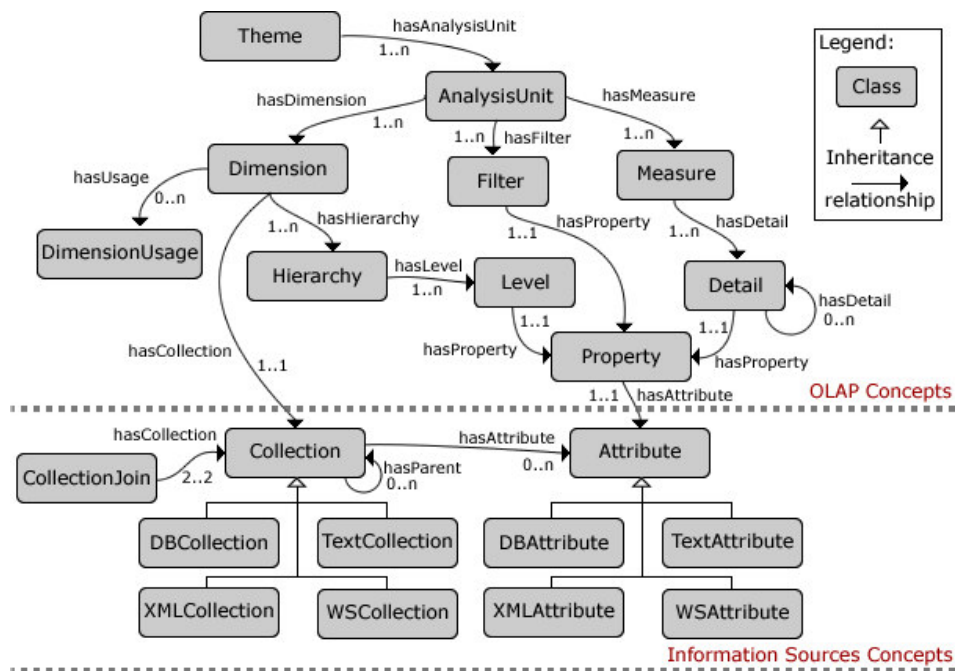


Figura 14 - A Ontologia de BI

(FONTE: SELL et al, 2008)

A tabela abaixo mostra uma descrição das principais classes:

Conceito	Descrição
<i>Theme</i> (Tema)	Representa documentos, tabelas de fato e dimensões associados com um processo de negócio. Em uma correlação com BI, pode representar um <i>data mart</i> (um exemplo de tema poderia ser P&D).
<i>AnalysisUnit</i> (Unidade de Análise)	Define as tabelas de fato e coleções de documentos relacionados a um assunto específico de um tema (em um tema P&D poderia ser definido, por exemplo, uma unidade de análise “Evasão Escolar”). Uma unidade de análise pode ter muitas dimensões e medidas.
<i>Measure</i> (Medida)	Esse conceito é utilizado para representar valores quantitativos, agregações ou sumarizações relacionadas ao conteúdo de uma unidade de análise. No contexto de BI, podem representar os fatos das

	tabelas de fato (por exemplo: número de estudantes).
<i>Filter</i> (Filtro)	Filtros são atributos dimensionais que podem ser aplicados para fazer operações de <i>slice</i> e <i>dice</i> em dados relacionados a uma unidade de análise (por exemplo: idade dos estudantes, sexo, etc.)
<i>Dimension</i> (Dimensão)	Descreve as dimensões de uma unidade de análise. Dimensões podem ter muitas hierarquias e propriedades (por exemplo, "Estado").
<i>Hierarchy</i> (Hierarquia)	Esse conceito descreve as hierarquias das dimensões. Cada hierarquia é composta por um ou mais níveis (por exemplo, cidade, estado e país).
<i>Level</i> (Nível)	Representação do nível hierárquico, podendo ser utilizado em operações de <i>drill-up</i> e <i>drill-down</i> .
<i>Detail</i> (Detalhamento)	Descreve como uma unidade de análise pode ser detalhada ou apresentada em seu nível atômico (por exemplo, nome e e-mail).
<i>Property</i> (Propriedade)	Identifica a terminologia utilizada para apresentar uma unidade de informação. Ela também mapeia instâncias de conceitos de atributos com instâncias de conceitos de detalhamento, filtro, nível e medida.
<i>DimensionUsage</i> (Uso da Dimensão)	Descreve como coleções de dados (dimensões) são ligadas às unidades de análise. Também descreve quais propriedades são utilizadas para efetuar junções entre as dimensões.
<i>Collection</i> (Coleção)	Esse conceito representa uma coleção de dados ou uma fonte de dados e descreve como elas se relacionam aos conceitos representados na ontologia de domínio.
CollectionJoin (Junção de Coleção)	Descreve como uma coleção pode se ligar com outra (junção). Também identifica que propriedades e

	operações são utilizadas para unir duas coleções.
Attribute (Atributo)	Corresponde aos itens contidos em coleções, como campos de tabelas, elementos XML, entidades extraídas de documentos, ou colunas de planilhas eletrônicas. Também associa esses elementos com conceitos representados na ontologia de domínio.

Tabela 4 - Descrição das principais classes da Ontologia de BI

(ADAPTADO DE: SELL et al, 2008)

Nas soluções tradicionais de BI, para cada acréscimo ou modificação de regras de negócio, é necessário mudar as rotinas de ETL e os esquemas dos *data marts* – esse desenvolvimento consome tempo e depende de diferentes técnicos para ser efetuado. As ontologias SBI compreendem os conceitos e as regras do negócio formalmente modelados, e essa semântica guia o acesso e a exploração de informações em tempo de decisão. Assim, mudanças nas regras de negócio podem ser feitas simplesmente pela modificação dos conceitos e relações representados na ontologia de domínio (SELL et al, 2008).

2.4.2. Módulos funcionais do SBI

As ontologias do SBI são utilizadas por módulos funcionais para dar suporte a ferramentas analíticas na localização e exploração de fontes de dados.

O **QueryManager** fornece um acesso transparente a base de dados heterogêneas, utilizando as ontologias para descrever o conteúdo das fontes de informação. Dessa forma, ele possibilita a combinação de resultados de consultas sobre fontes de dados estruturadas e não-estruturadas, independente das suas localizações. Com isso, a forma como os dados estão organizados e em que local estão mantidos torna-se transparente às ferramentas de apoio à decisão. As requisições das ferramentas analíticas são traduzidas em consultas pelo QueryManager com o auxílio do OntologyManager (SELL et al, 2008; NAPOLI et al, 2006).

Segundo Sell et al (2008), o **OntologyManager** é responsável em manipular a ontologia de BI e em retornar as informações necessárias para a realização de consultas. Ele informa os detalhes sobre as coleções de dados - como os nomes de esquemas, nomes de tabelas e suas definições de campos – e informa as suas relações com os conceitos definidos na ontologia de domínio.

As consultas são então processadas nas fontes de dados correspondentes. No caso de requisições que envolvem repositórios heterogêneos, o QueryManager quebra a mensagem e envia as partes para os diferentes *drivers* (para cada tipo de repositório de dados, existe um driver correspondente, que lida com as suas especificidades). O resultado retornado por cada *driver* é unido pelo QueryManager e mandado de volta para as ferramentas analíticas – processo que esconde a complexidade dos repositórios de dados dessas ferramentas (SELL et al, 2008; NAPOLI et AL, 2006).

2.4.3. Reasoning

Segundo Sell et al (2008), levando em consideração que as ontologias SBI são descritas em uma linguagem formal que possibilita a explicitação de regras de negócio e a definição de axiomas para especificar a relação entre os conceitos, operações de “*slice*” e “*drill*” semânticas são possibilitadas de forma que relações e regras são aplicadas para filtrar ou expandir os resultados das consultas, baseando-se em sinônimos, hipônimos² e outras relações especificadas pelas ontologias.

As inferências podem ser realizadas “on the fly” – estratégia na qual os raciocínios são realizados em tempo de decisão – e “em batch” – estratégia em que as inferências são feitas após o processo de ETL (SELL et al, 2008; SILVA, 2006).

2.4.4. Considerações sobre o uso das ontologias da plataforma SBI

O uso de ontologias e a possibilidade da realização de inferências semânticas são a base e o grande diferencial da arquitetura que embasa a plataforma SBI em relação às outras arquiteturas tradicionais de BI. Assim, como já foi dito, tem-se um poder analítico baseado na semântica e no conhecimento do negócio da organização. Contudo, para a aplicação prática dessa plataforma, são necessárias algumas etapas a mais que em uma aplicação de uma solução de BI tradicional. Dentre essas etapas, estão a manipulação das ontologias utilizadas pela plataforma, incluindo a criação de instâncias que representem seus conceitos e a correlação de diferentes conceitos e instâncias, através de um processo de mapeamento de ontologias.

² Palavras menos genéricas que especificam o significado de outra.

2.5. Integração e Mapeamento de Ontologias

Para Kalfoglou e Schorlemmer (2003), o mapeamento de ontologias pode fornecer uma camada comum, através da qual diferentes ontologias podem ser acessadas. Segundo esses autores, existe uma enorme quantidade e diversidade de trabalhos relativos a mapeamento de ontologias - eles costumam utilizar diferentes termos e conceitos, como: “alinhamento”, “merging”, “articulação”, “fusão”, “integração”, “morfismo”, etc. Dada essa diversidade, é difícil compreender todas as soluções fornecidas e os seus problemas inerentes – parte disso é por causa da falta de abordagens compreensivas, de padrões de terminologia, de detalhamentos técnicos que não foram revelados e de métricas de avaliação.

Kalfoglou e Schorlemmer (2003) definem “Mapeamento de Ontologias” como sendo a tarefa de relacionar o vocabulário de duas ontologias, que compartilham o mesmo domínio de discurso, de uma forma que a estrutura matemática das ontologias e suas interpretações desejadas – como especificadas pelos axiomas – são respeitadas. Mapeamentos de estruturas matemáticas que preservam determinadas estruturas são chamados de “morfismos” (por exemplo, uma função f aplicada a dois grupos preservando uma relação de tamanho entre eles – como “ $a \leq b$ implica em $f(a) \leq f(b)$ ” - é um morfismo). Dessa forma, pode-se também caracterizar um mapeamento de ontologias como sendo um morfismo de assinaturas ontológicas, no qual todas as interpretações que satisfazem os axiomas de uma ontologia satisfazem também os da outra. Além disso, para acomodar uma noção um pouco mais flexível de mapeamento de ontologias, esses autores apresentam a definição de “mapeamento parcial de ontologias”, no qual existe uma sub-ontologia que representa uma parte da primeira ontologia e, através dessa sub-ontologia, pode-se fazer um mapeamento total com a segunda ontologia.

Uma das abordagens apresentadas por Kalfoglou e Schorlemmer (2003) para o mapeamento de ontologias é a utilização de ontologias populadas – segundo esses autores, essa abordagem é encontrada em diversos casos na literatura. Ela é caracterizada pela utilização de instâncias de uma ontologia

fazendo referência a conceitos da outra ontologia – para isso são utilizadas relações de classificação entre as instâncias e os conceitos da ontologia. Essa também será a abordagem utilizada no presente trabalho para mapear as instâncias da ontologia de fontes de informação com as ontologias analítica e de domínio de aplicação.

Outro detalhe citado por Kalfoglou e Schorlemmer (2003) é que entre as técnicas de mapeamento de ontologias mais populares que eles encontraram na literatura, estão algumas que fazem uso de heurísticas. Segundo eles, as heurísticas são muito utilizadas por serem fáceis de implementar e de utilizar e por suportarem automação; por outro lado, podem falhar em alguns casos (mesmo as mais elaboradas). Geralmente elas são baseadas em características sintáticas ou estruturais, mas não englobam a semântica. Eles afirmam ainda que a maioria dos trabalhos que se propuseram a fazer mapeamentos de forma totalmente automatizada falhou. Por isso, é provável que uma intervenção humana deva ser necessária para explicitar alguns significados com mais exatidão.

Sendo assim, para Kalfoglou e Schorlemmer (2003), as heurísticas podem ser utilizadas como um mecanismo para iniciar um processo de mapeamento de forma semi-automática – como um modo de dar praticidade ao processo – mas, ainda assim, a intervenção de um usuário humano continua a ser muito bem vinda - como uma forma de validar ou endossar os resultados, de verificar as ontologias utilizadas como entrada, de modificar as regras de mapeamento, ou até de fazer alguns mapeamentos de forma totalmente manual (o que resultaria, teoricamente, em mapeamentos de maior qualidade). Para isso, esses usuários precisariam ser experts no domínio da aplicação e ter familiaridade com os formalismos e tecnologias utilizados no processo. Essa forma de mapeamento semi-automático, utilizando heurísticas e com intervenção humana, foi a abordagem utilizada no desenvolvimento do presente trabalho, conforme visto a seguir.

3. Desenvolvimento do Trabalho

Neste capítulo, é apresentado um relato sobre a realização das propostas definidas para o presente trabalho. Dessa forma, são definidas as ontologias que foram utilizadas - tendo-se como base a plataforma SBI, que apoiou a construção do ambiente apresentado. Além disso, são definidas as etapas necessárias para a utilização e manutenção dessas ontologias e a arquitetura que especifica o ambiente desenvolvido com esta finalidade, bem como as tecnologias e ferramentas utilizadas na construção do protótipo do mesmo. Por fim, são feitas comparações com algumas ferramentas comerciais que poderiam ser utilizadas para o mesmo propósito.

3.1. Ontologias Utilizadas

No momento da realização deste trabalho, alguns pontos da plataforma SBI - que está em constante evolução - já haviam sido modificados, em relação ao que foi apresentado por Sell et al (2008). Alguns desses pontos são as ontologias, utilizadas para suportar as anotações semânticas da plataforma. Essas ontologias, que são a base para a construção do ambiente aqui proposto, serão apresentadas sucintamente abaixo.

A “ontologia de BI” evoluiu em relação às suas versões anteriores, comportando novos conceitos e se tornando mais abrangente. Algumas das modificações foram realizadas visando à inclusão de conceitos relacionados à extração de conhecimento a partir de bases textuais (KDT). Com isso, alguns conceitos tiveram que ser criados e outros - que tinham relação somente com termos do BI tradicional, mas que poderiam ser comuns ao KDT - tiveram que ser repensados, remodelados e renomeados.

A figura abaixo apresenta uma representação hierárquica de parte dos conceitos dessa ontologia. Alguns conceitos, não utilizados no presente trabalho (inclusive os relacionados ao KDT), e as propriedades e relações foram omitidos

por respeito à propriedade intelectual do Instituto Stela e por não terem grande relevância para o entendimento deste trabalho.



Figura 15 - Parte da ontologia analítica utilizada no presente trabalho

(FONTE: gerada a partir do plugin "OWLviz" do Protégé, 2008)

Essa ontologia pode ser dividida internamente em outras sub-ontologias, através do agrupamento de conceitos que representam um universo de interesse em determinado momento do processo de mapeamento aqui proposto. Assim, dividiu-se a ontologia em dois principais grupos de conceitos: os conceitos relacionados à representação das fontes de informação (chamados de "ontologia de fontes de informação – ou, simplesmente, "ontologia ISO" – e representados

pelo *namespace* “iso”) e os conceitos relacionados à parte analítica da ontologia (chamados de “ontologia analítica” e representados pelo *namespace* “ano”).

Dessa forma, a ontologia ISO inclui os conceitos “Information Source”, “Merge”, “Attribute” e “Collection” (juntamente com as especializações dos dois últimos: “DBAttribute” e “DBCollection”). Já a ontologia analítica, inclui os conceitos “Analytical Element” e as suas sub-classes “Theme”, “ThemeUnit”, “BIThemeUnit”, “Grouping”, “GroupingUnit”, “MeasureUnit” e “Detail”.

Por fim, os outros conceitos presentes na ontologia - que são utilizados pelas duas sub-ontologias citadas anteriormente - incluem o grupo relacionado à apresentação visual das informações (indicado pelo *namespace* “vio”) e o grupo relacionado ao mapeamento e tratamento das informações (indicado pelo *namespace* “map”). O primeiro é formado pelos conceitos “VisualElement”, “Label” e “Icon” e o segundo por “Function”, “AggregateFunction”, “Expression”, “UnaryExpression”, “BinaryExpression” e “NestedExpression”.

Visto isso, identificaram-se três principais grupos de conceitos que devem ser manipulados para suportar a utilização da plataforma SBI: a “ontologia de domínio” e as duas sub-ontologias da “ontologia de BI” citadas anteriormente - a “ontologia ISO” e a “ontologia analítica”. Com isso, puderam-se identificar algumas etapas necessárias para o mapeamento dessas ontologias, conforme é visto a seguir.

3.2. Etapas Necessárias no Mapeamento das Ontologias

Foram identificadas algumas etapas necessárias para o mapeamento e utilização das ontologias da plataforma SBI, como forma de suportar a utilização dessa plataforma de uma maneira que exija menos esforço por parte dos usuários finais, do que exigiria se eles tivessem que manipular as ontologias usando as ferramentas tradicionais para realizar esse tipo de tarefa. Essas etapas são descritas a seguir.

- 1º) Disponibilização de classes que representem os conceitos das ontologias SBI, provendo métodos para criação de instâncias dessas classes (que representarão instâncias dos conceitos relacionados), para o estabelecimento de relacionamentos entre essas instâncias e para a manipulação das mesmas. Também deve ser incluso classes genéricas, que possam representar qualquer instância de ontologia de domínio.
- 2º) Disponibilização da ontologia de domínio da aplicação – possivelmente em um arquivo OWL.
- 3º) Realização da conexão com as fontes de informação que serão utilizadas – para efeitos desse trabalho, conexão com o banco de dados onde está armazenado o data warehouse da organização.
- 4º) Realização do mapeamento automático entre os conceitos da ontologia de fontes de informação com os recursos do data warehouse (como esquemas, tabelas e atributos), criando as instâncias dos conceitos que representarão esses recursos. Para isso poderão ser utilizadas meta-informações adquiridas no banco de dados sobre a estrutura dos esquemas utilizados.
- 5º) Possibilitar ao usuário uma forma de fazer ajustes manuais nessa ontologia, como a alteração do tipo de junção entre tabelas (ou mesmo a criação de novas junções), a aplicação de funções sobre os atributos, dentre outros. Esse processo vai influenciar diretamente nas consultas semânticas que serão realizadas baseadas nessa ontologia.

- 6º) Fazer o mapeamento automático das instâncias da ontologia de fontes de informação com os conceitos da ontologia analítica, criando as instâncias dessa última (que apontarão para as instâncias da primeira). Isso poderá ser realizado através de algumas heurísticas, como: “uma tabela de fato pode ser reconhecida como sendo uma tabela que uma chave composta formada por chaves estrangeiras de três ou mais tabelas”, ou “as medidas são representadas pelo prefixo “FT” nos nomes dos atributos de uma tabela de fato”. Além disso, deve ser dada ao usuário a possibilidade de iniciar esse processo informando explicitamente as variáveis que serão utilizadas no mapeamento automático, possivelmente através do uso de expressões regulares – informando, por exemplo, quais são os prefixos utilizados para representar as tabelas de fato ou as medidas nas fontes de informação.
- 7º) Possibilitar ao usuário uma forma de fazer, de forma manual, ajustes nessa ontologia analítica, corrigindo os resultados do mapeamento gerados automaticamente pela ferramenta - reclassificando as instâncias da ontologia de fontes de informação nos conceitos da ontologia analítica. Além disso, deve ser dada ao usuário a possibilidade de realizar outras configurações na ontologia – definindo manualmente características desejadas pelo analista (como a criação de novos temas e unidades de análise) ou características que dificilmente podem ser detectadas automaticamente (como a definição da hierarquia de atributos, por exemplo).
- 8º) Realizar de forma automática a criação em memória dos objetos que representam os conceitos da ontologia de domínio da aplicação, disponibilizada anteriormente.
- 9º) Possibilitar ao usuário uma forma de fazer manualmente o mapeamento das instâncias da ontologia de fontes de informação com os conceitos definidos na ontologia de domínio da aplicação.
- 10º) Realizar a persistência das ontologias geradas de forma a poderem ser carregadas pelas ferramentas compatíveis com a plataforma SBI.

Visto essas etapas e tendo-as em mente como requisitos para a realização do processo de manipulação das ontologias que apóiam a plataforma SBI, pode-se iniciar o projeto do ambiente proposto.

3.3. Ferramenta Proposta

A ferramenta proposta, de acordo com objetivos estabelecidos no presente trabalho - que foram mais detalhados na seção anterior, no item sobre o levantamento das etapas - possibilitará o mapeamento semi-automático dos recursos dos diferentes repositórios de informação com os conceitos da ontologia de fontes de informação. Dessa forma, serão criadas instâncias dos conceitos dessa ontologia, que representarão os recursos mapeados. Com isso, o usuário poderá fazer manualmente ajustes nessa ontologia, como alteração do tipo de junção (do padrão “inner join”, para “left join” ou “right join”), a aplicação de funções sobre os atributos, dentre outros. Além disso, a ferramenta possibilitará o mapeamento dessas instâncias com os conceitos da ontologia de análise e com os conceitos da ontologia domínio da aplicação, possibilitando a manipulação dessas informações pelas ferramentas analíticas e fazendo uso da linguagem de domínio, tornando mais fácil a interação pelo usuário. A arquitetura projetada com esse propósito pode ser vista a seguir.

3.3.1. Arquitetura

O ambiente proposto foi desenvolvido utilizando uma arquitetura cliente-servidor, conforme pode ser observado na figura abaixo:

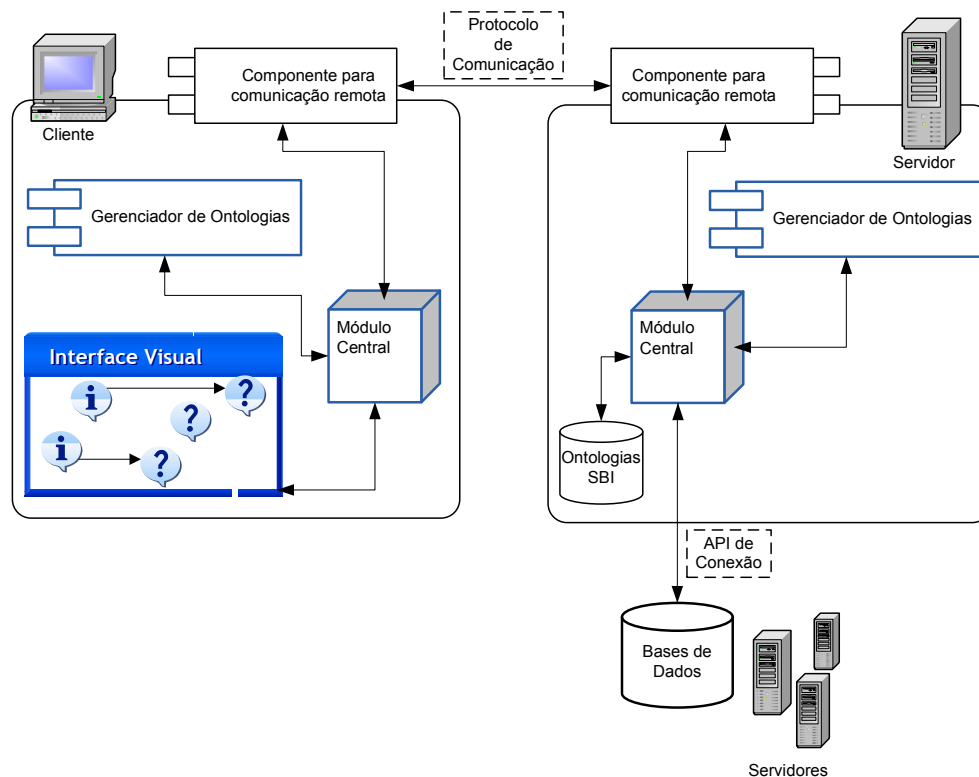


Figura 16 - Arquitetura do ambiente proposto

3.3.1.1. Servidor (“Back-end”)

No lado do servidor da arquitetura, existe um módulo central que coordena todos os processos de “back-end”. Esse módulo é responsável por fazer a conexão com os diferentes repositórios de dados, que podem ser selecionados pelo cliente.

No servidor também estão as classes que representam ontologias SBI, as classes projetadas para representar qualquer conceito de uma ontologia de domínio e um módulo gerenciador de ontologias - responsável por instanciar e

manipular os objetos dessas classes, além de auxiliar na persistência das ontologias.

Por fim, existe um módulo para realização da comunicação remota com os clientes, disponibilizando os serviços que são oferecidos pelo módulo central.

Os serviços disponibilizados estão relacionados abaixo, em uma disposição que reflete uma característica de dependência entre eles – os serviços inferiores só poderão ser acionados quando os superiores já tiverem sido.

a) Recuperação dos esquemas disponíveis:

É feita uma conexão com o SGBD e são enviados ao cliente os nomes dos esquemas disponíveis na base de dados desejada.

b) Mapeamento dos recursos da base com a ontologia ISO:

É feita uma conexão com o SGBD e, utilizando as informações por ele fornecidas, são mapeados automaticamente os recursos do esquema desejado com os conceitos da ontologia de fontes de informação. É então retornado para o cliente um objeto da classe “DBCcollection”, que representa o esquema com todas as suas tabelas (também representadas por instâncias da classe “DBCcollection”), que por sua vez são formadas por atributos (representados por “DBAttribute”). Já os relacionamentos entre as tabelas são representados por instâncias da classe “Merges”.

c) Mapeamento da ontologia ISO com a ontologia de análise

É feito o mapeamento automático das instâncias da ontologia ISO com os conceitos da ontologia analítica, retornando para o cliente uma instância padrão da classe “Theme”, com todas as suas unidades de tema (“BIThemeUnit”), que por sua vez possui instâncias representando as medidas (“MeasureUnit”), agrupamentos (“Grouping”) e unidades de agrupamento (“GroupingUnit”). Esse mapeamento pode ser realizado utilizando as heurísticas programadas para o sistema e/ou utilizando expressões fornecidas pelo cliente. Como exemplo, pode-se ter como

regras: criar sempre um tema padrão (onde ficarão inicialmente o restante das instâncias da ontologia) e criar uma unidade de tema para cada tabela de fato - reconhecidas por prefixos comuns (como “FT_” e “FATO_”), ou por terem a característica de possuir uma chave composta formada por várias chaves estrangeiras, ou ainda fazendo uso de expressões regulares informadas pelo cliente.

d) Persistência das ontologias

É feita a persistências das ontologias geradas, utilizando arquivo ou uma base de dados.

e) Recuperação e disponibilização de ontologias persistidas

Uma ontologia persistida pode ser recuperada em memória e os objetos criados podem ser disponibilizados para o cliente.

3.3.1.2. Cliente (“Front-end”)

No lado cliente, também existe um módulo central, que coordena todos os processos de “front-end”. Esse módulo é responsável por inicializar a interface visual (GUI) e gerenciar todas as requisições realizadas pelo usuário, fazendo a invocação remota dos serviços disponibilizados pelo servidor - com o auxílio do módulo de comunicação remota – e recebendo e tratando as respostas recebidas.

Com o auxílio da interface visual, o módulo central troca com os usuários as informações necessárias para a realização das etapas automáticas do processo de mapeamento das ontologias. Além disso, são disponibilizadas formas para os usuários fazerem refinamentos desses mapeamentos, além de formas para fazer as etapas que são totalmente manuais – sempre buscando a maior simplificação possível de todo o processo.

O cliente, assim como o servidor, também possui também um gerenciador de ontologias, que auxilia o módulo central a armazenar e manipular os objetos que representam as ontologias no lado do cliente.

3.3.1.3. Fluxo de execução do processo de mapeamento

O fluxo de execução sugerido para todo o processo, tendo como base a arquitetura criada, é visto a seguir.

Com o servidor rodando e oferecendo seus serviços adequadamente, o usuário executa o cliente, através de um navegador Web, e visualiza a tela principal do sistema. Ele então pode preencher os atributos de conexão com a base de dados desejada (como nome da base, localização na rede, usuário e senha) ou selecionar uma conexão salva anteriormente.

Feito isso, pode ser feita a invocação do serviço que faz a conexão com o SGBD e retorna o nome dos esquemas presentes na base de dados selecionada. Assim, o cliente pode escolher um esquema e solicitar ao servidor o serviço que mapeia os recursos desse esquema com os conceitos da ontologia de fontes de informação e que retorna um objeto da classe “DBCcollection” - que representa esse esquema e inclui as instâncias que representam todas as tabelas e seus relacionamentos e atributos. A GUI então apresenta essas instâncias, para que o usuário possa visualizá-las e fazer ajustes sobre elas – como criação de novos relacionamentos ou alterações dos relacionamentos criados automaticamente (modificando, por exemplo, o tipo de junção ou os atributos que fazem parte do relacionamento).

Finalizada a etapa de mapeamento semi-automático dos recursos da base de dados com os conceitos da ontologia ISO, o usuário pode então fazer a invocação do serviço que faz o mapeamento automático das instâncias dessa ontologia com os conceitos da ontologia analítica, retornando uma instância da classe “Theme”, contendo todas as instâncias dos conceitos derivados dessa classe (que já possuem referência para instâncias dos conceitos da ontologia ISO). Mas, antes da invocação desse serviço, o usuário poderá ainda escolher se o mapeamento deve ser realizado através das heurísticas do sistema, conforme já apresentado na seção que descreve os serviços disponibilizados pelo servidor, ou utilizando expressões criadas pelo próprio cliente.

A GUI então apresenta essas novas instâncias, para que o usuário possa visualizá-las e também fazer ajustes manuais sobre elas – como, por exemplo, a criação de novos temas, unidades de temas e agrupamentos; a modificação dos já existentes (gerados automaticamente), podendo ser feita a alteração dos recursos que fazem parte deles ou a modificação dos rótulos (também gerados automaticamente); a definição de hierarquia de atributos para suportar operações de *drill-up* e *drill down* (como, por exemplo, definir que “região” é pai de “estado”, que é pai de “município”); e a definição de detalhamentos (“Details”) das unidades de tema.

Finalizada a etapa de mapeamento semi-automático das instâncias da ontologia ISO com os conceitos da ontologia analítica, o usuário pode então realizar o mapeamento manual das instâncias da primeira ontologia com os conceitos da ontologia de domínio. Para isso, ele deve fazer o *upload* de um arquivo com essa ontologia ou informar a sua localização na rede, para que o servidor possa acessá-la e gerar as instâncias que correspondem aos seus conceitos. Esse serviço retorna para o cliente essas instâncias, que poderão ser apresentadas na GUI, possibilitando o início dessa operação de mapeamento.

Finalizadas as etapas de mapeamentos e configurações das ontologias, elas podem então ser persistidas, em arquivo ou em um banco de dados, para possibilitar a utilização pelas ferramentas analíticas compatíveis com a plataforma SBI. Além disso, a qualquer momento, o usuário pode solicitar a recuperação dessas ontologias, pela ferramenta de mapeamento, para realização de novas manutenções.

3.3.2. Implementação do Protótipo

O protótipo foi desenvolvido tendo como base a arquitetura e o fluxo de execução apresentados anteriormente. Na figura abaixo, observa-se a substituição dos componentes genéricos da arquitetura pelos componentes e tecnologias específicos que foram utilizados.

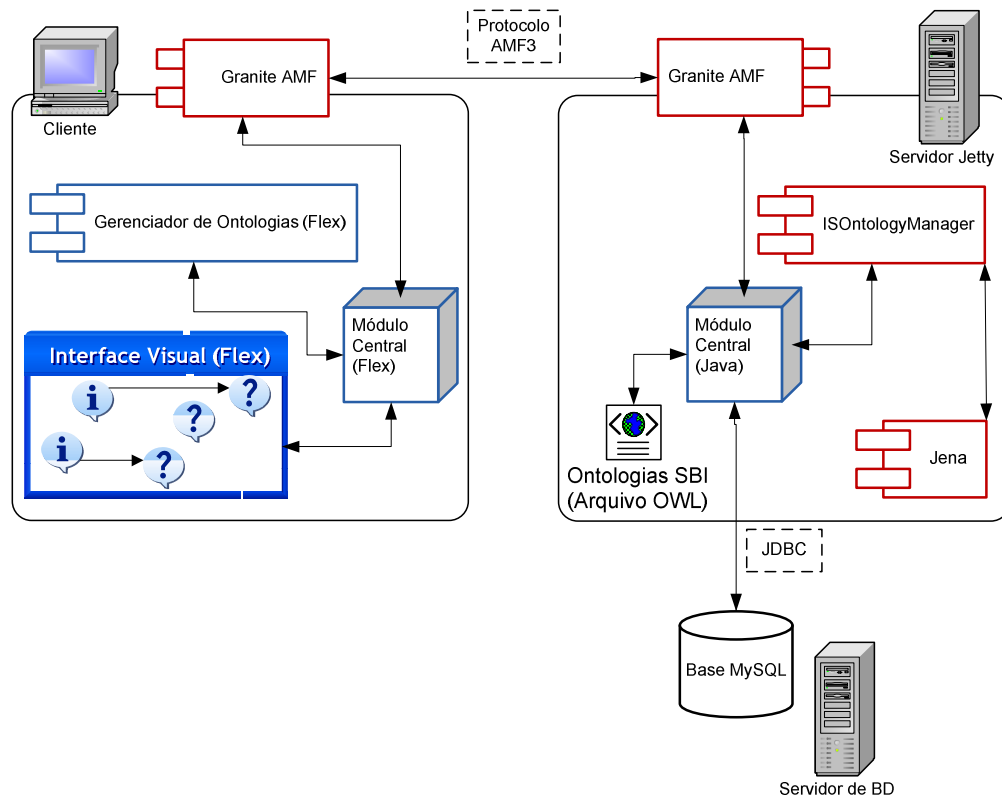


Figura 17 – Visão geral dos módulos utilizados no protótipo

A descrição dos componentes, ferramentas e tecnologias que foram utilizados no desenvolvimento do protótipo pode ser observada a seguir.

3.3.2.1. Descrição dos componentes, ferramentas e tecnologias

Para o desenvolvimento e aplicação do protótipo descrito no presente trabalho, foram utilizadas algumas ferramentas, tecnologias e componentes de terceiros, que são descritos a seguir.

- **Eclipse 3.4:** um ambiente de desenvolvimento (THE ECLIPSE FOUNDATION, 2008) que pode ser utilizado com o Java e, através do uso do plugin Flex Builder (ADOBE SYSTEMS INCORPORATED, 2008c), com o Flex.
- **Java SE 1.6:** para o desenvolvimento da camada de serviços (no lado servidor) foi utilizado o framework Java SE (SUN MICROSYSTEMS, 2008b).
- **Flex 3:** no desenvolvimento da camada cliente do protótipo, utilizou-se o framework Flex (<http://www.adobe.com/products/flex/>).
- **Protocolo AMF3:** o AMF 3 (Action Message Format) é um formato binário compactado, livre e aberto, que é utilizado para serializar objetos action script. Uma vez serializados, os objetos codificados pelo AMF podem ser utilizados para persistir ou recuperar o estado de uma aplicação entre diferentes sessões ou para permitir que dois pontos se comuniquem através da troca de dados fortemente tipados. Além disso, o AMF pode ser utilizado em invocações assíncronas de serviços remotos (ADOBE SYSTEMS INCORPORATED, 2008a). Esse protocolo foi utilizado na comunicação e na serialização e transmissão dos objetos (que representam as ontologias SBI) entre a camada cliente e a camada do servidor, através de uma implementação fornecida pelo Granite Data Services.
- **Granite Data Services 1.1:** também chamado de Granite DS ou, simplesmente, GDS, é uma alternativa livre e de código aberto (sob licença LGPL) para o Adobe LiveCycle Data Services (ADOBE SYSTEMS INCORPORATED, 2008b) para utilização em servidores de aplicação

J2EE (SUN MICROSYSTEMS, 2008a). O principal objetivo desse projeto é de fornecer um framework para o desenvolvimento de aplicações (Flex 2+ com EJB3, Seam, Spring, Guice ou Pojo), utilizando os benefícios do AMF3 (incluindo a invocação de objetos remotos). O GDS foi projetado para ser leve, robusto, rápido e altamente configurável (ADEQUATE SYSTEMS, 2008).

- **Servidor Jetty 7:** é um servidor Web de código aberto (sob licença Apache 2.0), implementado todo em Java (MORT BAY CONSULTING, 2008). Ele foi utilizado na disponibilização dos serviços da arquitetura por demonstrar ser simples, eficiente, muito rápido e integrável ao ambiente de desenvolvimento.
- **ISOntologyManager (OM):** é um componente que foi utilizado para fazer o papel do gerenciador de ontologias apresentado no lado servidor da arquitetura do ambiente proposto. Internamente, o OM faz uso do componente Jena.
- **Jena:** é um framework Java, para a construção de aplicações com base na Web semântica. Ele fornece um ambiente programático para RDF, RDFS e OWL (fornecendo APIs de manipulação – leitura, escrita, persistência em memória ou em repositórios) e para SPARQL (fornecendo um engine para consultas). Ele inclui também inclui um engine para inferências baseadas em regras (HEWLETT-PACKARD DEVELOPMENT COMPANY, 2008).
- **SGBD MySQL 5.0:** é o Sistema Gerenciador de Banco de Dados (<http://www.mysql.com/>) que foi utilizado na etapa de aplicação e validação do protótipo.

3.3.2.2. Considerações sobre os componentes, ferramentas e tecnologias

Para o desenvolvimento do protótipo, foram utilizadas as plataformas Java e Flex por fatores como: a experiência do autor com essas plataformas (por trabalhar diariamente com elas); a eficiência e robustez do Java, suficientes para o desenvolvimento da parte “back-end” do ambiente; o fato do Java ser multiplataforma; o fato do Flex ser um framework para desenvolvimento Web, com eficiência e robustez suficientes para o desenvolvimento da parte “front-end” do ambiente; e o bom aspecto visual proporcionado pela plataforma Flex.

Além disso, duas características fundamentais estiveram presentes na escolha pela plataforma Java para apoiar a construção da camada de serviços da aplicação: os componentes selecionados para fazerem parte do protótipo são construídos com Java; e o fato da API JDBC (Java Database Connectivity) do Java proporcionar muitas facilidades para a conexão com diferentes SGBDs, além de fornecer informações sobre os recursos das base de dados através de metadados - que são disponibilizados por essa API após a conexão com o banco.

Um dos componentes que foram utilizados na construção do protótipo foi o ISOntologyManager (ou, simplesmente, OM). Ele foi cedido gentilmente pelo Instituto Stela (INSTITUTO STELA, 2008), local de trabalho do autor, para auxiliar na viabilização da construção desse protótipo. O OM foi construído para ser um gerenciador de ontologias compatível com a plataforma SBI, e por isso foi escolhido para ter a mesma função nesse protótipo.

Já o GraniteDS, foi selecionado como componente para comunicação remota cliente-servidor, por ser um componente livre e de código aberto, que tem a capacidade de serializar objetos Java e Action Script (linguagem utilizada pela plataforma Flex) e transferi-los do servidor para os clientes, ou vice-versa (utilizando o protocolo AMF, um protocolo binário citado anteriormente). Além disso, o GraniteDS possui um plugin para geração de classes Action Script de forma automática a partir das classes Java correspondentes. Esta solução foi utilizada no presente trabalho, para gerar as classes que representam as ontologias SBI.

3.3.2.3. Considerações gerais sobre o desenvolvimento do protótipo

De uma forma geral, o protótipo pôde ser desenvolvido conforme o planejamento do ambiente proposto. No entanto, algumas mudanças foram realizadas em relação ao que foi previsto, como forma de simplificar o seu desenvolvimento - por se tratar apenas de um protótipo. Dentre essas mudanças estão: a utilização de um arquivo de propriedades no servidor com os dados para conexão no banco, ao invés dessa configuração ser feita via interface; a configuração do esquema utilizado como teste, foi deixada fixa no código; o não suporte a expressões regulares complexas para geração da ontologia de análise (somente à utilização de prefixos simples); e o não suporte ao mapeamento da ontologia ISO com a ontologia de domínio (até pelo fato das ferramentas analíticas da plataforma SBI ainda não suportarem o uso de ontologias de domínio). De qualquer forma, essas funções estarão presentes em futuras versões da ferramenta.

Algumas telas da parte cliente da aplicação desenvolvida podem ser observadas na seção 4.3 deste trabalho, que mostra imagens do protótipo em funcionamento em uma situação de simulação.

Após a concepção, o desenvolvimento e alguns testes primários que foram realizados com essa ferramenta, já se pode fazer uma certa comparação com outras soluções comerciais (ou que são resultados de outras iniciativas acadêmicas) que poderiam ser utilizadas com o mesmo propósito essencial do ambiente construído (manipulação de ontologias).

3.4. Comparação com Iniciativas Acadêmicas e Soluções Comerciais

Não se encontrou no mercado nenhuma ferramenta similar ao ambiente aqui proposto, até pelo fato dele ser pouco genérico - sendo especificado em função da plataforma SBI, se propondo a resolver problemas específicos para manutenção de ontologias dessa arquitetura. De qualquer forma, outras ferramentas para criação e edição de ontologias poderiam ser utilizadas para esse propósito, mas deixariam o processo mais lento e iriam impor mais dificuldades para o usuário final. Isso pode ser concluído pelo fato delas não disponibilizarem formas semi-automáticas para auxiliar os mapeamentos e por não apresentarem características de abstração sobre os detalhes das ontologias. Assim, seria exigido do usuário um maior conhecimento dessas ontologias e um grande esforço no processo de mapeamento e configuração das mesmas (pois o ambiente proposto faz diversas manipulações complexas na ontologia, através de ações simples na interface do usuário). Uma das ferramentas que poderiam ser adotadas, mesmo tendo em vista as dificuldades relatadas, é o Protégé - um dos ambientes mais conhecidos e utilizados para manipular ontologias.

3.4.1. Protégé

O Protégé é um editor de ontologias e um framework para bases de conhecimento. Ele é uma livre e de código aberto. Ele implementa um conjunto rico de estruturas para modelagem de conhecimento e de ações para possibilitar a criação, visualização e manipulação de ontologias em várias formas de representação. Ele também pode ser customizado para possibilitar uma maneira amigável – baseada em termos do domínio da aplicação - de criar modelos de conhecimento e dar entrada com dados. As ontologias do Protégé podem ser exportadas em uma variedade de formatos, incluindo RDF, OWL e XMLSchema. Além disso, o Protégé pode ser estendido, através de uma arquitetura baseada em plug-ins e APIs Java (Application Programming Interface), para construção ou prototipação rápida de ferramentas e aplicações baseadas em conhecimento.

A plataforma Protégé suporta duas principais maneiras de modelar ontologias: através do editor Protégé-Frames e do editor Protégé-OWL.

O editor Protégé-Frames possibilita aos usuários construir e popular ontologias do tipo “*frame-based*” (baseadas em modelos), em conformidade com o protocolo “Open Knowledge Base Connectivity protocol”, ou “OKBC protocol” (ver SRI INTERNATIONAL, 2008). Nesse modelo, uma ontologia consiste em um grupo de classes organizadas em uma hierarquia de classificação para representar os conceitos de um domínio, em um grupo de *slots* associados às classes para descrever as suas propriedades e relações, e em um grupo de instâncias dessas classes – exemplares individualizados dos conceitos, que possuem valores específicos em suas propriedades. Já o editor Protégé-OWL possibilita aos usuários construir ontologias baseadas na Web semântica, particularmente utilizando a linguagem OWL (*Web Ontology Language*) da W3C, que já foi vista em uma seção anterior do presente trabalho. Na figura abaixo, observa-se a imagem de uma tela capturada do Protégé em execução (editor Protégé-OWL).

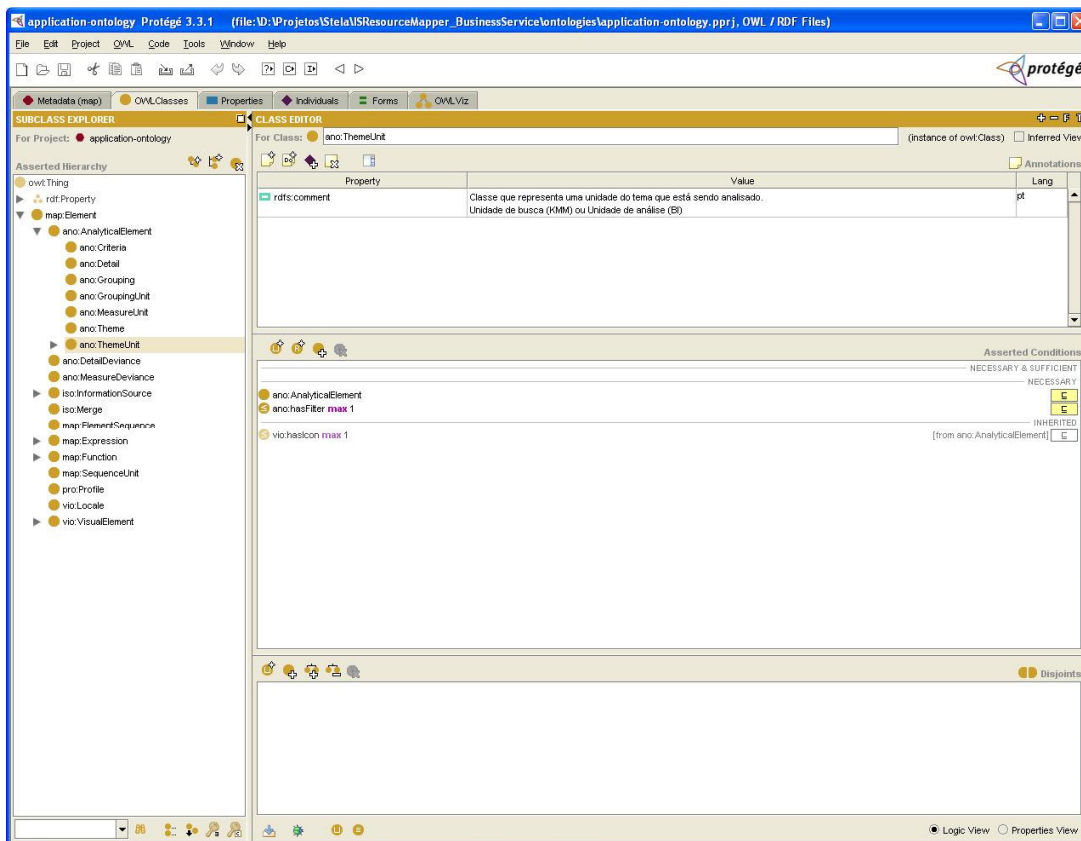


Figura 18 - Imagem do Protégé em execução

O Protégé é apoiado por uma comunidade muito forte de acadêmicos e desenvolvedores, por governos (pelo setor público) e por usuários corporativos, que o utilizam para soluções em gestão do conhecimento em áreas diversas como biomedicina, armazenamento de conhecimento e modelagem corporativa. Na ocasião do presente trabalho, os números retirados do site oficial do Protégé (em 11/10/2008) eram de 104744 usuários registrados e 17246 que fazem parte da lista de discussão para os usuários do Protégé.

3.4.2. WebProtégé

O WebProtégé é uma versão web do Protégé tradicional, que está em desenvolvimento - fase de prototipação (STANFORD, 2008b). Na aplicação de demonstração – disponibilizada para o público em geral (STANFORD, 2008c) – nota-se que apenas o editor Protégé-OWL está disponibilizado. A figura abaixo mostra uma captura de tela dessa aplicação em uso.

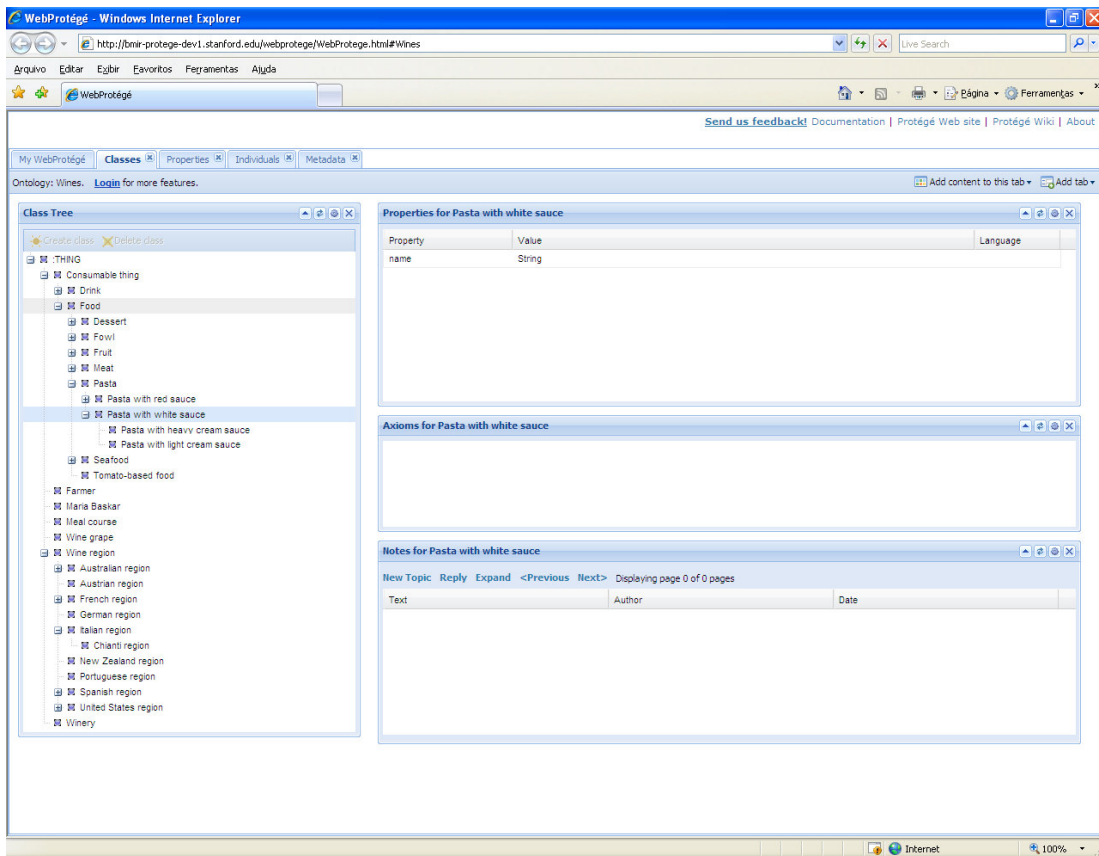


Figura 19 - WebProtégé

Com o WebProtégé se terá um ambiente colaborativo, rodando na Web, para construção de ontologias. A ferramenta proposta no presente trabalho também foi concebida para rodar sobre a Web, facilitando o acesso às ontologias que suportam a plataforma SBI. Futuramente, esse ambiente poderá ser evoluído para possibilitar uma construção colaborativa, com vários usuários interagindo ao mesmo tempo com as ontologias - assim como é a proposta do WebProtégé.

4. Aplicação do Protótipo

Neste capítulo, é feito um estudo de viabilidade e aplicação do ambiente proposto no capítulo anterior, através da apresentação de um protótipo funcional. O objetivo da prototipação é de simular certas funcionalidades da ferramenta proposta. Para isto estarão disponíveis uma base de dados relacional e uma ferramenta analítica, compatível com a plataforma SBI. Essa ferramenta deverá fazer pesquisas sobre a base disponibilizada, utilizando as ontologias configuradas através do protótipo.

4.1. Base de dados utilizada

A base de dados utilizada na aplicação do protótipo é parte de um data warehouse da Plataforma Lattes Institucional – ou PLI - do CNPq (PLATAFORMA LATTES INSTITUCIONAL, 2001), que foi concedida pela Universidade Federal de Santa Catarina para fins de pesquisa. Essa plataforma é um sistema de gestão curricular relacionada à área de Ciência e Tecnologia. Foi utilizado um data mart que contempla o histórico das atuações acadêmicas e profissionais das pessoas da instituição. A parte do modelo dimensional utilizada pode ser verificada na figura abaixo.

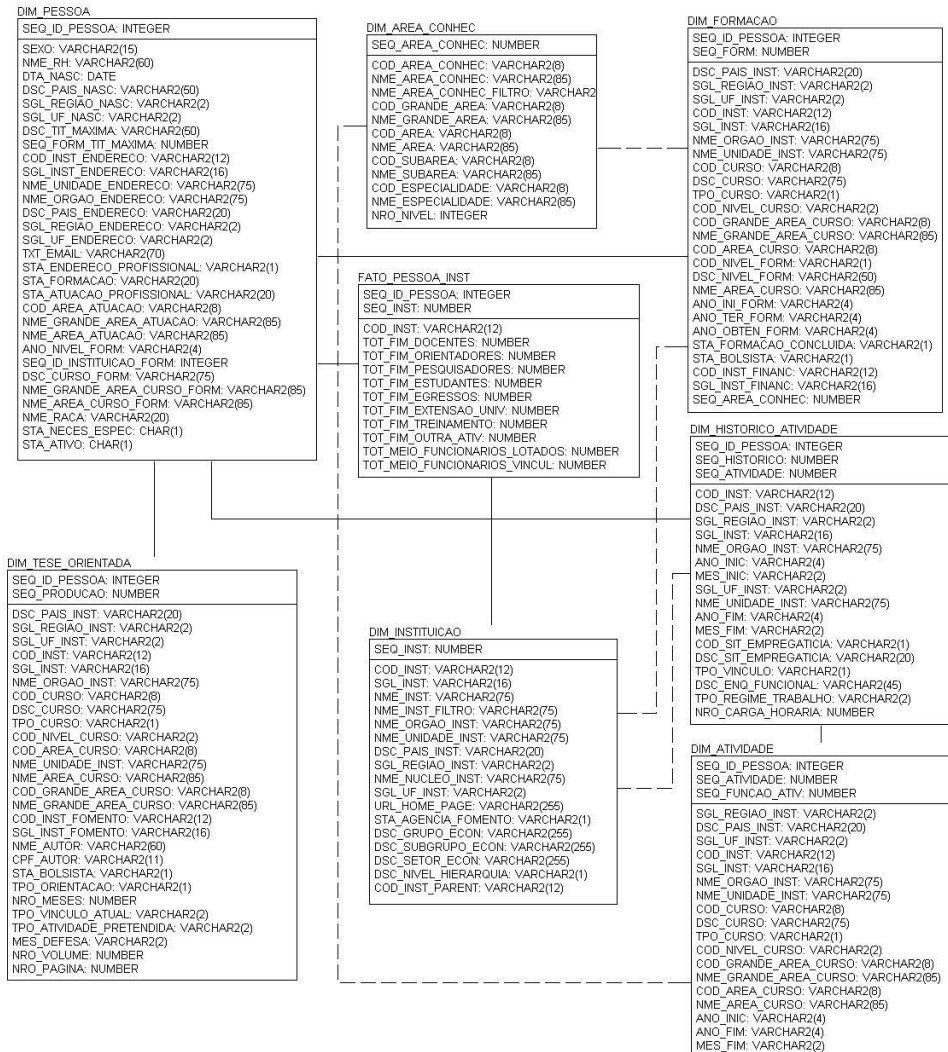


Figura 20 - Parte do modelo dimensional da PLI utilizado na validação do protótipo

4.2. Ferramenta analítica utilizada

Para a realização das pesquisas sobre a base citada anteriormente, utilizando as ontologias configuradas através do protótipo, foi utilizada uma ferramenta analítica, cedida pelo Instituto Stela, que é compatível com a plataforma SBI – o ISExtracta (NAPOLI, 2006; SELL et al, 2008). Segundo Napoli (2006), o objetivo do ISExtracta é guiar o tomador de decisão no processo de localização e combinação de dados para produzir indicadores e extrair conhecimento a partir de repositórios mantidos na organização ou acessíveis através da Web.

4.3. Manipulação das ontologias utilizando o protótipo

A utilização do protótipo para a realização do processo de mapeamento semi-automático das ontologias foi realizado conforme as etapas previstas na seção 3.3.1.3 do presente trabalho (fluxo de execução do processo de mapeamento), levando em consideração as limitações apresentadas na seção 3.3.2.3 (considerações gerais sobre o desenvolvimento do protótipo). A interface do cliente foi dividida em três seções, representadas por três abas: “Ontologia ISO”, “Ontologia Analítica” e “Ontologia de Domínio”. Algumas imagens capturadas durante a execução da aplicação e realização de todo o processo podem ser observadas a seguir.

4.3.1. Manipulação da Ontologia ISO

Na imagem abaixo, observa-se a tela inicial do cliente do protótipo, mostrando a aba “Ontologia ISO”.

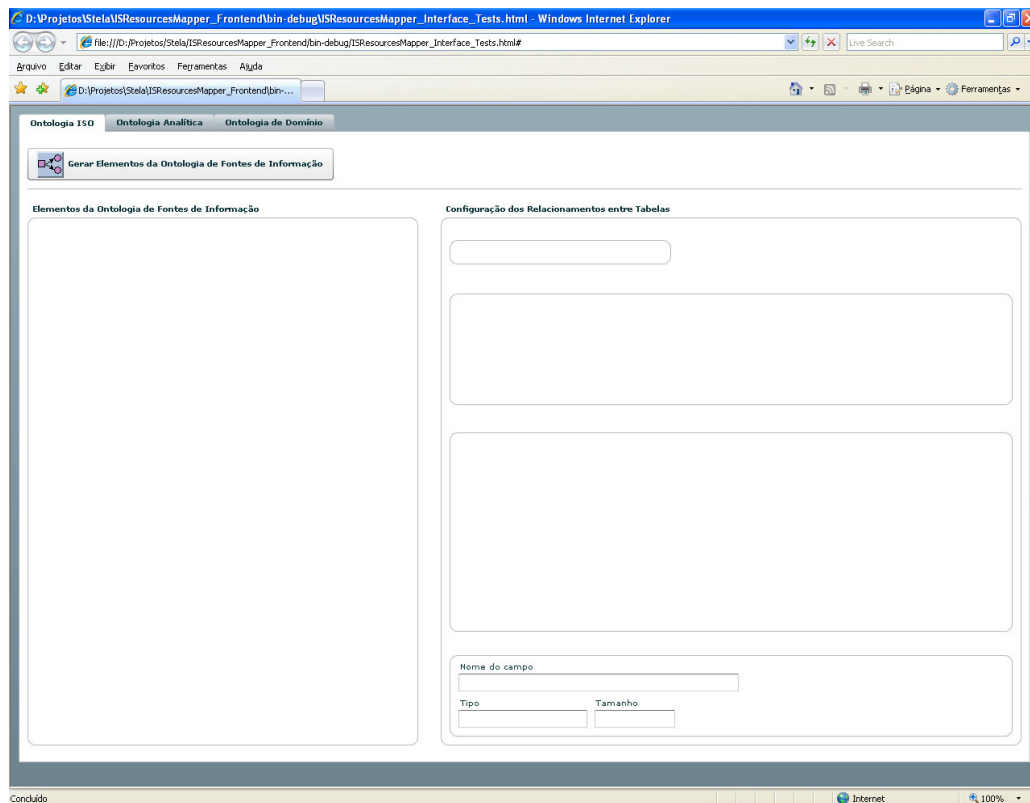


Figura 21 – Tela inicial do protótipo – Aba “Ontologia ISO”

A seguir, é apresentada uma imagem que mostra o resultado do clique no botão “Gerar Elementos da Ontologia de Fontes de Informação”, em que foi acionado o serviço que mapeia os recursos de um esquema da base do PLI (definido a priori) com os conceitos da ontologia de fontes de informação e que retorna as instâncias que representam todas as tabelas e seus relacionamentos e atributos. A interface então apresenta essas instâncias, para que o usuário possa visualizá-las e fazer ajustes sobre elas – como criação de novos relacionamentos ou alterações dos relacionamentos criados automaticamente.

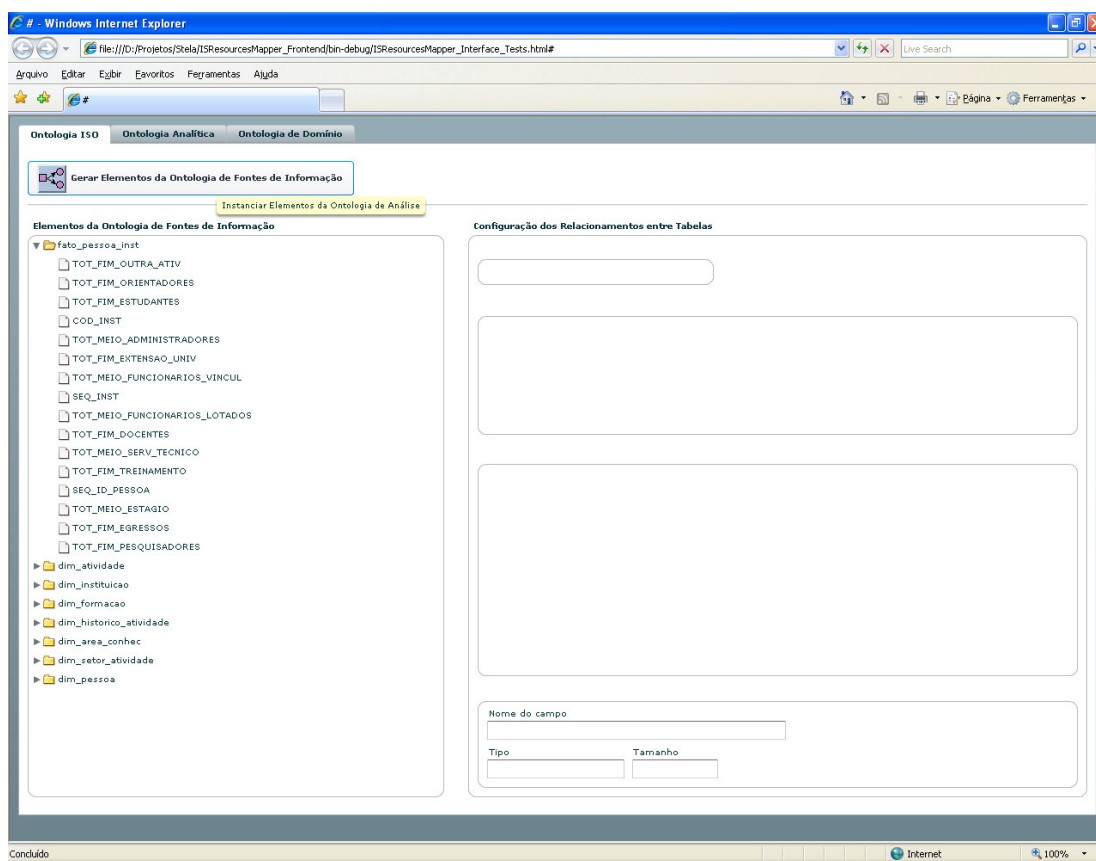


Figura 22 - Resultado do clique no botão para gerar elementos da ontologia ISO

Na próxima imagem, pode-se observar a possibilidade de criação de novos relacionamentos ou de alterações dos relacionamentos criados automaticamente, modificando, por exemplo, o tipo de junção ou os atributos que fazem parte do relacionamento.

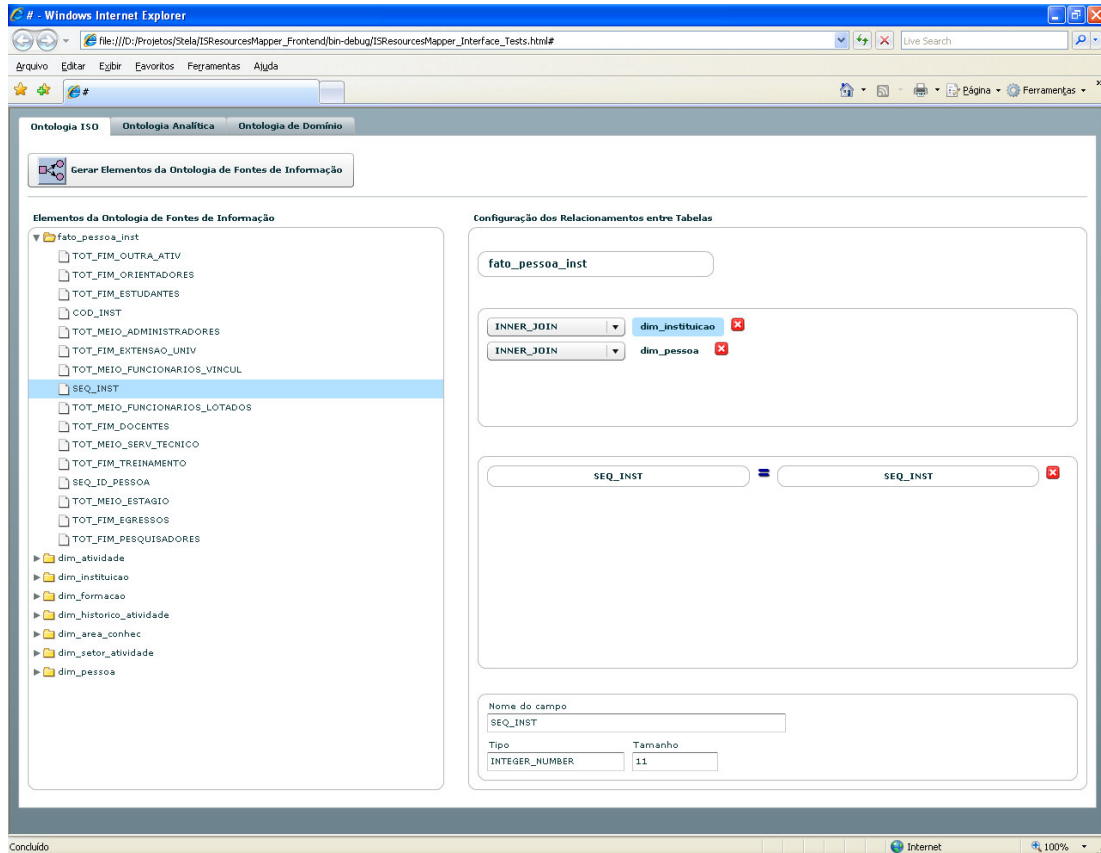


Figura 23 - Possibilidade de configuração dos relacionamentos entre as tabelas

Terminado as configurações das instâncias da ontologia ISO, pode-se iniciar então a manipulação da ontologia de análise ou da ontologia de domínio. Optou-se neste momento pela primeira.

4.3.2. Manipulação da Ontologia de Análise

Esta imagem mostra a tela inicial da aba “Ontologia Analítica”. Pode-se sempre observar do lado esquerdo da tela as instâncias da ontologia ISO (geradas e configuradas anteriormente).

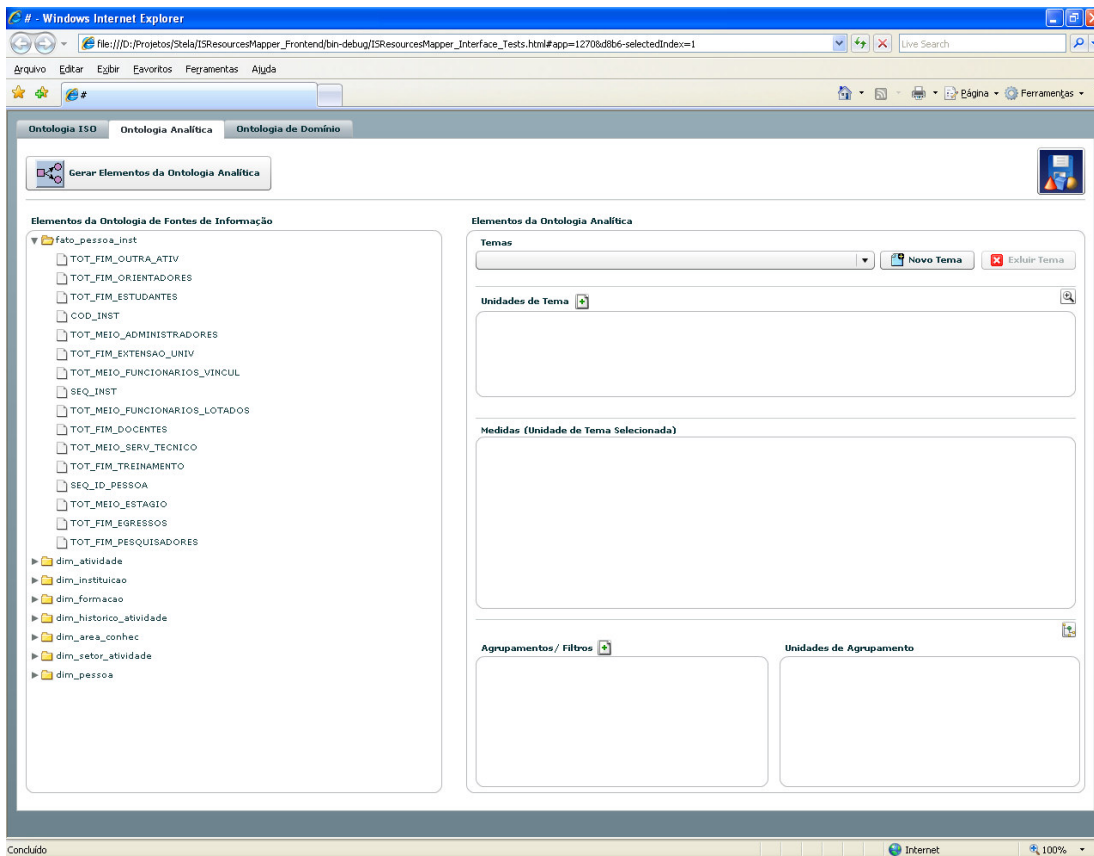


Figura 24 - Aba "Ontologia Analítica"

Após o clique no botão “Gerar Elementos da Ontologia Analítica”, é apresentado a tela a seguir – em que o usuário pode escolher se o mapeamento deve realizado através das heurísticas do sistema (escolhendo a opção “automaticamente”) ou utilizando expressões criadas pelo próprio cliente (escolhendo a opção “manualmente”). Definido isso, ao clique no botão “OK”, é feita a invocação do serviço que faz o mapeamento automático das instâncias da ontologia ISO com os conceitos da ontologia.

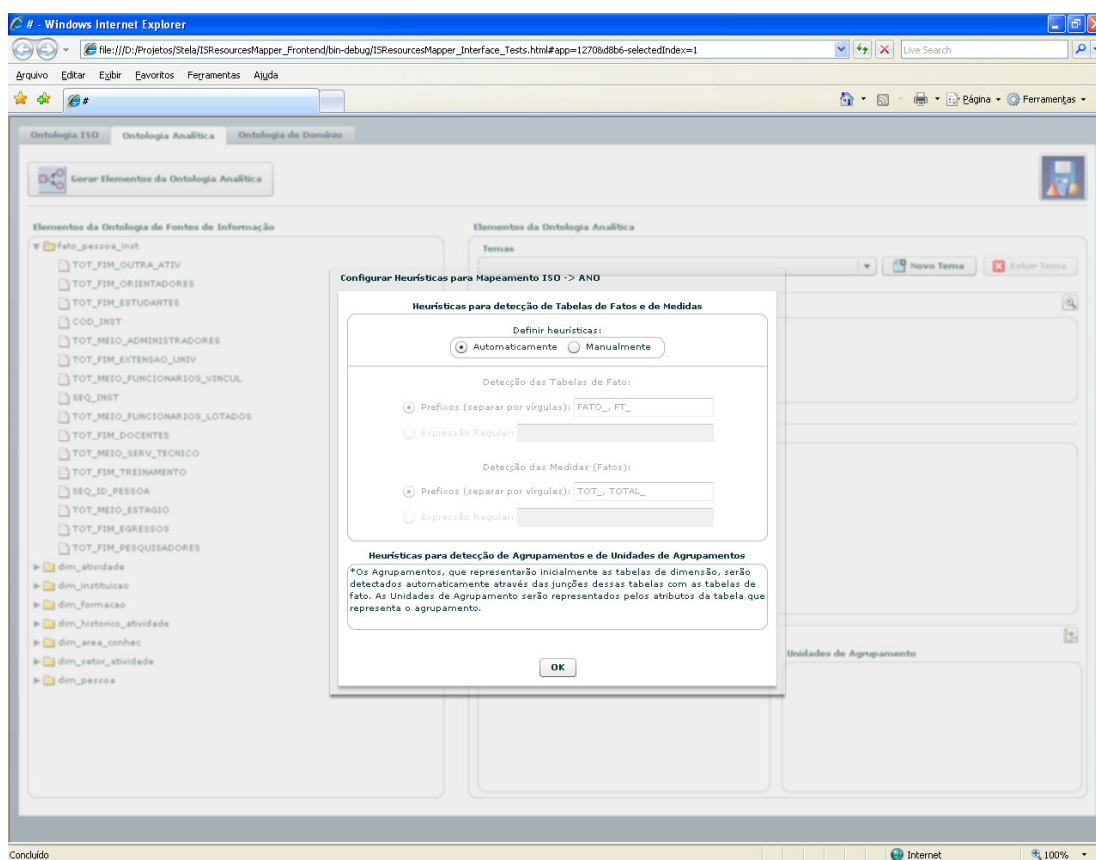


Figura 25 - Tela para definição das heurísticas utilizadas no mapeamento

É retornado então uma instância padrão da classe “Theme”, a “Theme.Default”, que contem todas as instâncias dos conceitos da ontologia analítica derivados dessa classe. A GUI então apresenta essas novas instâncias, para que o usuário possa visualizá-las e também fazer ajustes manuais sobre elas como, por exemplo, a criação de novos temas, unidades de temas e agrupamentos, e a modificação das instâncias já existentes (geradas automaticamente).

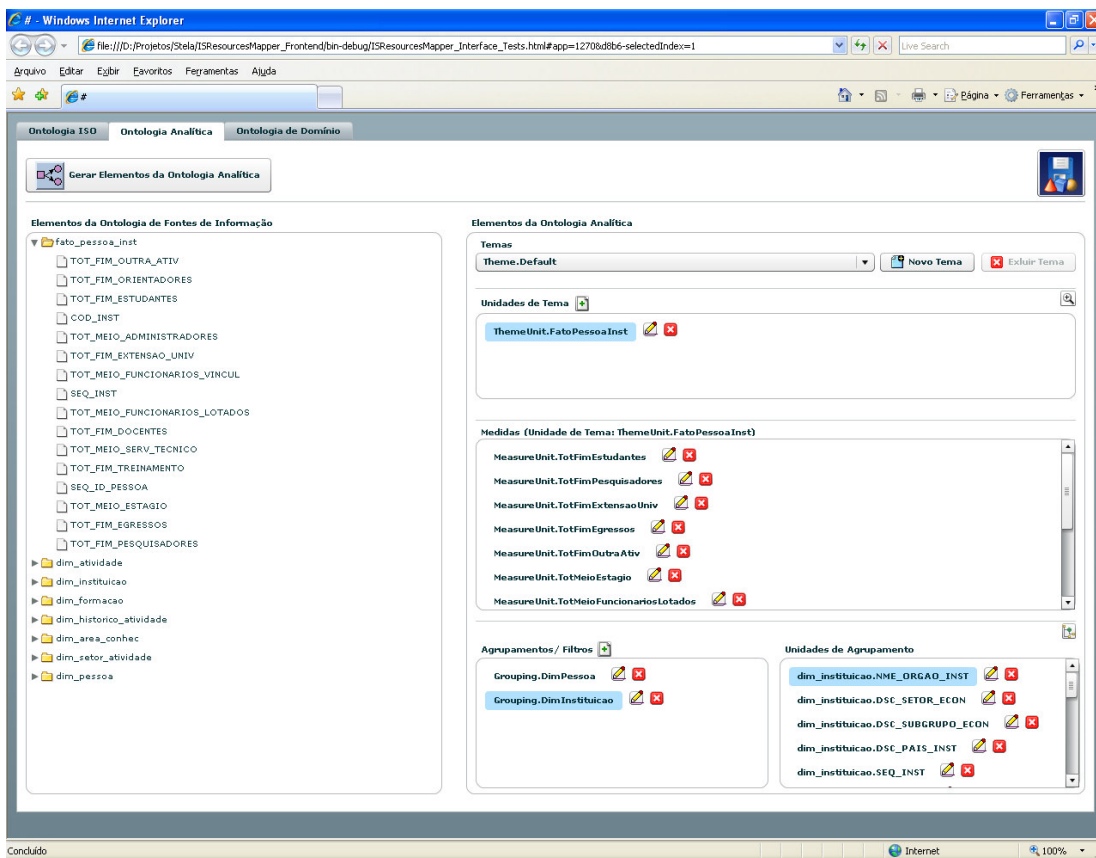


Figura 26 - Possibilidades de configuração da Ontologia Analítica

Para criar instâncias novas da ontologia analítica, basta arrastar e soltar as instâncias da ontologia ISO nos compartimentos adequados (“drag and drop”). Essas instâncias recebem um nome padrão, que pode ser alterado em seguida.

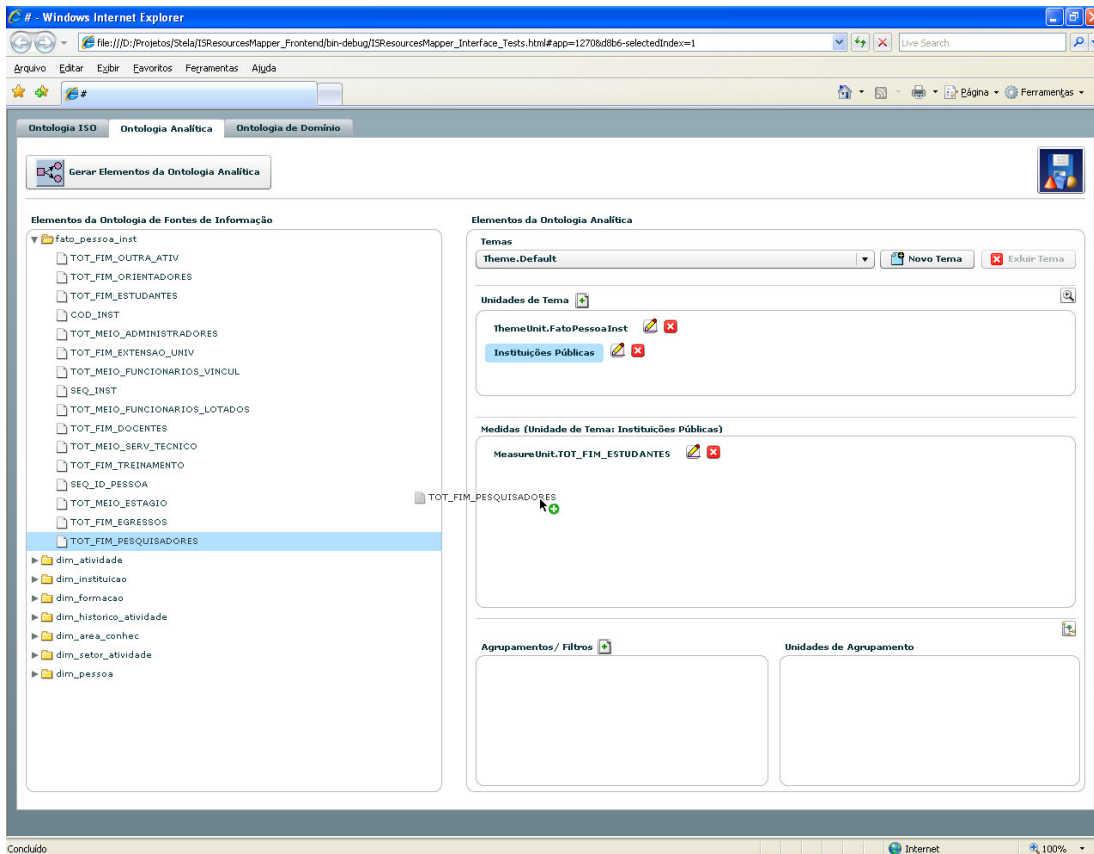


Figura 27 - Ilustração da capacidade de "drag and drop" da aplicação

Além disso, clicando-se na lupa do lado superior direito da tela, pode-se definir os detalhes (“Details”) das unidades de tema selecionada, conforme visto na figura abaixo.

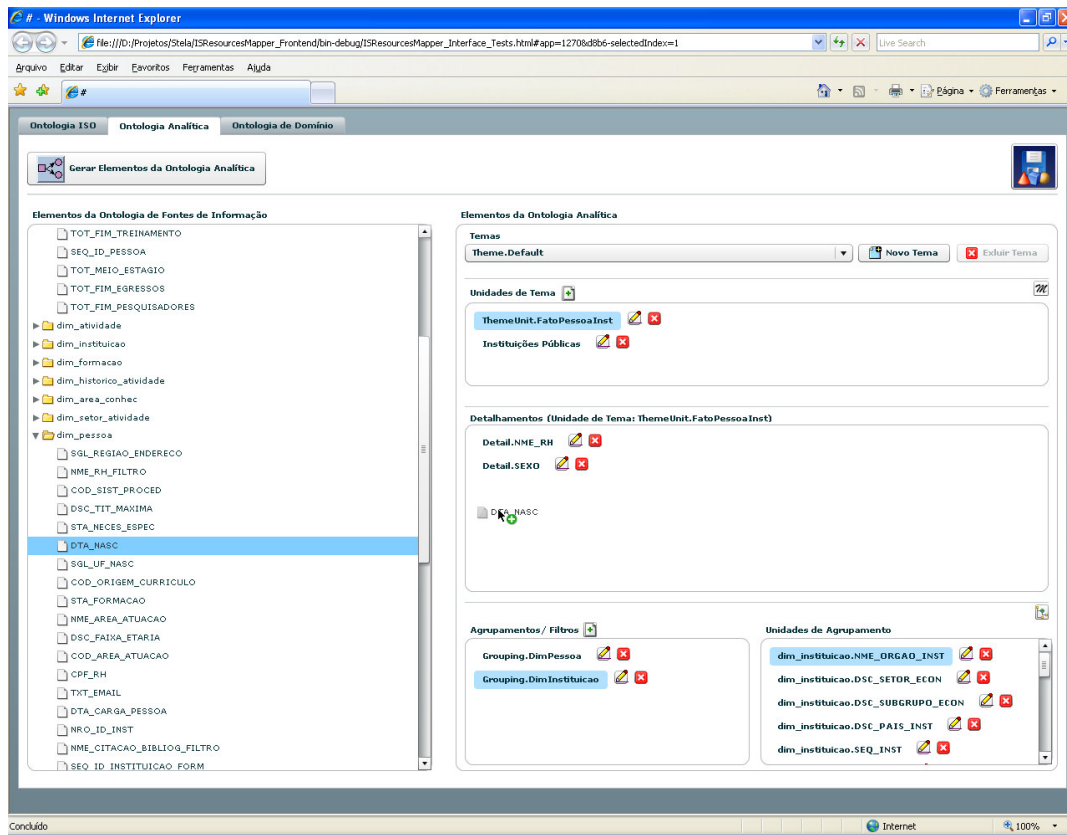


Figura 28 - Configuração dos detalhes da Unidade de Tema selecionada

E clicando-se no botão que se encontra no canto inferior direito, pode-se definir a hierarquia de atributos, para suportar operações de *drill-up* e *drill down*. A figura abaixo mostra a definição de que “Código da Grande Área” é pai de “Código da Área de Conhecimento”, que é pai de “Código da Sub-área”.

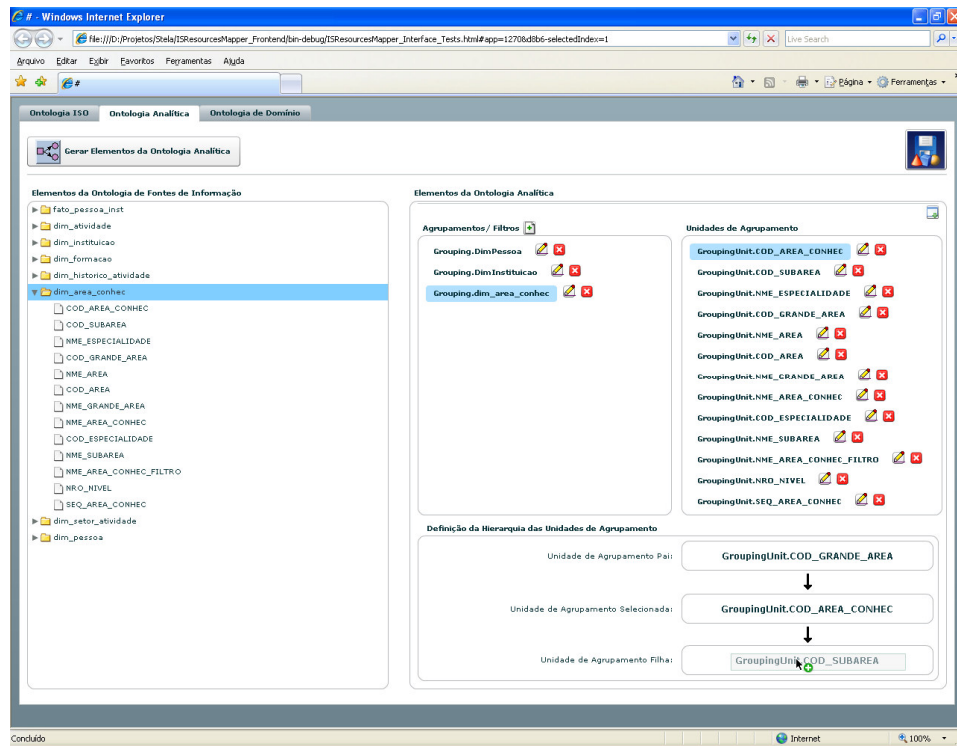


Figura 29 - Configuração de uma hierarquia de atributos

Terminada a etapa de configuração da ontologia de análise, pode-se então realizar o mapeamento manual das instâncias da primeira ontologia com os conceitos da ontologia de domínio, clicando-se na próxima aba.

4.3.3. Manipulação da Ontologia de Domínio

A figura a seguir mostra a aba “Ontologia de Domínio”. Nesta área, pode-se carregar a representação gráfica de uma ontologia de domínio de aplicação e realizar o mapeamento de seus conceitos com as instâncias da ontologia ISO (sempre presentes no lado esquerdo da interface). A interface provê um modo simples de navegar nos conceitos da ontologia, que podem ser rearranjados (para mudarem de lugar no espaço disponível) e apresentados conforme o grau de separação hierárquica escolhido. Para fins de construção desse protótipo, foi apresentada apenas a simulação de como poderia ser a interação com uma ontologia de domínio e a realização do mapeamento dos conceitos dessa ontologia com as instâncias da ontologia de fontes de informação.

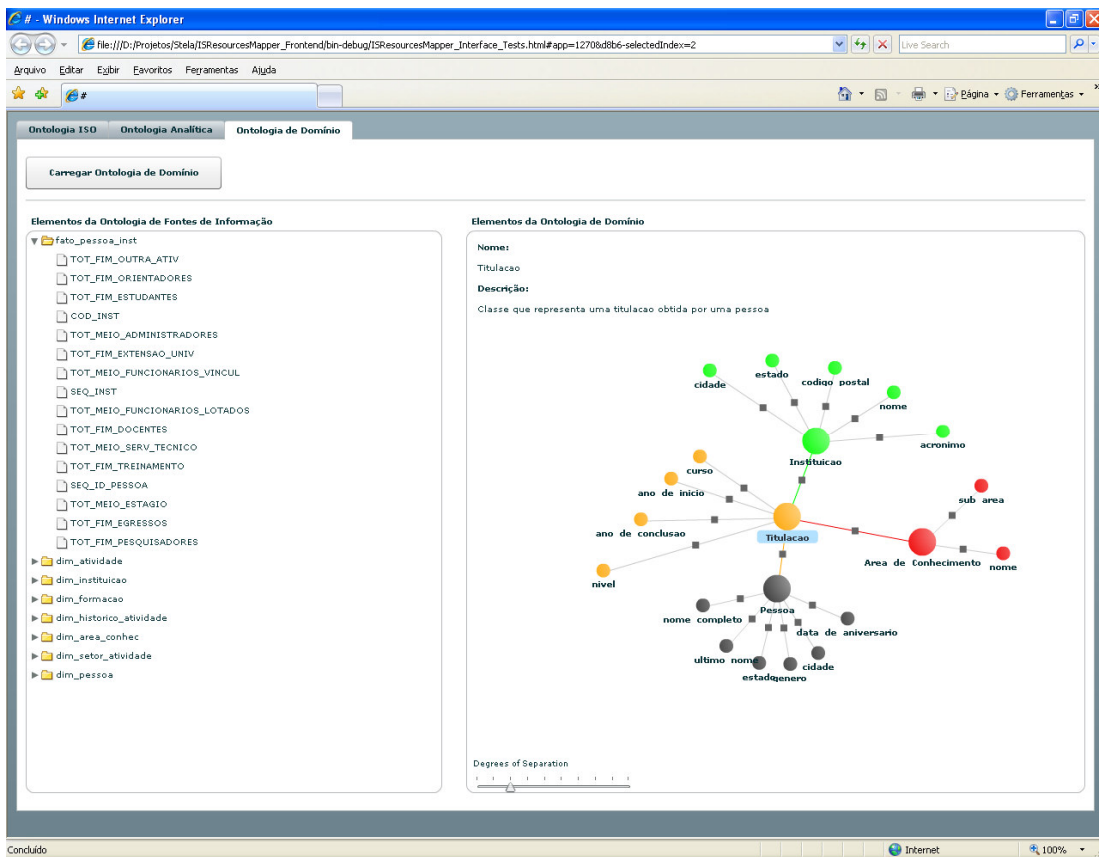


Figura 30 - Aba "Ontologia de Domínio"

4.4. Ferramenta analítica utilizando as ontologias configuradas com o auxílio do protótipo

Após o término do processo de mapeamento e a persistência das ontologias geradas, configurou-se o ISExtracta para utilizar essas ontologias e para fazer a conexão na base do PLI (através da configuração de arquivos de propriedades).

A interface do ISExtracta utiliza os conceitos da ontologia de BI, apresentando e organizando os dados disponibilizados pelas instâncias dessa ontologia. Dessa forma, pôde-se perceber claramente o sucesso na etapa de geração e configuração dessas instâncias. Algumas imagens capturadas durante a utilização dessa ferramenta podem ser observadas abaixo.

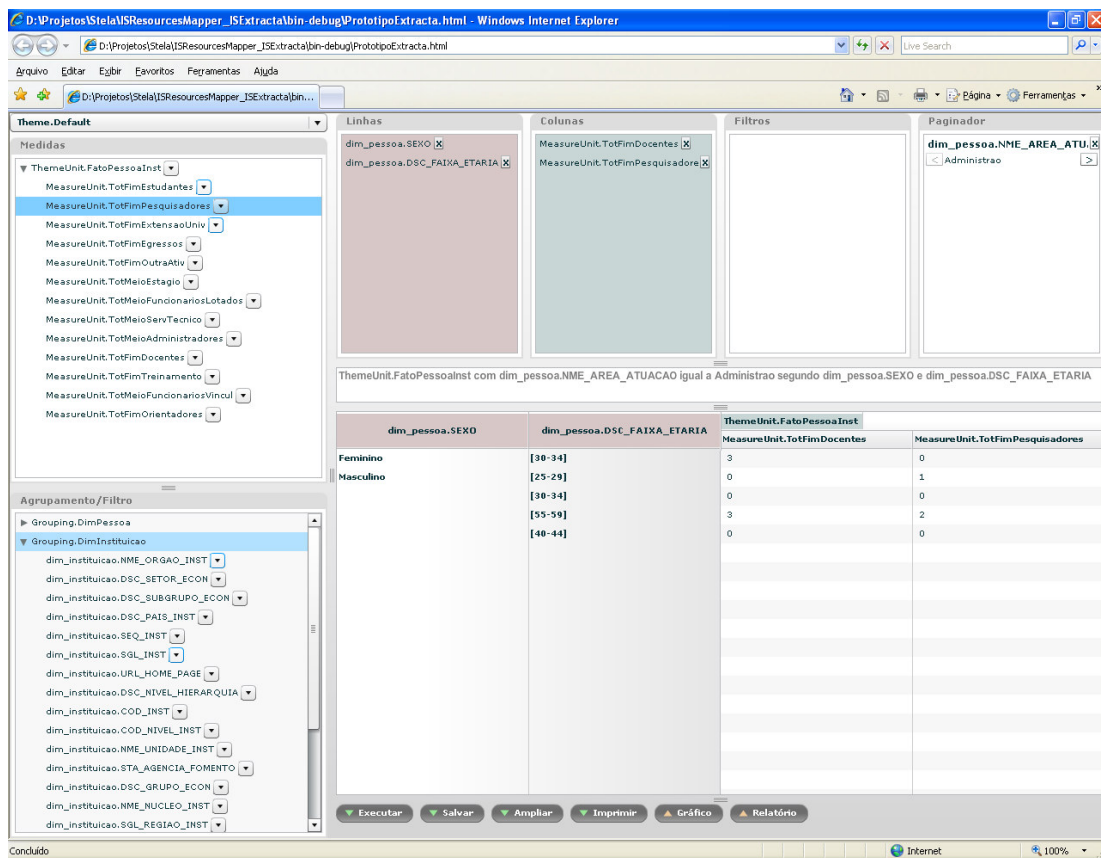


Figura 31 - Utilização do ISExtracta para validar os mapeamentos realizados

4.5. Considerações sobre a aplicação do ambiente

No estudo de viabilidade e aplicação do ambiente proposto, verifica-se, conforme foi previsto, a relativa facilidade na manipulação das ontologias SBI - pela abstração que a ferramenta faz sobre elas, tornando o processo mais simples e intuitivo possível.

5. Considerações Finais

De uma forma geral, considerou-se que os objetivos estabelecidos para o presente trabalho foram atingidos, seguindo a metodologia apresentada. Foram identificadas as etapas necessárias para a manutenção das ontologias da Plataforma SBI (uma solução de BI semântico) e projetou-se um ambiente para facilitar este processo – o que pode auxiliar na implantação dessa solução nas organizações. Para a validação do ambiente proposto, desenvolveu-se um protótipo aplicado em uma simulação real. Nesta simulação, que fez uso de um data warehouse real, realizou-se as configurações das ontologias por meio da ferramenta desenvolvida e fez-se uso de uma ferramenta analítica compatível com a plataforma SBI para fazer análises no DW, utilizando as ontologias configuradas.

5.1. Contribuições

Através da utilização do ambiente apresentado, o usuário - possivelmente um analista de negócios ou engenheiro do conhecimento - pode manipular as diferentes ontologias da Plataforma SBI utilizando somente recursos visuais, de uma maneira simples e intuitiva, sem necessidade de conhecimentos computacionais avançados. Dessa forma, pode ser facilitada a implantação e a manutenção dessa plataforma nas organizações. Este trabalho serve também para demonstrar que, através do uso de um ferramental adequado, é possível facilitar a utilização de soluções baseadas em tecnologias semânticas – algumas complexidades inerentes a esse tipo de soluções, como a manipulação de

ontologias citada no presente trabalho, podem ficar transparentes para o usuário final.

5.2. Limitações e Trabalhos Futuros

Para o propósito inicial deste trabalho, a realização do mapeamento semi-automático dos recursos de diferentes repositórios de dados com os conceitos da ontologia de fontes de informação, suportou-se apenas o uso de repositório estruturado, do tipo relacional. Ainda para essa etapa, somente realizou-se mapeamentos diretos entre os diferentes recursos, ou seja, do tipo “um-para-um”. Em futuras versões da ferramenta, além de permitir o uso conjunto de múltiplos repositórios – estruturados e semi-estruturados - estão previstos mapeamentos mais complexos, como “muitos-para-um” ou “um-para-muitos”. Assim, estão previstas operações como agregação de mais de um recurso dos repositórios de dados e o mapeamento desse agregado com apenas um conceito da ontologia de fontes de informação, ou ainda cortes (filtros) sobre um único recurso e o mapeamento dos mesmos em diferentes conceitos da ontologia. Para isso, o uso de expressões mais complexas e de funções para realização dos mapeamentos devem ser implementadas.

O mapeamento das instâncias da ontologia de fontes de informação com os conceitos de ontologias de domínio de aplicação apresentou-se apenas uma simulação de como poderia ser a interação desse mapeamento. Dessa forma, futuras versões da ferramenta poderão realmente possibilitar esse processo. Pode-se suportar também a utilização de mais de uma ontologia de domínio simultaneamente, através da possibilidade de mapear os seus conceitos (mapeamento de conceitos equivalentes).

De uma forma geral, é relevante a realização de um projeto gráfico para o ambiente, buscando apresentar as informações para os usuários de uma maneira simplista – garantindo mais usabilidade para a ferramenta. Também é necessário adequar a ferramenta para uso em outros idiomas, através da sua internacionalização - já que a Plataforma SBI, que embasou este desenvolvimento, tem como foco usuários de diversas nacionalidades.

6. Referências Bibliográficas

ADOBE SYSTEMS INCORPORATED. **AMF 3 Specification**. 2008a. Disponível em: <http://opensource.adobe.com/wiki/download/attachments/1114283/amf3_spec_05_05_08.pdf>. Acesso em: 10 out. 2008.

ADOBE SYSTEMS INCORPORATED. **Adobe LiveCycle Data Services ES**. 2008b. Disponível em: <<http://www.adobe.com/products/livecycle/dataservices/>>. Acesso em: 10 out. 2008.

ADOBE SYSTEMS INCORPORATED. **Flex Builder 3 features**. 2008c. Disponível em: <http://www.adobe.com/products/flex/features/flex_builder/>. Acesso em: 30 ago. 2008.

ADEQUATE SYSTEMS. **Granite Data Services** (Free, Open Source, Flex & J2EE). Disponível em: <<http://www.graniteds.org/>>. Acesso em: 10 out. 2008.

ANZANELLO, Cynthia Aurora. **OLAP Conceitos e Utilização**. 2002. Disponível em: <http://www.inf.ufrgs.br/~clesio/cmp151/cmp15120021/artigo_cynthia.pdf>. Acesso em: 07 set. 2008.

BECKETT, Dave; MCBRIDE, Brian (Ed.). **RDF/XML Syntax Specification (Revised)**: W3C Recommendation 10 February 2004. Disponível em: <<http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/>>. Acesso em: 20 set. 2008.

BERNERS-LEE, Tim; HANDLER, James; LASSILA, Ora. **The Semantic Web**: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American, v.2, n.1, p.1-12, May 2001. Disponível em: <<http://www.sciam.com/article.cfm?id=the-semantic-web>>. Acesso em: 02 ago. 2008.

BERNERS-LEE, Tim. **Uniform Resource Identifier (URI): Generic Syntax**. Request for Comments: 3986. January 2005. Disponível em: <<http://labs.apache.org/webarch/uri/rfc/rfc3986.html>>. Acesso em: 10 ago. 2008.

BRAY, Tim et al. (Ed.). **Extensible Markup Language (XML) 1.0 (Fourth Edition)**: W3C Recommendation 16 August 2006, edited in place 29 September 2006. Disponível em: <<http://www.w3.org/TR/2006/REC-xml-20060816/>>. Acesso em: 28 set. 2008.

BRICKLEY, Dan; MCBRIDE, Brian (Ed.). **RDF Vocabulary Description Language 1.0**: RDF Schema. W3C Recommendation 10 February 2004. Disponível em: <<http://www.w3.org/TR/2004/REC-rdf-schema-20040210/>>. Acesso em: 20 set. 2008.

CARVALHO, Isamir Machado de; MENDES, Sérgio Peixoto; VERAS, Vivianne Muniz (Org.). **Gestão do Conhecimento: uma estratégia empresarial**. Brasília, DF: J.J Gráfica e Comunicação Ltda., 2006. 348 p.

CODY, W. F. et al. The Integration of Business Intelligence and Knowledge Management. **IBM Systems Journal**, p. 697-713. 2002. Disponível em: <<http://www.research.ibm.com/journal/sj/414/cody.pdf>>. Acesso em: 18 out. 2008.

DAVID, Paul A.; FORAY, Dominique. Economic Fundamentals of the Knowledge Society. **Policy Futures In Education**, Oxford, p. 20-49. jan. 2003. Disponível em: <http://www.worlds.co.uk/pdf/freetoview.asp?j=pfie&vol=1&issue=1&year=2003&article=2_David_PFIE_1_1>. Acesso em: 13 out. 2008.

DAVIES, John; FENSEL, Dieter; HARMELEN, Frank Van (Ed.). **Towards the Semantic Web: Ontology-Driven Knowledge Management**. Chichester, West Sussex, England: John Wiley & Sons, 2003. 288 p.

DECKER, Stefan et al. The Semantic Web: The Roles of XML and RDF. **Ieee Internet Computing**, v. 5, n. 4, p.63-74, set./out. 2000. Bimestral. Disponível em: <<http://ieeexplore.ieee.org/iel5/4236/18994/00877487.pdf?arnumber=877487>>. Acesso em: 02 out. 2008.

FALLSIDE, David C.; WALMSLEY, Priscilla. **XML Schema Part 0: Primer Second Edition**. W3C Recommendation 28 October 2004. Disponível em: <<http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/>>. Acesso em: 30 ago. 2008.

GOMEZ-PEREZ, A.; CORCHO, O.; FERNANDEZ-LOPEZ, M. **Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web**. London: Springer-Verlag, 2004.

GONZAGA, Tatiana Sousa. **Uma metodologia para o desenvolvimento de instrumentos de análise multidimensional da informação em projetos de governo eletrônico voltados ao cidadão**. 2005. 130 f. Dissertação (Mestrado em Engenharia de Produção). Programa de Pós-graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis, 2005.

GRUBER, Thomas R.. **A Translation Approach to Portable Ontology Specifications**. Knowledge System Laboratory, Stanford University, 1993. Disponível em: <<http://tomgruber.org/writing/ontolingua-kaj-1993.pdf>>. Acesso em: 20 out. 2008.

GUARINO, Nicola. **Formal Ontology and Information Systems**. 1998.

HAJNYSZ, Mateusz. **Next Generation Business Intelligence for Small and Mid-size Enterprises: Adoption, Preferences and Offers in Poland**. 2007.

152 f. Dissertação (Mestrado) - Center For Information And Communication Technologies (cict), Technical University Of Denmark, Kongens Lyngby, 2007. Disponível em: <http://www.imm.dtu.dk/pubdb/views/edoc_download.php/5396/pdf/imm5396.pdf>. Acesso em: 17 out. 2008.

HAYES, Patrick; MCBRIDE, Brian (Ed.). **RDF Semantics**. W3C Recommendation 10 February 2004. Disponível em: <<http://www.w3.org/TR/2004/REC-rdf-mt-20040210/>>. Acesso em: 20 set. 2008.

HEWLETT-PACKARD DEVELOPMENT COMPANY, LP. **Jena: A Semantic Web Framework for Java**. 2008. Disponível em: <<http://jena.sourceforge.net/license.html>>. Acesso em: 20 ago. 2008.

INMON, W. H.. **Building the Data Warehouse**. Fourth Edition, Indianapolis, IN: Wiley Publishing, Inc., 2005. 541 p. ISBN-13: 978-0-7645-9944-6.

INSTITUTO STELA. **Instituto Stela**. Disponível em: <<http://portal.stela.org.br/>>. Acesso em: 31 out. 2008.

KALFOGLOU, Yannis, SCHORLEMMER, Marco. **Ontology Mapping: the state of the art**. The Knowledge Engineering Review, 18 (1). pp. 1-31., 2003.

KIFER, Michael, BRUIJN, Jos de, BOLEY, Harold, FENSEL, Dieter. **A Realistic Architecture for the Semantic Web**. RuleML 2005 Conference Proceedings, Galway, Ireland, 2005. Springer LNCS 3791. pp. 17-29. NRC 48533. Disponível em: <<http://iti-iti.nrc-cnrc.gc.ca/iti-publications-iti/docs/NRC-48533.pdf>>. Acesso em: 05 out. 2008.

KIMBALL, Ralph et al. **The Data Warehouse Lifecycle Toolkit**. 2nd Ed, Indianapolis, IN: Wiley Publishing, Inc., 2008. ISBN: 978-0-470-14977-5.

KLYNE, Graham; CARROLL, Jeremy J.; MCBRIDE, Brian. **Resource Description Framework (RDF): Concepts and Abstract Syntax**. W3C Recommendation 10 February 2004. Disponível em: <<http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>>. Acesso em: 20 ago. 2008.

LOGIXML, INC.. **The Next Generation of BI**. White Paper. 2007. Disponível em: <<http://www.logixml.com/rdPage.aspx?rdReport=WhitepaperBI2>>. Acesso em: 17 out. 2008.

MARCHAL, Benoît. **XML Conceitos e Aplicações**. São Paulo: Berkeley, 2000. ISBN: 85-7251-564-X.

MARSHALL, Rosalie. **Gartner predicts slow growth in BI during 2008**. IT Week. 2008. Disponível em: <<http://www.computing.co.uk/itweek/news/2207083/gartner-predicts-slow-growth-bi>>. Acesso em: 15 ago. 2008.

MOATS, R.. **URN Syntax**. Request for Comments: 2141. May 1997. Disponível em: <<http://www.ietf.org/rfc/rfc2141.txt>>. Acesso em: 20 ago. 2008.

MORT BAY CONSULTING. **Jetty**. Disponível em: <<http://www.mortbay.org/jetty/>>. Acesso em: 10 out. 2008.

NAPOLI, Márcio et al. **Um framework para concepção de ferramentas de apoio à decisão baseadas em ontologias**. 15f. Instituto Stela e Programa de Pós-Graduação em Engenharia e Gestão do Conhecimento da Universidade Federal de Santa Catarina, Florianópolis, 2006.

PINHEIRO, Carlos André Reis. **Inteligência Analítica: Mineração de Dados e Descoberta de Conhecimento**. Rio de Janeiro, RJ: Ciência Moderna Ltda., 2008. 397 p.

PLATAFORMA LATTES INSTITUCIONAL, CNPq. Desenvolvido pelo Grupo Stela, 2001. Disponível em <<http://lattes.ufsc.br>>. Acesso em: 20 de out. 2008.

RADEN, Neil. **Business Intelligence 2.0: Simpler, More Accessible, Inevitable**. 01/02/2007. Disponível em: <<http://www.intelligententerprise.com/showArticle.jhtml;jsessionId=KAK15UEUQBOGWQSNDLRCKH0CJUNN2JVN?articleID=197002610>>. Acesso em: 17 out. 2008.

SELL, Denilson. **Uma Arquitetura para Business Intelligence Baseada em Tecnologias Semânticas para Suporte a Aplicações Analíticas**. 2006. 210 f. Tese (Doutorado) - Programa de Pós-graduação em Engenharia de Produção, Universidade Federal de Santa Catarina, Florianópolis, SC, 2006.

SELL, Denilson; SILVA, Dhiogo Cardoso da; BEPPLER, Fabiano Duarte, NAPOLI, Marcio; GHISI, Fernando Benedet; PACHECO, Roberto C. S.; TODESCO, José Leomar. SBI: A Semantic Framework to Support Business Intelligence. In: FIRST INTERNATIONAL WORKSHOP ON ONTOLOGY-SUPPORTED BUSINESS INTELLIGENCE (OBI2008), 978-1-60558-219-11008, 2008, Karlsruhe, Germany. **ACM Proceedings**. Karlsruhe: Acm Digital Library, 2008. p. 1 - 11. (no prelo)

SILVA, Dhiogo Cardoso da. **Aplicação de Ontologias para o Suporte ao Processamento ETL em Soluções de Business Intelligence**. 2006. 83 f. Trabalho de Conclusão de Curso (Graduação) - Curso de Sistemas de Informação, Departamento de Informática e Estatística, Universidade Federal de Santa Catarina, Florianópolis, SC, 2006.

SILVA, Geiza C.; LIMA, Tarcízio de Souza. **RDF e RDFS na Infra-estrutura de suporte à Web Semântica**. Revista Eletrônica de Iniciação Científica. Ano II. Volume II. Número 1. 2002. ISSN 1519-8219. Disponível em: <http://www.sbc.org.br/reic/edicoes/2002e1/>. Acessado em: 25 set. 2008.

SMITH, Michael K.; WELTY, Chris; MCGUINNESS, Deborah L. (Ed.). **OWL Web Ontology Language Guide: W3C Recommendation 10 February 2004**. Disponível em: <<http://www.w3.org/TR/2004/REC-owl-guide-20040210/>>. Acesso em: 22 ago. 2008.

SRI INTERNATIONAL. **Open Knowledge Base Connectivity**. Disponível em: <<http://www.ai.sri.com/~okbc/>>. Acesso em: 10 out. 2008.

STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH (Stanford - Califórnia). **Protégé**. Disponível em: <<http://protege.stanford.edu>>. Acesso em: 11 out. 2008.

STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH (Stanford - Califórnia). **WebProtégé**. Disponível em: <<http://protegewiki.stanford.edu/index.php/WebProtege>>. Acesso em: 11 out. 2008(b).

STANFORD CENTER FOR BIOMEDICAL INFORMATICS RESEARCH (Stanford - Califórnia). **WebProtégé - Demo**. Disponível em: <<http://bmir-protege-dev1.stanford.edu/webprotege/WebProtege.html>>. Acesso em: 11 out. 2008(c).

SUN MICROSYSTEMS. **Java 2 Platform, Enterprise Edition (J2EE) Overview**. 2008a. Disponível em: <<http://java.sun.com/j2ee/overview.html>>. Acesso em: 10 out. 2008.

SUN MICROSYSTEMS. **Java SE at a Glance**. 2008b. Disponível em: <<http://java.sun.com/javase/>>. Acesso em: 10 out. 2008.

THE ECLIPSE FOUNDATION. **Eclipse**. 2008. Disponível em: <<http://www.eclipse.org/>>. Acesso em: 10 out. 2008.

UNICODE, INC. **The Unicode® Standard: A Technical Introduction**. Disponível em: <<http://www.unicode.org/standard/principles.html>>. Acesso em: 20 set. 2008.

ANEXO A - Código-fonte do protótipo

Mapper_Frontend (cliente flex)

Mapper_Main.mxml

```
<?xml version="1.0" encoding="utf-8"?>

<!-- Classe principal, que atua como um controlador geral -->
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute"
    xmlns:mapper="br.ufsc.inf.feco.mapper.view.**"
    preinitialize="preinit()" initialize="init()">

    <mx:Script><![CDATA[

        import mx.core.IFlexDisplayObject;
        import mx.managers.PopUpManager;
        import br.ufsc.inf.feco.mapper.view.popups.PanelHeuristicsConfigSimple;
        import br.ufsc.inf.feco.mapper.events.PanelHeuristicsConfigExecute;
        import br.ufsc.inf.feco.mapper.view.popups.LoadAggregateFunctionInstances;
        import br.ufsc.inf.feco.mapper.view.popups.GenerateBIOntologyInstances;
        import br.ufsc.inf.feco.mapper.view.popups.LoadISOInstances;
        import br.ufsc.inf.feco.mapper.view.popups.LoadMergesInstances;
        import br.ufsc.inf.feco.mapper.view.popups.LoadBIOInstances;
        import br.ufsc.inf.feco.mapper.model.MapperModelLocator;
        import br.ufsc.inf.feco.mapper.model.MapperModel;
        import br.ufsc.inf.feco.mapper.view.popups.GenerateISOInstances;
        import br.ufsc.inf.feco.mapper.view.popups.LoadLocalesInstances;
        import br.ufsc.inf.feco.ontologymanager.OntologyManager;
        import br.ufsc.inf.feco.mapper.events.GetOntologyInstancesEvent;
        import br.ufsc.inf.feco.mapper.business.MapperService;

        private var _mapperModel: MapperModel = MapperModelLocator.getInstance().mapperModel;

        private function preinit(): void{
            Security.loadPolicyFile("http://192.168.0.250:8080/isbi/crossdomain.xml");
        }

        private function init(): void{
            addEventListener(GetOntologyInstancesEvent.GENERATE_ISO, getOntologyInstancesEventGenerateISOHandler);
            addEventListener(GetOntologyInstancesEvent.GENERATE_BI, getOntologyInstancesEventGenerateBIHandler);
            addEventListener(GetOntologyInstancesEvent.LOAD_ALL, getOntologyInstancesEventLoadAllHandler);
        }

        private function getOntologyInstancesEventGenerateISOHandler(event: GetOntologyInstancesEvent): void{
            var dom : GenerateISOInstances = new GenerateISOInstances();
            dom.addEventListener(Event.COMPLETE, downloadISOntologyIntancesCompleted);
            PopUpManager.addPopUp(dom, DisplayObject(Application.application), true);
            PopUpManager.centerPopUp(dom);
        }

        public function downloadISOntologyIntancesCompleted(event: Event): void {
            PopUpManager.removePopUp(IFlexDisplayObject(event.target));

            _mapperModel.populateXmlTreeProvider(OntologyManager.dbCollectionSchema);
            loadMergesInstances();
        }

        private function loadMergesInstances(): void{
            var dom : LoadMergesInstances = new LoadMergesInstances();
            dom.addEventListener(Event.COMPLETE, downloadMergeIntancesCompleted);
            PopUpManager.addPopUp(dom, DisplayObject(Application.application), true);
            PopUpManager.centerPopUp(dom);
        }

        public function downloadMergeIntancesCompleted(event: Event): void {
            PopUpManager.removePopUp(IFlexDisplayObject(event.target));

            var downloadLocales: LoadLocalesInstances = new LoadLocalesInstances();
            downloadLocales.addEventListener(Event.COMPLETE, downloadLocaleIntancesCompleted);
            PopUpManager.addPopUp(downloadLocales, DisplayObject(Application.application), true);
            PopUpManager.centerPopUp(downloadLocales);
        }

        public function downloadLocaleIntancesCompleted(event: Event): void {
            PopUpManager.removePopUp(IFlexDisplayObject(event.target));
        }

        private function getOntologyInstancesEventGenerateBIHandler(event: GetOntologyInstancesEvent): void{
            var panelHeuristicsConfig : PanelHeuristicsConfigSimple = new PanelHeuristicsConfigSimple();
            panelHeuristicsConfig.addEventListener(PanelHeuristicsConfigExecute.CONFIRM, panelHeuristicsConfigExecuteHandler);
            PopUpManager.addPopUp(panelHeuristicsConfig, DisplayObject(Application.application), true);
            PopUpManager.centerPopUp(panelHeuristicsConfig);
        }

    ]]></mx:Script>


```

```

private function panelHeuristicsConfigExecuteHandler(event: PanelHeuristicsConfigExecute): void{
    PopUpManager.removePopUp(IFlexDisplayObject(event.target));
    if(event.type == PanelHeuristicsConfigExecute.CONFIRM){
        var dom : GenerateBIOntologyInstances = new GenerateBIOntologyInstances(event.tableFactPrefix,
event.measurePrefix);

        dom.addEventListener(Event.COMPLETE, downloadBIOntologyInstancesCompleted);
        PopUpManager.addPopUp(dom, DisplayObject(Application.application), true);
        PopUpManager.centerPopUp(dom);
    }
}

public function downloadBIOntologyInstancesCompleted(event: Event): void {
    PopUpManager.removePopUp(IFlexDisplayObject(event.target));
    var downloadAggregateFunctions: LoadAggregateFunctionInstances = new LoadAggregateFunctionInstances();
    downloadAggregateFunctions.addEventListener(Event.COMPLETE, downloadAggregateFunctionsCompleted);
    PopUpManager.addPopUp(downloadAggregateFunctions, DisplayObject(Application.application), true);
    PopUpManager.centerPopUp(downloadAggregateFunctions);
}

public function downloadAggregateFunctionsCompleted(event: Event): void {
    PopUpManager.removePopUp(IFlexDisplayObject(event.target));
    loadThemes();
}

private function loadThemes(): void{
    var themes: Array;
    var theme: Theme;
    var themeObj: Object;

    themes = OntologyManager.themes;

    var i: uint;
    for(i=0; i< themes.length; i++){
        theme = Theme(themes[i]);
        themeObj = new Object;
        themeObj.uriTheme = theme.URI;
        themeObj.labelTheme =
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(theme.labels.values[0]).label
        _mapperModel.cboThemesProvider.addItem(themeObj);
    }

    if(themes.length > 0){
        _mapperModel.selectedTheme = Theme(themes[0]);
        canvasMapperBI.loadThemeUnits(_mapperModel.selectedTheme);
    }
}

private function getOntologyInstancesEventLoadAllHandler(event: GetOntologyInstancesEvent): void{
    var panelLoadOntology : LoadISOInstances = new LoadISOInstances();
    panelLoadOntology.addEventListener(Event.COMPLETE, downloadAllOntologyInstancesCompleted);
    PopUpManager.addPopUp(panelLoadOntology, DisplayObject(Application.application), true);
    PopUpManager.centerPopUp(panelLoadOntology);
}

public function downloadAllOntologyInstancesCompleted(event: Event): void {
    PopUpManager.removePopUp(mx.core.FlexDisplayObject(event.target));

    _mapperModel.populateXmlTreeProvider(OntologyManager.dbCollectionSchema);

    loadMergesInstances();

    var dom : LoadBIOInstances = new LoadBIOInstances();
    dom.addEventListener(Event.COMPLETE, downloadBIOntologyInstancesCompleted);
    PopUpManager.addPopUp(dom, DisplayObject(Application.application), true);
    PopUpManager.centerPopUp(dom);
}
]]</mx:Script>

<mx:Canvas width="100%" height="100%">
    <mx:TabNavigator id="tabNavigator" bottom="30" top="10" left="10" right="10" creationPolicy="all">
        <mapper:MapperISO id="canvasMapperISO" width="100%" height="100%"/>
        <mapper:MapperBI id="canvasMapperBI" width="100%" height="100%"/>
        <!-- <mapper:MapperDomain id="canvasMapperDomain" width="100%" height="100%"/> -->
    </mx:TabNavigator>
    <mx:Canvas width="100%" height="30" bottom="0" x="0"/>
</mx:Canvas>
</mx:Application>

```

MapperService.as

```

package br.ufsc.inf.feco.mapper.business{

import br.ufsc.inf.feco.mapper.events.ResultDapacketEvent;
import br.ufsc.inf.feco.ontologymanager.OntologyManager;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.Function;

import flash.events.Event;
import flash.utils.ByteArray;

```



```

import mx.rpc.events.FaultEvent;
import mx.rpc.events.ResultEvent;
import mx.rpc.remoting.mxml.RemoteObject;

import org.granite.collections.BasicMap;

/**
 * Classe que controla a invocação e o retorno dos métodos remotos (no lado java)
 */
public class MapperService{

    /** Endereço onde está disponível o servlet do granite */
    public static const _url : String = "http://127.0.0.1:8080/mapper/graniteamf";

    /** Objeto para invocação remota de métodos */
    private var _remote: RemoteObject;

    /** Função que será executada após a execução do método remoto */
    private var _functionExecute: Function;

    /** Evento do "parent" que invocou o serviço remoto */
    private var _eventParent: Event;

    /**
     * Construtor
     * (nele é passado a função que se quer executar no retorno da chamada de um método remoto)
     */
    public function MapperService(functionExecute: Function, eventParent: Event) {
        _functionExecute = functionExecute;
        _eventParent = eventParent;
        _remote = new RemoteObject();
        _remote.endpoint = _url;
        _remote.destination = "mapper"; //alias no arquivo de configuração dos serviços disponíveis do granite
        _remote.showBusyCursor = true;
        _remote.addEventListener(ResultEvent.RESULT, onResult);
        _remote.addEventListener(FaultEvent.FAULT, onFault);
    }

    public function generateDBCollectionSchema(): void{
        _remote.generateDBCollectionSchema();
    }

    public function getDBCollectionSchema(): void{
        _remote.getDBCollectionSchema();
    }

    public function generateThemes(tableFactPrefix: String, measurePrefix: String): void {
        _remote.generateThemes(tableFactPrefix, measurePrefix);
    }

    public function getThemes(): void{
        _remote.getThemes();
    }

    public function getMerges(): void{
        _remote.getMerges();
    }

    public function getLocales(): void{
        _remote.getLocales();
    }

    public function getAggregateFunctions(): void{
        _remote.getAggregateFunctions();
    }

    public function saveThemes(): void{
        _remote.saveThemes(OntologyManager.themes, OntologyManager.deletedURIs);
    }

    /**
     * Evento que é disparado se houver uma falha no método remoto
     */
    private function onFault(event: FaultEvent): void {
        trace(event);
        onResult(new ResultEvent(ResultEvent.RESULT));
    }

    /**
     * Evento disparado no retorno com sucesso da invocação de
     * um método remoto
     */
    private function onResult(event: ResultEvent) : void {
        trace(event);
        if((event.result != null) && (_functionExecute != null)){
            //executa a função determinada
            _functionExecute(event);
        }
    }
}

```

```

    }
}

/**
 * Retorna a instância do RemoteObject, que faz a invocação
 * dos métodos remotos
 *
 */
public function getRemoteObject(): RemoteObject {
    return _remote;
}
}
}
}

```

ElementTypes.as

```

package br.ufsc.inf.feco.mapper.constants{

/**
 * Classe que representa os tipos de elementos utilizados no sistema.
 *
 */
public class ElementTypes{

    private var _type: int = 0;

    public function ElementTypes(type: int){
        super();
        _type = type;
    }

    public static var THEME : ElementTypes = new ElementTypes(0);
    public static var THEME_UNIT : ElementTypes = new ElementTypes(1);
    public static var MEASURE : ElementTypes = new ElementTypes(2);
    public static var GROUPING : ElementTypes = new ElementTypes(3);
    public static var GROUPING_UNIT : ElementTypes = new ElementTypes(4);
    public static var DETAIL : ElementTypes = new ElementTypes(5);
    public static var DB_COLLECTION : ElementTypes = new ElementTypes(6);
    public static var DB_ATTRIBUTE : ElementTypes = new ElementTypes(7);
    public static var MERGE : ElementTypes = new ElementTypes(8);
    public static var MERGE_DETAIL : ElementTypes = new ElementTypes(9);

    public function equals(type: ElementTypes): Boolean {
        if (this == type) return true;
        if (type == null) return false;
        return (_type == type._type);
    }

    public function getType(): int {
        return _type;
    }

    public function isAnalytical(): Boolean{
        if(_type == THEME._type) || (_type == THEME_UNIT._type) || (_type == MEASURE._type) ||
            (_type == GROUPING._type) || (_type == GROUPING_UNIT._type) || (_type == DETAIL._type)
            return true;
        return false;
    }

    public function isInformationSource(): Boolean{
        if(_type == DB_COLLECTION._type) || (_type == DB_ATTRIBUTE._type)
            return true;
        return false;
    }

    public function isTheme(): Boolean {
        return (_type == THEME._type)
    }

    public function isThemeUnit(): Boolean {
        return (_type == THEME_UNIT._type)
    }

    public function isMeasure(): Boolean {
        return (_type == MEASURE._type)
    }

    public function isGrouping(): Boolean {
        return (_type == GROUPING._type);
    }

    public function isGroupingUnit(): Boolean {
        return (_type == GROUPING_UNIT._type);
    }

    public function isDetail(): Boolean {
        return (_type == DETAIL._type);
    }
}

```

```

public function isDBCollection(): Boolean {
    return (_type == DB_COLLECTION._type);
}

public function isDBAttribute(): Boolean {
    return (_type == DB_ATTRIBUTE._type);
}

public function isMerge(): Boolean {
    return (_type == MERGE._type);
}

public function isMergeDetail(): Boolean {
    return (_type == MERGE_DETAIL._type);
}

public function getName(): String {
    if (_type == THEME._type)
        return 'theme';
    else if (_type == THEME_UNIT._type)
        return 'themeUnit';
    else if (_type == MEASURE._type)
        return 'measure';
    else if (_type == GROUPING._type)
        return 'grouping';
    else if (_type == GROUPING_UNIT._type)
        return 'groupingUnit';
    else if (_type == DETAIL._type)
        return 'detail';
    else if (_type == DB_COLLECTION._type)
        return 'dbCollection';
    else if (_type == DB_ATTRIBUTE._type)
        return 'dbAttribute';
    else if (_type == MERGE._type)
        return 'merge';
    else if (_type == MERGE_DETAIL._type)
        return 'mergeDetail';
    return null;
}

public static function getAnalyticalElementTypeByName(name: String): ElementTypes{
    if (name == "theme")
        return THEME;
    else if (name == "themeUnit")
        return THEME_UNIT;
    else if (name == "measure")
        return MEASURE;
    else if (name == "grouping")
        return GROUPING;
    else if (name == "groupingUnit")
        return GROUPING_UNIT;
    else if (name == "detail")
        return DETAIL;
    else if (name == "dbCollection")
        return DB_COLLECTION;
    else if (name == "dbAttribute")
        return DB_ATTRIBUTE;
    else if (name == "merge")
        return MERGE;
    else if (name == "mergeDetail")
        return MERGE_DETAIL;
    return null;
}
}
}
}

```

ConfigThemeEvent.as

```

package br.ufsc.inf.feco.mapper.events{

import flash.events.Event;

public class ConfigThemeEvent extends Event{

    public static var CONFIRM: String = "confirm";
    public static var CANCEL: String = "cancel";

    private var _themeName: String;

    public function ConfigThemeEvent(type:String, themeName: String=null, bubbles:Boolean=true, cancelable:Boolean=false){
        super(type, bubbles, cancelable);
        _themeName = themeName;
    }

    override public function clone(): Event {
        return new ConfigThemeEvent(type, themeName, bubbles, cancelable);
    }

    public function get themeName(): String{
        return _themeName;
    }
}

```

```
}  
}  
}
```

GetOntologyInstancesEvent.as

```
package br.ufsc.inf.feco.mapper.events{  
  
    import flash.events.Event;  
  
    public class GetOntologyInstancesEvent extends Event{  
  
        public static var GENERATE_ISO: String = "generateISO";  
        public static var GENERATE_BI: String = "generateBI";  
        public static var LOAD_ALL: String = "loadAll";  
  
        public function GetOntologyInstancesEvent(type:String, bubbles:Boolean=false, cancelable:Boolean=false){  
            super(type, bubbles, cancelable);  
        }  
  
        override public function clone(): Event {  
            return new GetOntologyInstancesEvent(type, bubbles, cancelable);  
        }  
    }  
}
```

HBoxAnalyticalElementClick.as

```
package br.ufsc.inf.feco.mapper.events{  
  
    import br.ufsc.inf.feco.mapper.constants.ElementTypes;  
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;  
  
    import flash.events.Event;  
  
    public class HBoxAnalyticalElementClick extends Event{  
  
        public static var LINKBUTTONCLICK: String = "linkbuttonclick";  
  
        private var _elementType: ElementTypes;  
        private var _URI: URI_AS;  
        private var _parentURI: URI_AS;  
  
        public function HBoxAnalyticalElementClick(type:String, elementType:ElementTypes, URI: URI_AS, parentURI: URI_AS, bubbles:Boolean=true,  
cancelable:Boolean=false){  
            super(type, bubbles, cancelable);  
            _elementType = elementType;  
            _URI = URI;  
            _parentURI = parentURI;  
        }  
  
        override public function clone(): Event {  
            return new HBoxAnalyticalElementClick(type, elementType, URI, parentURI, bubbles, cancelable);  
        }  
  
        public function get elementType(): ElementTypes{  
            return _elementType;  
        }  
  
        public function get URI(): URI_AS{  
            return _URI;  
        }  
  
        public function get parentURI(): URI_AS{  
            return _parentURI;  
        }  
    }  
}
```

HBoxAnalyticalElementRemove.as

```
package br.ufsc.inf.feco.mapper.events{  
  
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;  
    import br.ufsc.inf.feco.mapper.constants.ElementTypes;  
  
    import flash.events.Event;  
  
    public class HBoxAnalyticalElementRemove extends Event{  
  
        public static var REMOVE: String = "remove";  
  
        private var _elementType: ElementTypes;  
        private var _URI: URI_AS;
```

```

        private var _parentURI: URI_AS;

        public function HBoxAnalyticalElementRemove(type:String, elementType:ElementTypes, URI: URI_AS, parentURI: URI_AS, bubbles:Boolean=true,
cancelable:Boolean=false){
            super(type, bubbles, cancelable);
            _elementType = elementType;
            _URI = URI;
            _parentURI = parentURI;
        }

        override public function clone(): Event {
            return new HBoxAnalyticalElementRemove(type, elementType, URI, parentURI, bubbles, cancelable);
        }

        public function get elementType(): ElementTypes{
            return _elementType;
        }

        public function get URI(): URI_AS{
            return _URI;
        }

        public function get parentURI(): URI_AS{
            return _parentURI;
        }
    }
}

```

HBoxAnalyticalElementUpdateLabel.as

```

package br.ufsc.inf.feco.mapper.events{

    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;
    import br.ufsc.inf.feco.mapper.constants.ElementTypes;

    import flash.events.Event;

    public class HBoxAnalyticalElementUpdateLabel extends Event{

        public static var UPDATE_LABEL: String = "updateLabel";

        private var _elementType: ElementTypes;
        private var _URI: URI_AS;
        private var _parentURI: URI_AS;
        private var _label: String;

        public function HBoxAnalyticalElementUpdateLabel(type:String, elementType:ElementTypes, URI: URI_AS, parentURI: URI_AS, label:String,
bubbles:Boolean=true, cancelable:Boolean=false){
            super(type, bubbles, cancelable);
            _elementType = elementType;
            _URI = URI;
            _parentURI = parentURI;
            _label = label;
        }

        override public function clone(): Event {
            return new HBoxAnalyticalElementUpdateLabel(type, elementType, URI, parentURI, label, bubbles, cancelable);
        }

        public function get elementType(): ElementTypes{
            return _elementType;
        }

        public function get URI(): URI_AS{
            return _URI;
        }

        public function get parentURI(): URI_AS{
            return _parentURI;
        }

        public function get label(): String{
            return _label;
        }
    }
}

```

HBoxMergeDetailModification.as

```

package br.ufsc.inf.feco.mapper.events{

    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;

    import flash.events.Event;

    public class HBoxMergeDetailModification extends Event{

```

```

        public static var UPDATE_SECOND_ATTRIBUTE: String = "updateSecondAttribute";
        public static var REMOVE: String = "remove";

        private var _secAttributeURIName: String;
        private var _secTableURIName: String;
        private var _mergeURI: URI_AS;
        private var _binaryExpressionURI: URI_AS;

        public function HBoxMergeDetailModification(type:String, secAttributeURIName: String, secTableURIName: String, mergeURI: URI_AS,
binaryExpressionURI: URI_AS, bubbles:Boolean=false, cancelable:Boolean=false){
            super(type, bubbles, cancelable);
            _secAttributeURIName = secAttributeURIName;
            _secTableURIName = secTableURIName;
            _mergeURI = mergeURI;
            _binaryExpressionURI = binaryExpressionURI;
        }

        override public function clone(): Event {
            return new HBoxMergeDetailModification(type, secAttributeURIName, secTableURIName, mergeURI, binaryExpressionURI, bubbles, cancelable);
        }

        public function get secAttributeURIName(): String{
            return _secAttributeURIName;
        }

        public function get secTableURIName(): String{
            return _secTableURIName;
        }

        public function get mergeURI(): URI_AS{
            return _mergeURI;
        }

        public function get binaryExpressionURI(): URI_AS{
            return _binaryExpressionURI;
        }
    }
}

```

PanelHeuristicsConfigExecute.as

```

package br.ufsc.inf.feco.mapper.events{

    import flash.events.Event;

    public class PanelHeuristicsConfigExecute extends Event{

        public static var CONFIRM: String = "confirm";
        public static var CANCEL: String = "cancel";

        private var _automatic: Boolean;
        private var _tableFactPrefix: String;
        private var _measurePrefix: String;

        public function PanelHeuristicsConfigExecute(type:String, automatic: Boolean, tableFactPrefix: String, measurePrefix: String, bubbles:Boolean=true,
cancelable:Boolean=false){
            super(type, bubbles, cancelable);
            _automatic = automatic;
            _tableFactPrefix = tableFactPrefix;
            _measurePrefix = measurePrefix;
        }

        override public function clone(): Event {
            return new PanelHeuristicsConfigExecute(type, automatic, tableFactPrefix, measurePrefix, bubbles, cancelable);
        }

        public function get automatic(): Boolean{
            return _automatic;
        }

        public function get tableFactPrefix(): String{
            return _tableFactPrefix;
        }

        public function get measurePrefix(): String{
            return _measurePrefix;
        }
    }
}

```

MapperModelLocator.as

```

package br.ufsc.inf.feco.mapper.model{

    import com.adobe.cairngorm.CairngormError;
    import com.adobe.cairngorm.CairngormMessageCodes;
    import com.adobe.cairngorm.model.IModelLocator;
}

```

```

import flash.events.EventDispatcher;

/**
 * Classe responsável por implementar a característica singleton para o model da aplicação
 * (garantindo que o modelo utilizado será sempre o mesmo e não será possível instanciar
 * dois modelos)
 */
public class MapperModelLocator extends EventDispatcher implements IModelLocator{

    private static var instance : MapperModelLocator;

    private var _mapperModel: MapperModel = new MapperModel();

    public function MapperModelLocator(){
        super();
        if (instance != null)
            throw new CairngormError(CairngormMessageCodes.SINGLETON_EXCEPTION, "MapperModelLocator");
        instance = this;
    }

    public static function getInstance(): MapperModelLocator{
        if (MapperModelLocator.instance == null)
            MapperModelLocator.instance = new MapperModelLocator();

        return instance;
    }

    public function set mapperModel(mapperModel: MapperModel): void {
        if (_mapperModel != mapperModel) {
            _mapperModel = mapperModel;
            //dispatchEvent(new MapperModelChange());
        }
    }

    public function get mapperModel(): MapperModel {
        return _mapperModel;
    }

    public function notifyListen(event: String, listener: Function):void {
        this.addEventListener(event, listener);
    }
}

```

MapperModel.as

```

package br.ufsc.inf.feco.mapper.model{
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.BIThemeUnit;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Grouping;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.GroupingUnit;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.MeasureUnit;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Theme;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBAttribute;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBCollection;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.Merge;

    import mx.collections.ArrayCollection;
    import mx.collections.XMLListCollection;

    /**
     * Classe que representa o model da aplicação
     * Deve ser instanciada usando a classe MapperModelLocator
     * (que garante a propriedade "singleton" - será usado sempre o mesmo modelo)
     */
    public class MapperModel{

        //nome do esquema utilizado
        private var _strSchemaName: String;

        //data provider da árvore que mostra o esquema de banco de dados na interface
        [Bindable]
        private var _dataSourcesTreeData: XMLListCollection;

        //xml usado no data provider da árvore
        [Bindable]
        private var _xmlTreeProvider : XML;

        //data provider do combobox que lista todos os temas da ontologia de bi
        [Bindable]
        private var _cboThemesProvider: ArrayCollection = new ArrayCollection();

        //representa uma tabela selecionada (ontologia de fontes-de-informação).
        private var _selectedDBCollection: DBCollection;
        //representa um atributo selecionado (ontologia de fontes-de-informação).
        private var _selectedDBAttribute: DBAttribute;
        //representa uma junção selecionada (ontologia de fontes-de-informação).
    }
}

```

```

private var _selectedMerge: Merge;
//representa um tema selecionado (ontologia de BI).
private var _selectedTheme: Theme;
//representa uma unidade de tema selecionada (ontologia de BI).
private var _selectedThemeUnit: BIThemeUnit;
//representa uma medida selecionada (ontologia de BI).
private var _selectedMeasure: MeasureUnit;
//representa um agrupamento selecionado (ontologia de BI).
private var _selectedGrouping: Grouping;
//representa uma unidade de agrupamento selecionada (ontologia de BI).
private var _selectedGroupingUnit: GroupingUnit;

/**
 * Métodos de acesso às propriedades (GETs e SETs)
 */
public function get strSchemaName(): String{
    return _strSchemaName;
}

public function set strSchemaName(strSchemaName: String): void{
    _strSchemaName = strSchemaName;
}

public function get dataSourcesTreeData(): XMLListCollection{
    return _dataSourcesTreeData;
}

[Bindable]
public function set dataSourcesTreeData(dataSourcesTreeData: XMLListCollection): void{
    _dataSourcesTreeData = dataSourcesTreeData;
}

public function get xmlTreeProvider(): XML{
    return _xmlTreeProvider;
}

public function set xmlTreeProvider(xmlTreeProvider: XML): void{
    _xmlTreeProvider = xmlTreeProvider;
}

public function get cboThemesProvider(): ArrayCollection{
    return _cboThemesProvider;
}

[Bindable]
public function set cboThemesProvider(cboThemesProvider: ArrayCollection): void{
    _cboThemesProvider = cboThemesProvider;
}

public function get selectedDBCollection(): DBCollection{
    return _selectedDBCollection;
}

public function set selectedDBCollection(selectedDBCollection: DBCollection): void{
    _selectedDBCollection = selectedDBCollection;
}

public function get selectedDBAttribute(): DBAttribute{
    return _selectedDBAttribute;
}

public function set selectedDBAttribute(selectedDBAttribute: DBAttribute): void{
    _selectedDBAttribute = selectedDBAttribute;
}

public function get selectedTheme(): Theme{
    return _selectedTheme;
}

public function set selectedTheme(selectedTheme: Theme): void{
    _selectedTheme = selectedTheme;
}

public function get selectedThemeUnit(): BIThemeUnit{
    return _selectedThemeUnit;
}

public function set selectedThemeUnit(selectedThemeUnit: BIThemeUnit): void{
    _selectedThemeUnit = selectedThemeUnit;
}

public function get selectedMeasure(): MeasureUnit{
    return _selectedMeasure;
}

public function set selectedMeasure(selectedMeasure: MeasureUnit): void{
    _selectedMeasure = selectedMeasure;
}

public function get selectedGrouping(): Grouping{
    return _selectedGrouping;
}

```



```

    }

    public function set selectedGrouping(selectedGrouping: Grouping): void{
        _selectedGrouping = selectedGrouping;
    }

    public function get selectedGroupingUnit(): GroupingUnit{
        return _selectedGroupingUnit;
    }

    public function set selectedGroupingUnit(selectedGroupingUnit: GroupingUnit): void{
        _selectedGroupingUnit = selectedGroupingUnit;
    }

    public function get selectedMerge(): Merge{
        return _selectedMerge;
    }

    public function set selectedMerge(selectedMerge: Merge): void{
        _selectedMerge = selectedMerge;
    }

    /**
     * Método que popula o xml que será usado no data provider da árvore que mostra o esquema
     * de banco de dados na interface.
     */
    public function populateXmlTreeProvider(dbCollectionSchema: DBCollection): void{
        strSchemaName = dbCollectionSchema.URI.name;

        xmlTreeProvider = <schema/>;

        informationSourcesTreeData = dbCollectionSchema.informationSources.values;

        for each(var dbCollectionTable2: DBCollection in informationSourcesTreeData){
            var xmlNode: XML =
                <table source={dbCollectionTable2.source} uriNameSpace={dbCollectionTable2.URI.ns.ns}
uriName={dbCollectionTable2.URI.name}/>;

            var isTableAttributes: ArrayCollection = dbCollectionTable2.informationSources.values;
            for each (var isAttribute: DBAttribute in isTableAttributes){
                var xmlAttNode: XML = <attribute source={isAttribute.source} uriNameSpace={isAttribute.URI.ns.ns}
uriName={isAttribute.URI.name} />;

                xmlNode.appendChild(xmlAttNode);
            }
            xmlTreeProvider.appendChild(xmlNode);
        }
        //trace(_mapperModel.xmlTreeProvider.toString());

        dataSourcesTreeData = new XMLListCollection(xmlTreeProvider.table);
    }
}

```

MapperISO.mxml

```

<?xml version="1.0" encoding="utf-8"?>

<!--Classe que representa o conteúdo da aba de mapeamento e configuração da ontologia de fontes-de-informação -->
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" xmlns:view="br.ufsc.inf.feco.mapper.view.*"
label="Ontologia ISO" width="100%" height="100%" initialize="init()">

    <mx:Script><![CDATA[
import br.ufsc.inf.feco.mapper.events.GetOntologyInstancesEvent;
import br.ufsc.inf.feco.mapper.view.popups.SaveBIOntology;
import br.ufsc.inf.feco.mapper.view.popups.LoadLocalesInstances;
import br.ufsc.inf.feco.mapper.events.HBoxMergeDetailModification;
import mx.collections.XMLListCollection;
import mx.core.UIComponent;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.Merge$Type;
import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementUpdateLabel;
import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementClick;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.Element;
import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementRemove;
import br.ufsc.inf.feco.mapper.model.MapperModel;
import br.ufsc.inf.feco.mapper.constants.ElementTypes;
import mx.core.Application;
import br.ufsc.inf.feco.mapper.model.MapperModelLocator;
import mx.core.IFlexDisplayObject;
import mx.managers.PopUpManager;
import br.ufsc.inf.feco.mapper.view.popups.GenerateISOInstances;
import org.granite.collections.BasicMap;
import mx.controls.Alert;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.BinaryExpression$Type;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.UnaryExpression;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.Merge;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBCollection;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBAttribute;
import mx.collections.ArrayCollection;
import mx.controls.ComboBox;

```

```

import br.ufsc.inf.feco.ontologymanager.OntologyManager;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.BinaryExpression;
import mx.managers.DragManager;
import mx.core.DragSource;
import mx.events.DragEvent;

private var _mapperModel: MapperModel = MapperModelLocator.getInstance().mapperModel;

[Bindable]
private var _cboMergeTypes: ArrayCollection = new ArrayCollection(Merge$Type.constants);

private function init(): void{
    addEventListener(HBoxAnalyticalElementClick.LINKBUTTONCLICK, hBoxAnalyticalElementClickHandler);
    addEventListener(HBoxAnalyticalElementRemove.REMOVE, hBoxAnalyticalElementRemoveHandler);
    addEventListener(HBoxMergeDetailModification.UPDATE_SECOND_ATTRIBUTE, hBoxMergeDetailUpdateSecondAttribute);
    addEventListener(HBoxMergeDetailModification.REMOVE, hBoxMergeDetailRemove);
}

private function btnCreateInstancesClickHandler() : void {
    dispatchEvent(new GetOntologyInstancesEvent(GetOntologyInstancesEvent.GENERATE_ISO, true));
}

private function hBoxMergeDetailUpdateSecondAttribute(event: HBoxMergeDetailModification): void{
    var attribute: DBAttribute = OntologyManager.getDBAttributeByURIName(event.secTableURIName, event.secAttributeURIName);
    HBoxMergeDetail(event.target).dbAttribute2URI = attribute.URI;
    UnaryExpression(OntologyManager.getBinaryExpression(event.binaryExpressionURI,
event.mergeURI).secondExpression).informationSource = attribute;
}

private function hBoxMergeDetailRemove(event: HBoxMergeDetailModification): void{
    contMergeDetails.removeChild(HBoxMergeDetail(event.target.parent));
}

private function treeISO1ClickHancler(event: MouseEvent): void{
    if(XML(treeISO1.selectedItem).localName() == "table"){
        var dbCollectionClicked: DBCollection = OntologyManager.getDBCollectionByURIName(XML(treeISO1.selectedItem).@uriName);
        if(_mapperModel.selectedDBCollection != dbCollectionClicked){
            _mapperModel.selectedDBCollection = dbCollectionClicked;
            lblSelectedDBCollection.text = _mapperModel.selectedDBCollection.source;
            loadDBCollectionMerges();
        }
    }
    }else if(XML(treeISO1.selectedItem).localName() == "attribute"){
        var dbAttributeClicked: DBAttribute = OntologyManager.getDBAttributeByURIName(XML(treeISO1.selectedItem).parent().@uriName,
XML(treeISO1.selectedItem).@uriName);
        if(_mapperModel.selectedDBAttribute != dbAttributeClicked){
            _mapperModel.selectedDBAttribute = dbAttributeClicked;
            txtDBAttributeName.text = _mapperModel.selectedDBAttribute.source;
            if(_mapperModel.selectedDBAttribute.attributeType)
                txtDBAttributeType.text = _mapperModel.selectedDBAttribute.attributeType.toString(); //erro
            else
                txtDBAttributeType.text = "";
            if(_mapperModel.selectedDBAttribute.attributeSize)
                txtDBAttributeSize.text = String(_mapperModel.selectedDBAttribute.attributeSize);
            else
                txtDBAttributeSize.text = "";
        }
    }
}

private function loadDBCollectionMerges(): void{
    var merge: Merge;
    var sourceCollection: DBCollection;
    var targetCollection: DBCollection;
    var hBoxMerge: HBox;
    var cboMergeType: ComboBox;
    var hBoxAnalyticalElement: HBoxAnalyticalElement;
    var i: uint;
    contMergedTables.removeAllChildren();
    for(i=0;i< OntologyManager.merges.length; i++){
        merge = Merge(OntologyManager.merges[i]);
        sourceCollection = DBCollection(merge.sourceCollection);
        targetCollection = DBCollection(merge.targetCollection);
        if(sourceCollection.URI.name == _mapperModel.selectedDBCollection.URI.name){
            hBoxMerge = new HBox;
            cboMergeType = new ComboBox;
            cboMergeType.dataProvider = _cboMergeTypes;
            cboMergeType.selectedItem = merge.mergeType;
            cboMergeType.addEventListener(Event.CHANGE, changeMergeType);
            hBoxMerge.addChild(cboMergeType);
            hBoxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.MERGE, null, merge.URI, targetCollection.source);
            hBoxMerge.addChild(hBoxAnalyticalElement);
            contMergedTables.addChild(hBoxMerge);
        }else if(targetCollection.URI.name == _mapperModel.selectedDBCollection.URI.name){
            hBoxMerge = new HBox;
            cboMergeType = new ComboBox;
            cboMergeType.dataProvider = _cboMergeTypes;
            setComboMergeType(cboMergeType, merge.mergeType);
            cboMergeType.addEventListener(Event.CHANGE, changeMergeType);
            hBoxMerge.addChild(cboMergeType);
            hBoxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.MERGE, null, merge.URI, sourceCollection.source);
        }
    }
}

```

```

        hboxMerge.addChild(hboxAnalyticalElement);
        contMergedTables.addChild(hboxMerge);
    }
    if(contMergedTables.getChildren().length > 0){
        _mapperModel.selectedMerge = OntologyManager.getMerge(HBoxAnalyticalElement(HBox(contMergedTables.getChildAt(0)).getChildAt(1)).URI);
        selectHBoxAnalyticalElement(ElementTypes.MERGE, _mapperModel.selectedMerge);
    }else{
        contMergeDetails.removeAllChildren();
    }
}

private function setComboMergeType(cboMergeType: ComboBox, mergeType: Merge$Type): void{
    var i: uint;
    for(i = 0; i < _cboMergeTypes.length; i++){
        if(Merge$Type(_cboMergeTypes.getItemAt(i)).name == mergeType.name){
            cboMergeType.selectedItem = Merge$Type(_cboMergeTypes.getItemAt(i));
            break;
        }
    }
}

private function hboxAnalyticalElementClickHandler(event: HBoxAnalyticalElementClick): void{
    if(event.elementType == ElementTypes.MERGE){
        if(_mapperModel.selectedMerge != null){
            var m: uint;
            for (m=0; m<contMergedTables.getChildren().length; m++){
                if(HBoxAnalyticalElement(HBox(contMergedTables.getChildren()[m]).getChildAt(1)).URI ==
                _mapperModel.selectedMerge.URI){
                    HBoxAnalyticalElement(HBox(contMergedTables.getChildren()[m]).getChildAt(1)).selectLinkButton(false);
                }
            }
            _mapperModel.selectedMerge = Merge(OntologyManager.getMerge(event.URI));
            loadMergeDetails();
        }
    }
}

private function selectHBoxAnalyticalElement(type: ElementTypes, element: Element): void{
    var i: uint;
    if(type == ElementTypes.MERGE){
        for (i=0; i< contMergedTables.getChildren().length; i++){
            if(HBoxAnalyticalElement(HBox(contMergedTables.getChildren()[i]).getChildAt(1)).URI == element.URI){
                HBoxAnalyticalElement(HBox(contMergedTables.getChildren()[i]).getChildAt(1)).selectLinkButton(true);
            }
        }
        _mapperModel.selectedMerge = Merge(element);
        loadMergeDetails();
    }
}

private function hboxAnalyticalElementRemoveHandler(event: HBoxAnalyticalElementRemove): void{
    trace(event.URI);
    var i: uint;
    if(event.elementType == ElementTypes.MERGE){
        OntologyManager.removeMerge(event.URI);
        for (i=0; i< contMergedTables.getChildren().length; i++){
            if(HBoxAnalyticalElement(HBox(contMergedTables.getChildren()[i]).getChildAt(1)).URI == event.URI){
                contMergedTables.removeChildAt(i);
            }
        }
    }else if(event.elementType == ElementTypes.MERGE_DETAIL){
        for (i=0; i< contMergeDetails.getChildren().length; i++){
            if(HBoxMergeDetail(contMergeDetails.getChildren()[i]).dbAttribute1URI == event.URI){
                contMergeDetails.removeChildAt(i);
            }
        }
    }
}

private function contMergedTablesDragEnterHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        var ds: DragSource = event.dragSource;
        if(!ds.hasFormat("treeItems")) return;
        var items: Array = ds.dataForFormat("treeItems") as Array;
        //percorre todos os treeitens selecionados para drag
        for(var i:Number=0; i < items.length; i++){
            var item:XML = XML(items[i]);
            if((item.localName() != "table"))
                return;
        }
        DragManager.acceptDragDrop(UIComponent(event.currentTarget));
    }
}

private function contMergedTablesDragOverHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        DragManager.showFeedback(DragManager.COPY);
    }
}

private function contMergedTablesDragDropHandler(event: DragEvent): void{
    var ds: DragSource = event.dragSource;
}

```

```

var dropTarget: VBox = VBox(event.currentTarget);
var arr: Array;

if( ds.hasFormat("items") ) {
    arr = ds.dataForFormat("items") as Array;
}else if( ds.hasFormat("treeltems") ) {
    arr = ds.dataForFormat("treeltems") as Array;
}

if(_mapperModel.selectedDBCollection){
    for(var i:Number=0; i < arr.length; i++) {
        var node:XML = XML(arr[i]);
        var hboxMerge: HBox = new HBox;
        var hboxAnalyticalElement: HBoxAnalyticalElement;
        var cboMergeType: ComboBox;
        var tabela: DBCollection;

        if(node.localName() == "table"){
            tabela = OntologyManager.getDBCollectionByURIName(node.@uriName);
            cboMergeType = new ComboBox;
            cboMergeType.dataProvider = _cboMergeTypes;
            cboMergeType.selectedIndex = 0;
            hboxMerge.addChild(cboMergeType);
            var merge: Merge = OntologyManager.createMerge();
            merge.sourceCollection = _mapperModel.selectedDBCollection;
            merge.targetCollection = tabela;
            merge.mergeType = MergeSType.INNER_JOIN;
            merge.bidirectional = true;
            _mapperModel.selectedMerge = merge;
            OntologyManager.merges.push(merge);
            hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.MERGE, null, merge.URI,

tabela.source);

            hboxMerge.addChild(hboxAnalyticalElement);
            contMergedTables.addChild(hboxMerge);
        }
    }
}else{
    Alert.show("Selecione uma tabela antes.", "Atenção!");
}
}

private function contMergeDetailsDragEnterHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        var ds: DragSource = event.dragSource;
        if(!ds.hasFormat("treeltems")) return;
        var items: Array = ds.dataForFormat("treeltems") as Array;
        //percorre todos os treeitens selecionados para drag
        for(var i:Number=0; i < items.length; i++){
            var item:XML = XML(items[i]);
            if((item.localName() != "attribute"))
                return;
        }
        DragManager.acceptDragDrop(UIComponent(event.currentTarget));
    }
}

private function contMergeDetailsDragOverHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        DragManager.showFeedback(DragManager.COPY);
    }
}

private function loadMergeDetails(): void{
    var hboxMergeDetail: HBoxMergeDetail;
    var binaryExpression: BinaryExpression = _mapperModel.selectedMerge.relation;
    var attribute: DBAttribute;
    var attribute2: DBAttribute;
    contMergeDetails.removeAllChildren();
    if(binaryExpression.firstExpression is UnaryExpression){
        attribute = DBAttribute(UnaryExpression(_mapperModel.selectedMerge.relation.firstExpression).informationSource);
        hboxMergeDetail = new HBoxMergeDetail(_mapperModel.selectedMerge.URI, _mapperModel.selectedMerge.relation.URI, attribute.URI,

attribute.source);

        attribute2 = DBAttribute(UnaryExpression(_mapperModel.selectedMerge.relation.secondExpression).informationSource);
        hboxMergeDetail.dbAttribute2URI = attribute2.URI;
        hboxMergeDetail.lblAttribute2Text = attribute2.source;
        contMergeDetails.addChild(hboxMergeDetail);
    }else{
        while(binaryExpression.secondExpression is BinaryExpression){
            attribute = DBAttribute(UnaryExpression(BinaryExpression(binaryExpression.firstExpression).firstExpression).informationSource);
            hboxMergeDetail = new HBoxMergeDetail(_mapperModel.selectedMerge.URI, BinaryExpression(binaryExpression.firstExpression).URI, attribute.URI,

attribute.source);

            attribute2 = DBAttribute(UnaryExpression(BinaryExpression(binaryExpression.firstExpression).secondExpression).informationSource);
            hboxMergeDetail.dbAttribute2URI = attribute2.URI;
            hboxMergeDetail.lblAttribute2Text = attribute2.source;
            contMergeDetails.addChild(hboxMergeDetail);
        }
    }
}

private function contMergeDetailsDragDropHandler(event: DragEvent): void{

```

```

var ds:DragSource = event.dragSource;
var dropTarget:VBox = VBox(event.currentTarget);
var arr:Array;

if( ds.hasFormat("items") ) {
    arr = ds.dataForFormat("items") as Array;
}else if( ds.hasFormat("treeltems") ) {
    arr = ds.dataForFormat("treeltems") as Array;
}

if(_mapperModel.selectedMerge){
    for(var i:Number=0; i < arr.length; i++) {
        var node:XML = XML(arr[i]);
        var hboxMergeDetail: HBoxMergeDetail;
        var attribute: DBAttribute;

        if(node.localName() == "attribute"){
            attribute = OntologyManager.getDBAttributeByURIName(node.parent().@uriName, node.@uriName);
            var unaryFirstExpression: UnaryExpression;
            var unarySecondExpression: UnaryExpression;
            if(contMergeDetails.getChildren().length==0){
                unaryFirstExpression = OntologyManager.createUnaryExpression();
                unaryFirstExpression.informationSource = attribute;

                unarySecondExpression = OntologyManager.createUnaryExpression();

                var binaryExpression: BinaryExpression =
OntologyManager.createBinaryExpression(BinaryExpression$Type.EQUAL);

                binaryExpression.firstExpression = unaryFirstExpression;
                binaryExpression.secondExpression = unarySecondExpression;

                _mapperModel.selectedMerge.relation = binaryExpression;

                hboxMergeDetail = new HBoxMergeDetail(_mapperModel.selectedMerge.URI,
binaryExpression.URI, attribute.URI, attribute.source);

                contMergeDetails.addChild(hboxMergeDetail);

            }else{
                var hboxMergeDetailLast: HBoxMergeDetail =
HBoxMergeDetail(contMergeDetails.getChildAt(contMergeDetails.getChildren().length-1));
                if(hboxMergeDetailLast.dbAttribute2URI){
                    var binaryExpressionParent: BinaryExpression =
OntologyManager.createBinaryExpression(BinaryExpression$Type.AND);

                    binaryExpressionParent.firstExpression =
OntologyManager.getBinaryExpression(hboxMergeDetailLast.binaryExpressionURI, hboxMergeDetailLast.mergeURI);

                    unaryFirstExpression = OntologyManager.createUnaryExpression();
                    unaryFirstExpression.informationSource = attribute;
                    unarySecondExpression = OntologyManager.createUnaryExpression();

                    var binaryExpression2: BinaryExpression =
OntologyManager.createBinaryExpression(BinaryExpression$Type.EQUAL);

                    binaryExpression2.firstExpression = unaryFirstExpression;
                    binaryExpression2.secondExpression = unarySecondExpression;
                    binaryExpressionParent.secondExpression = binaryExpression2;

                    if(contMergeDetails.getChildren().length==1){
                        _mapperModel.selectedMerge.relation =
binaryExpressionParent;

                    }else{
                        var hboxMergeDetailBeforeLast: HBoxMergeDetail =
HBoxMergeDetail(contMergeDetails.getChildAt(contMergeDetails.getChildren().length-2));
                        var binaryExpressionBeforeLast: BinaryExpression =
OntologyManager.getBinaryExpression(hboxMergeDetailBeforeLast.binaryExpressionURI, hboxMergeDetailBeforeLast.mergeURI);
                        binaryExpressionBeforeLast.secondExpression =
binaryExpressionParent;

                    }

                    hboxMergeDetail = new
HBoxMergeDetail(_mapperModel.selectedMerge.URI, binaryExpression2.URI, attribute.URI, attribute.source);
                    contMergeDetails.addChild(hboxMergeDetail);

                }else{
                    Alert.show("É necessário preencher primeiro o segundo atributo da junção
anterior.", "Atenção!");
                }
            }
        }
    }
}

}else{
    Alert.show("Crie antes um relacionamento entre tabelas.", "Atenção!");
}
}

]]></mx:Script>

<mx:Button label="Gerar Elementos da Ontologia de Fontes de Informação" id="btnCreateInstances" click="btnCreateInstancesClickHandler()"
toolTip="Instanciar Elementos da Ontologia de Análise" width="380" height="40" icon="@Embed(source=/assets/mapping.gif)" left="10" top="10"/>
<view:BtnLoadOntology right="68"/>
<view:BtnSaveOntology right="10"/>
<mx:HRule y="60" left="10" right="10"/>

```

```

<mx:HBox top="70" right="0" left="10" bottom="10">
  <mx:Canvas width="40%" height="100%">
    <view:TreeISO id="treeISO1" dataProvider="{_mapperModel.dataSourcesTreeData}" click="treeISO1ClickHandler(event)" top="26"
bottom="10" cornerRadius="10" right="0" left="0" dragEnabled="true" allowMultipleSelection="true" dropEnabled="false" dragMoveEnabled="false"/>
    <mx:Label text="Elementos da Ontologia de Fontes de Informação" fontWeight="bold" left="4" y="7"/>
  </mx:Canvas>
  <mx:Canvas width="60%" height="100%">
    <mx:Label y="7" fontWeight="bold" left="14" text="Configuração dos Relacionamentos entre Tabelas"/>
    <mx:Canvas borderStyle="solid" cornerRadius="10" left="10" right="10" bottom="10" top="26">
      <mx:Label y="10" left="15" fontWeight="bold"/>
      <mx:Label y="75" left="15" fontWeight="bold"/>
      <mx:Canvas y="27" width="275" height="30" cornerRadius="10" borderStyle="solid" left="10">
        <mx:Label left="10" verticalCenter="0" id="lblSelectedDBCollection" fontSize="12" fontWeight="bold"/>
      </mx:Canvas>
      <mx:Canvas y="93" height="138" cornerRadius="10" borderStyle="solid" left="10" right="10">
        <mx:VBox id="contMergedTables" dragEnter="contMergedTablesDragEnterHandler(event)"
dragOver="contMergedTablesDragOverHandler(event)" dragDrop="contMergedTablesDragDropHandler(event)" backgroundColor="#FFFFFF" backgroundAlpha="0" left="10"
top="10" right="20" bottom="10" />
      </mx:Canvas>
      <mx:Label y="249" fontWeight="bold" left="15"/>
      <mx:Canvas cornerRadius="10" borderStyle="solid" right="10" left="10" bottom="140" top="266">
        <mx:VBox id="contMergeDetails" dragEnter="contMergeDetailsDragEnterHandler(event)"
dragOver="contMergeDetailsDragOverHandler(event)" dragDrop="contMergeDetailsDragDropHandler(event)" backgroundColor="#FFFFFF" backgroundAlpha="0" left="10"
top="10" right="20" bottom="10" />
      </mx:Canvas>
      <mx:Label fontWeight="bold" left="15" bottom="110"/>
      <mx:Canvas cornerRadius="10" borderStyle="solid" left="10" right="10" bottom="10" height="101">
        <mx:Label x="10" y="5" text="Nome do campo"/>
        <mx:TextInput y="22" editable="false" left="10" width="50%" id="txtDBAttributeName"/>
        <mx:Label x="10" y="50" text="Tipo"/>
        <mx:TextInput y="67" editable="false" left="10" width="160" id="txtDBAttributeType"/>
        <mx:Label x="178" y="50" text="Tamanho"/>
        <mx:TextInput y="67" editable="false" left="179" width="100" id="txtDBAttributeSize"/>
      </mx:Canvas>
    </mx:Canvas>
  </mx:HBox>
</mx:Canvas>

```

MapperBI.mxml

```

<?xml version="1.0" encoding="utf-8"?>
<!--Classe que representa o conteúdo da aba de mapeamento e configuração da ontologia de BI-->
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" xmlns:view="br.ufsc.inf.feco.mapper.view.*"
label="Ontologia Analítica" width="100%" height="100%" initialize="init!"/>
<mx:Script><![CDATA[
import br.ufsc.inf.feco.mapper.events.GetOntologyInstancesEvent;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Detail;
import mx.events.CloseEvent;
import mx.core.UIComponent;
import mx.controls.Alert;
import mx.core.Application;
import br.ufsc.inf.feco.mapper.model.MapperModel;
import br.ufsc.inf.feco.mapper.model.MapperModelLocator;
import mx.effects.Resize;
import mx.core.IFlexDisplayObject;
import mx.managers.PopUpManager;
import br.ufsc.inf.feco.mapper.view.popups.ConfigTheme;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.AnalyticalElement;
import mx.managers.DragManager;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;
import mx.core.DragSource;
import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementUpdateLabel;
import br.ufsc.inf.feco.mapper.events.ConfigThemeEvent;
import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementRemove;
import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementClick;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.Element;
import br.ufsc.inf.feco.mapper.constants.ElementTypes;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.GroupingUnit;
import org.granite.collections.IMap;
import mx.effects.Move;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBCollection;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.MeasureUnit;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.UnaryExpression;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBAttribute;
import mx.collections.ArrayCollection;
import mx.events.EffectEvent;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.BIThemeUnit;
import br.ufsc.inf.feco.ontologymanager.OntologyManager;
import mx.events.DragEvent;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Grouping;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.ThemeUnit;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Theme;

```

```

import br.ufsc.inf.feco.mapper.events.PanelHeuristicsConfigExecute;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label;

private var _mapperModel: MapperModel = MapperModelLocator.getInstance().mapperModel;

[Bindable]
[Embed(source="assets/define_hierarchy.gif")]
public var iconDefineHierarchy:Class;

[Bindable]
[Embed(source="assets/back_to_mapping.png")]
public var iconBackToMapping:Class;

[Bindable]
[Embed(source="assets/details2.gif")]
public var iconThemeUnitDetails:Class;

[Bindable]
[Embed(source="assets/measures.gif")]
public var iconMeasures:Class;

private function init(): void{
    addEventListener(HBoxAnalyticalElementClick.LINKBUTTONCLICK, hboxAnalyticalElementClickHandler);
    addEventListener(HBoxAnalyticalElementRemove.REMOVE, hboxAnalyticalElementRemoveHandler);
    addEventListener(HBoxAnalyticalElementUpdateLabel.UPDATE_LABEL, hboxAnalyticalElementUpdateLabelHandler);
}

////////////////////////////////////
//INVOCAÇÃO DO SERVIÇO E CARREGAMENTO DAS INSTÂNCIAS DA ONTOLOGIA DE BI////////////////////////////////////
////////////////////////////////////

private function btnCreateBIInstancesClickHandler() : void {
    dispatchEvent(new GetOntologyInstancesEvent(GetOntologyInstancesEvent.GENERATE_BI, true));
}

public function loadThemeUnits(theme: Theme): void{
    var themeUnits: ArrayCollection;
    var themeUnit: BIThemeUnit;
    var hboxAnalyticalElement: HBoxAnalyticalElement;

    themeUnits = theme.themeUnits.values;

    contThemeUnits.removeAllChildren();

    var i: uint;
    for(i=0; i< themeUnits.length; i++){
        themeUnit = BIThemeUnit(themeUnits[i]);
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.THEME_UNIT, theme.URI, themeUnit.URI,
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(themeUnit.labels.values[0]).label);
        contThemeUnits.addChild(hboxAnalyticalElement);
    }

    if(themeUnits.length > 0){
        selectThemeUnit(BIThemeUnit(themeUnits[0]));
        HBoxAnalyticalElement(contThemeUnits.getChildAt(0)).selectLinkButton(true);
    }else{
        contMeasures.removeAllChildren();
        contGrouping.removeAllChildren();
        contGroupingUnits.removeAllChildren();
    }
}

private function selectThemeUnit(themeUnit: BIThemeUnit): void{
    _mapperModel.selectedThemeUnit = themeUnit;
    lblSelectedThemeUnit.text = "(Unidade de Tema: " +
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(_mapperModel.selectedThemeUnit.labels.values[0]).label + ")";
    lblSelectedThemeUnit2.text = "(Unidade de Tema: " +
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(_mapperModel.selectedThemeUnit.labels.values[0]).label + ")";

    loadThemeUnitMeasures(_mapperModel.selectedThemeUnit);
    loadThemeUnitGroupings(_mapperModel.selectedThemeUnit);
    loadThemeUnitDetails(_mapperModel.selectedThemeUnit);
}

private function loadThemeUnitMeasures(themeUnit: BIThemeUnit): void{
    var measureUnits: ArrayCollection;
    var measureUnit: MeasureUnit;
    var hboxAnalyticalElement: HBoxAnalyticalElement;

    measureUnits = themeUnit.measureUnits.values;
    contMeasures.removeAllChildren();

    var i: uint;
    for(i=0; i< measureUnits.length; i++){
        measureUnit = MeasureUnit(measureUnits[i]);
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.MEASURE, themeUnit.URI, measureUnit.URI,
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(measureUnit.labels.values[0]).label);
        contMeasures.addChild(hboxAnalyticalElement);
    }
}

```

```

}

private function loadThemeUnitDetails(themeUnit: ThemeUnit): void{
    var details: ArrayCollection;
    var detail: Detail;
    var hboxAnalyticalElement: HBoxAnalyticalElement;

    details = themeUnit.details.values;
    contThemeUnitDetails.removeAllChildren();

    var i: uint;
    for(i=0; i< details.length; i++){
        detail = Detail(details[i]);
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.DETAIL, themeUnit.URI, detail.URI,
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(detail.labels.values[0]).label);
        contThemeUnitDetails.addChild(hboxAnalyticalElement);
    }
}

private function loadThemeUnitGroupings(themeUnit: ThemeUnit): void{
    var groupings: ArrayCollection;
    var grouping: Grouping;
    var hboxAnalyticalElement: HBoxAnalyticalElement

    groupings = themeUnit.groupings.values;
    contGrouping.removeAllChildren();

    var i: uint;
    for(i=0; i< groupings.length; i++){
        grouping = Grouping(groupings[i]);
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.GROUPING, themeUnit.URI, grouping.URI,
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(grouping.labels.values[0]).label);
        contGrouping.addChild(hboxAnalyticalElement);
    }

    if(groupings.length > 0){
        _mapperModel.selectedGrouping = Grouping(groupings[0]);
        HBoxAnalyticalElement(contGrouping.getChildAt(0)).selectLinkButton(true);
        loadGroupingUnits(_mapperModel.selectedGrouping);
    }else{
        contGroupingUnits.removeAllChildren();
    }
}

private function loadGroupingUnits(grouping: Grouping): void{
    var groupingUnits: ArrayCollection;
    var groupingUnit: GroupingUnit;
    var hboxAnalyticalElement: HBoxAnalyticalElement;

    groupingUnits = grouping.groupingUnits.values;
    contGroupingUnits.removeAllChildren();

    var i: uint;
    for(i=0; i< groupingUnits.length; i++){
        groupingUnit = GroupingUnit(groupingUnits[i]);
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.GROUPING_UNIT, grouping.URI, groupingUnit.URI,
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(groupingUnit.labels.values[0]).label);
        //habilitar drag pra poder arrastar os GU pros containers que definem as hierarquias dos GUS
        hboxAnalyticalElement.addEventListener(MouseEvent.CLICK, mouseMoveHandler);
        contGroupingUnits.addChild(hboxAnalyticalElement);
    }

    if(groupingUnits.length>0)
        selectHBoxAnalyticalElement(ElementTypes.GROUPING_UNIT, GroupingUnit(groupingUnits[0]));
}

private function mouseMoveHandler(event: MouseEvent):void {
    var dragInitiator: HBoxAnalyticalElement=HBoxAnalyticalElement(event.currentTarget);

    var dragLabel: String = dragInitiator.getLabel();
    var dragURI: URI_AS = dragInitiator.URI;
    var dragParentURI: URI_AS = dragInitiator.parentURI;

    var ds:DragSource = new DragSource();

    ds.addData(dragLabel, 'label');
    ds.addData(dragURI, 'uri');
    ds.addData(dragParentURI, 'parentUri');

    DragManager.doDrag(dragInitiator, ds, event);
}

////////////////////////////////////////////////////////////////////

private function createBITHemeUnit(label: String): BITHemeUnit{
    var element: BITHemeUnit = OntologyManager.createBITHemeUnit(label);
    _mapperModel.selectedTheme.themeUnits.put(element.URI, element);
    return element;
}

private function createGrouping(label: String): Grouping{
    var element: Grouping = OntologyManager.createGrouping(label);
}

```



```

        _mapperModel.selectedThemeUnit.groupings.put(element.URI, element);
        return element;
    }

private function createDetail(label: String): Detail{
    var element: Detail = OntologyManager.createDetail(label);
    _mapperModel.selectedThemeUnit.details.put(element.URI, element);
    return element;
}

//////////////////////////////////////////////////////////////////////////
//DRAG AND DROP OPERATIONS//////////////////////////////////////////////////////////////////////////
//////////////////////////////////////////////////////////////////////////

private function contThemeUnitDragEnterHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        var ds: DragSource = event.dragSource;
        if(!ds.hasFormat("treeltems")) return;
        var items: Array = ds.dataForFormat("treeltems") as Array;
        //percorre todos os treitens selecionados para drag
        for(var i:Number=0; i < items.length; i++) {
            var item:XML = XML(items[i]);
            if( item.localName() != "table" ) return;
        }
        DragManager.acceptDragDrop(UIComponent(event.currentTarget));
    }
}

//Mostra o sinal de + se pode ser copiado
private function contThemeUnitDragOverHandler(event: DragEvent): void{
    if( event.dragInitiator is TreeISO) {
        DragManager.showFeedback(DragManager.COPY);
    }
}

private function contThemeUnitDragDropHandler(event: DragEvent): void{
    var ds:DragSource = event.dragSource;
    var dropTarget:VBox = VBox(event.currentTarget);
    var arr:Array;

    if( ds.hasFormat("items") ) {
        arr = ds.dataForFormat("items") as Array;
    }else if( ds.hasFormat("treeltems") ) {
        arr = ds.dataForFormat("treeltems") as Array;
    }

    for(var i:Number=0; i < arr.length; i++) {
        var node:XML = XML(arr[i]);
        if(_mapperModel.selectedTheme){
            var hboxAnalyticalElement: HBoxAnalyticalElement;
            var tuLabel: String = "TU." + node.@source;
            var tu: BIThemeUnit = createBIThemeUnit(tuLabel);
            hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.THEME_UNIT, _mapperModel.selectedTheme.URI, tu.URI, tuLabel);
            contThemeUnits.addChild(hboxAnalyticalElement);
            deselectSelectedThemeUnitHBoxAnalyticalElement();
            _mapperModel.selectedThemeUnit = tu;

            var measure: MeasureUnit;
            var tabela: DBCollection;
            tabela = OntologyManager.getDBCollectionByURIName(node.@uriName);
            var atributos: IMap = tabela.informationSources;
            var atributo: DBAttribute;
            var j: uint;
            for (j=0; j< atributos.length; j++){
                atributo = atributos.values[j];
                measure = OntologyManager.createMeasureUnitByDBAttribute(atributo);
                _mapperModel.selectedThemeUnit.measureUnits.put(measure.URI, measure);
            }

            selectHBoxAnalyticalElement(ElementTypes.THEME_UNIT, tu);
        }else{
            Alert.show("Selecione um tema antes.", "Atenção!");
        }
    }
}

private function contMeasuresDragEnterHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        var ds: DragSource = event.dragSource;
        if(!ds.hasFormat("treeltems")) return;
        var items: Array = ds.dataForFormat("treeltems") as Array;
        //percorre todos os treitens selecionados para drag
        for(var i:Number=0; i < items.length; i++) {
            var item:XML = XML(items[i]);
            if((item.localName() != "table" ) && (item.localName() != "attribute"))
                return;
        }
        DragManager.acceptDragDrop(UIComponent(event.currentTarget));
    }
}

//Mostra o sinal de + se pode ser copiado

```

```

private function contMeasuresDragOverHandler(event: DragEvent): void{
    if( event.dragInitiator is TreeISO ) {
        DragManager.showFeedback(DragManager.COPY);
    }
}

private function contMeasuresDragDropHandler(event: DragEvent): void{
    var ds:DragSource = event.dragSource;
    var dropTarget:VBox = VBox(event.currentTarget);
    var arr:Array;

    if( ds.hasFormat("items") ) {
        arr = ds.dataForFormat("items") as Array;
    }else if( ds.hasFormat("treeltems") ) {
        arr = ds.dataForFormat("treeltems") as Array;
    }

    for(var i:Number=0; i < arr.length; i++) {
        var node:XML = XML(arr[i]);
        if(_mapperModel.selectedThemeUnit){
            var hboxAnalyticalElement: HBoxAnalyticalElement;
            var measure: MeasureUnit;
            var tabela: DBCollection;
            var atributo: DBAttribute;

            if(node.localName() == "table"){
                tabela = OntologyManager.getDBCollectionByURIName(node.@uriName);
                var atributos: IMap = tabela.informationSources;
                var j: uint;
                for (j=0; j< atributos.length; j++){
                    atributo = atributos.values[j];
                    measure = OntologyManager.createMeasureUnitByDBAttribute(atributo);
                    _mapperModel.selectedThemeUnit.measureUnits.put(measure.URI, measure);
                    hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.MEASURE,
_mapperModel.selectedThemeUnit.URI, measure.URI, br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(measure.labels.values[0]).label);
                    contMeasures.addChild(hboxAnalyticalElement);
                }
            }else if(node.localName() == "attribute"){
                atributo = OntologyManager.getDBAttributeByURIName(node.parent().@uriName, node.@uriName);
                measure = OntologyManager.createMeasureUnitByDBAttribute(atributo);
                _mapperModel.selectedThemeUnit.measureUnits.put(measure.URI, measure);
                hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.MEASURE, _mapperModel.selectedThemeUnit.URI,
measure.URI, br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(measure.labels.values[0]).label);
                contMeasures.addChild(hboxAnalyticalElement);
            }
        }
    }

    }else{
        Alert.show("Selecione uma unidade de tema antes.", "Atenção!");
    }
}

private function contGroupingDragEnterHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        var ds: DragSource = event.dragSource;
        if(!ds.hasFormat("treeltems")) return;
        var items: Array = ds.dataForFormat("treeltems") as Array;
        //percorre todos os treeitens selcionados para drag
        for(var i:Number=0; i < items.length; i++) {
            var item:XML = XML(items[i]);
            if((item.localName() != "table"))
                return;
        }
        DragManager.acceptDragDrop(UIComponent(event.currentTarget));
    }
}

//Mostra o sinal de + se pode ser copiado
private function contGroupingDragOverHandler(event: DragEvent): void{
    if( event.dragInitiator is TreeISO ) {
        DragManager.showFeedback(DragManager.COPY);
    }
}

private function contGroupingDragDropHandler(event: DragEvent): void{
    var ds:DragSource = event.dragSource;
    var dropTarget:VBox = VBox(event.currentTarget);
    var arr:Array;

    if( ds.hasFormat("items") ) {
        arr = ds.dataForFormat("items") as Array;
    }else if( ds.hasFormat("treeltems") ) {
        arr = ds.dataForFormat("treeltems") as Array;
    }

    for(var i:Number=0; i < arr.length; i++) {
        var node:XML = XML(arr[i]);
        if(_mapperModel.selectedThemeUnit){
            var hboxAnalyticalElement: HBoxAnalyticalElement;
            var atributo: DBAttribute;
            var groupingLabel: String = "GR." + node.@source;
            var grouping: Grouping = createGrouping(groupingLabel);

```

```

groupingLabel);
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.GROUPING, _mapperModel.selectedThemeUnit.URI, grouping.URI,
        contGrouping.addChild(hboxAnalyticalElement);
deselectSelectedGroupingHBoxAnalyticalElement();
_mapperModel.selectedGrouping = grouping;
        var groupingUnit: GroupingUnit;
        var groupingUnitLabel: String;
        for each (var attribute: XML in node.attribute){
            atributo = OntologyManager.getDBAttributeByURIName(node.@uriName, attribute.@uriName);
            groupingUnit = OntologyManager.createGroupingUnitByDBAttribute(atributo);
            _mapperModel.selectedGrouping.groupingUnits.put(groupingUnit.URI, groupingUnit);

            hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.GROUPING_UNIT,
_mapperModel.selectedGrouping.URI, groupingUnit.URI, br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(groupingUnit.labels.values[0]).label);
            contGroupingUnits.addChild(hboxAnalyticalElement);
        }
        selectHBoxAnalyticalElement(ElementTypes.GROUPING, grouping);
    }else{
        Alert.show("Selecione uma unidade de tema antes.", "Atenção!");
    }
    }
    contThemeUnitDragExitHandler(event);
}

private function contGroupingUnitsDragEnterHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO){
        var ds: DragSource = event.dragSource;
        if(ds.hasFormat("treeltems")) return;
        var items: Array = ds.dataForFormat("treeltems") as Array;
        //percorre todos os treeltems selecionados para drag
        for(var i:Number=0; i < items.length; i++) {
            var item:XML = XML(items[i]);
            if((item.localName() != "table") && (item.localName() != "attribute"))
                return;
        }
        DragManager.acceptDragDrop(UIComponent(event.currentTarget));
    }
}

//Mostra o sinal de + se pode ser copiado
private function contGroupingUnitsDragOverHandler(event: DragEvent): void{
    if(event.dragInitiator is TreeISO) {
        DragManager.showFeedback(DragManager.COPY);
    }
}

private function contGroupingUnitsDragDropHandler(event: DragEvent): void{
    if(!(event.dragInitiator is HBoxAnalyticalElement)){
        var ds: DragSource = event.dragSource;
        var dropTarget: VBox = VBox(event.currentTarget);
        var arr: Array;

        if(ds.hasFormat("items")) {
            arr = ds.dataForFormat("items") as Array;
        }else if(ds.hasFormat("treeltems")) {
            arr = ds.dataForFormat("treeltems") as Array;
        }

        for(var i:Number=0; i < arr.length; i++) {
            var node:XML = XML(arr[i]);
            if(_mapperModel.selectedGrouping){
                var hboxAnalyticalElement: HBoxAnalyticalElement;
                var groupingUnit: GroupingUnit;
                var tabela: DBCollection;
                var atributo: DBAttribute;

                if(node.localName() == "table"){
                    tabela = OntologyManager.getDBCollectionByURIName(node.@uriName);
                    var atributos: IMap = tabela.informationSources;
                    var j: uint;
                    for (j=0; j< atributos.length; j++){
                        atributo = atributos.values[j];
                        groupingUnit = OntologyManager.createGroupingUnitByDBAttribute(atributo);
                        _mapperModel.selectedGrouping.groupingUnits.put(groupingUnit.URI, groupingUnit);

                        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.GROUPING_UNIT,
_mapperModel.selectedGrouping.URI, groupingUnit.URI, br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(groupingUnit.labels.values[0]).label);
                        contGroupingUnits.addChild(hboxAnalyticalElement);
                        if(j==0)
                            selectHBoxAnalyticalElement(ElementTypes.GROUPING_UNIT, groupingUnit);
                    }
                }else if(node.localName() == "attribute"){
                    atributo = OntologyManager.getDBAttributeByURIName(node.parent().@uriName, node.@uriName);

                    groupingUnit = OntologyManager.createGroupingUnitByDBAttribute(atributo);
                    _mapperModel.selectedGrouping.groupingUnits.put(groupingUnit.URI, groupingUnit);

                    hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.GROUPING_UNIT,
_mapperModel.selectedGrouping.URI, groupingUnit.URI, br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(groupingUnit.labels.values[0]).label);
                    contGroupingUnits.addChild(hboxAnalyticalElement);
                    selectHBoxAnalyticalElement(ElementTypes.GROUPING_UNIT, groupingUnit);
                }
            }
        }
    }
}

```

```

        }else{
            Alert.show("Selecione um agrupamento antes.", "Atenção!");
        }
    }
}

private function contGroupingUnitHierarchyDragEnterHandler(event: DragEvent): void {
if (event.dragSource.hasFormat("label")) {
var dropTarget: Canvas = Canvas(event.currentTarget);
if (dropTarget == contGroupingUnitSelected){

    DragManager.acceptDragDrop(dropTarget);
}else if (dropTarget == contGroupingUnitHierarquicalParent){

    DragManager.acceptDragDrop(dropTarget);
}else if (dropTarget == contGroupingUnitHierarquicalChild){

    DragManager.acceptDragDrop(dropTarget);
}
}
}

//Mostra o sinal de + se pode ser copiado
private function contGroupingUnitHierarchyDragOverHandler(event: DragEvent): void{
if (event.dragInitiator is HBoxAnalyticalElement) {
    DragManager.showFeedback(DragManager.COPY);
}
}

private function contGroupingUnitHierarchyDragDropHandler(event: DragEvent): void {
var dropTarget: Canvas = Canvas(event.currentTarget);
var uri: URI_AS = URI_AS(event.dragSource.dataForFormat("uri"));
var parentUri: URI_AS = URI_AS(event.dragSource.dataForFormat("parentUri"));
var label: String = String(event.dragSource.dataForFormat("label"));
if (dropTarget == contGroupingUnitHierarquicalParent){
    _mapperModel.selectedGroupingUnit.hierarchicalParent = OntologyManager.getGroupingUnit(parentUri, uri);
    _mapperModel.selectedGroupingUnit.hierarchicalParent.hierarchicalChild = _mapperModel.selectedGroupingUnit;
    lblGroupingUnitHierarquicalParent.text = label;
}else if (dropTarget == contGroupingUnitHierarquicalChild){
    _mapperModel.selectedGroupingUnit.hierarchicalChild = OntologyManager.getGroupingUnit(parentUri, uri);
    _mapperModel.selectedGroupingUnit.hierarchicalChild.hierarchicalParent = _mapperModel.selectedGroupingUnit;
    lblGroupingUnitHierarquicalChild.text = label;
}
}

private function contThemeUnitDetailsDragEnterHandler(event: DragEvent): void{
if (event.dragInitiator is TreelSO){
var ds: DragSource = event.dragSource;
if (ds.hasFormat("treelitems")) return;
var items: Array = ds.dataForFormat("treelitems") as Array;
//percorre todos os treelitems selecionados para drag
for (var i: Number=0; i < items.length; i++){
var item: XML = XML(items[i]);
if ((item.localName() != "table") && (item.localName() != "attribute"))
return;
}
}
DragManager.acceptDragDrop(UIComponent(event.currentTarget));
}

private function contThemeUnitDetailsDragOverHandler(event: DragEvent): void{
if (event.dragInitiator is TreelSO){
    DragManager.showFeedback(DragManager.COPY);
}
}

private function contThemeUnitDetailsDragDropHandler(event: DragEvent): void{
var ds: DragSource = event.dragSource;
var dropTarget: VBox = VBox(event.currentTarget);
var arr: Array;

if (ds.hasFormat("items")) {
arr = ds.dataForFormat("items") as Array;
}else if (ds.hasFormat("treelitems")) {
arr = ds.dataForFormat("treelitems") as Array;
}

for (var i: Number=0; i < arr.length; i++) {
var node: XML = XML(arr[i]);
if (_mapperModel.selectedThemeUnit){
var hboxAnalyticalElement: HBoxAnalyticalElement;
var detail: Detail;
var detailLabel: String;
var tabela: DBCollection;
var atributo: DBAttribute;
if (node.localName() == "table"){
tabela = OntologyManager.getDBCollectionByURIName(node.@uriName);
var atributos: IMap = tabela.informationSources;
var j: uint;
for (j=0; j < atributos.length; j++){
atributo = atributos.values[j];
detailLabel = "Detail.";
}
}
}
}
}

```

```

        detailLabel += atributo.source;
        detail = createDetail(detailLabel);
        detail.attribute = atributo;
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.DETAIL,
_mapperModel.selectedThemeUnit.URI, detail.URI, detailLabel);
        contThemeUnitDetails.addChild(hboxAnalyticalElement);
    }
    }else if(node.localName() == "attribute"){
        atributo = OntologyManager.getDBAttributeByURIName(node.parent().@uriName, node.@uriName);
        detailLabel = "Detail.";
        detailLabel += node.@source;
        detail = createDetail(detailLabel);
        detail.attribute = atributo;
        hboxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.DETAIL,
_mapperModel.selectedThemeUnit.URI, detail.URI, detailLabel);
        contThemeUnitDetails.addChild(hboxAnalyticalElement);
    }
}
}
}
}
}
}

////////////////////////////////////
//REMANEJAMENTO DA GUI E EFEITOS////////////////////////////////////
////////////////////////////////////

private function btnDefineHierarchyClickHandler(): void{
    if(Number(canvasGroupingsConfig.getStyle("top")) == 450){
        maximizeCanvasGroupingsConfig();
        //mudar icone
        btnDefineHierarchy.setStyle("icon",iconBackToMapping);
        btnDefineHierarchy.setToolTip = "Retornar para a tela de mapeamento";
    }else{
        minimizeCanvasGroupingsConfig();
        btnDefineHierarchy.setStyle("icon",iconDefineHierarchy);
        btnDefineHierarchy.setToolTip = "Definir hierarquia das unidades de agrupamento";
    }
}

private function maximizeCanvasGroupingsConfig(): void{
    var move: Move = new Move;
    move.duration = 750;
    move.yFrom = canvasGroupingsConfig.y;
    move.yTo = 0;
    move.addEventListener(Event.EFFECT_END, maximizeEffectFinished);
    canvasAnalyticalSuperior.visible = false;
    var hboxGroupingsHeight: Number = hboxGroupings.height;
    hboxGroupings.height = hboxGroupingsHeight;
    move.play([canvasGroupingsConfig],false);
}

private function maximizeEffectFinished(event: EffectEvent): void{
    canvasGroupingsConfig.setStyle("top", 0);

    var resize: Resize = new Resize();
    resize.duration = 750;
    resize.heightFrom = hboxGroupings.height;
    resize.heightTo = canvasWithBorderAnalyticalElements.height - canvasDefineHierarchy.height - 60;
    resize.addEventListener(Event.EFFECT_END, resizeEffectFinished);
    resize.play([hboxGroupings],false);

    canvasDefineHierarchy.clearStyle("top");
    canvasDefineHierarchy.height = 215;
    canvasDefineHierarchy.setStyle("bottom",10);
    canvasDefineHierarchy.visible = true;
}

private function minimizeCanvasGroupingsConfig(): void{
    var resize: Resize = new Resize();
    resize.duration = 750;
    resize.heightFrom = hboxGroupings.height;
    resize.heightTo = canvasWithBorderAnalyticalElements.height - 450 - 25;
    resize.addEventListener(Event.EFFECT_END, minimizeEffectFinished);
    resize.play([hboxGroupings],false);
}

private function minimizeEffectFinished(event: EffectEvent): void{
    backTeste();
}

private function backTeste(): void{
    var move: Move = new Move;
    move.duration = 750;
    move.yFrom = 0;
    move.yTo = 450;
    move.addEventListener(Event.EFFECT_END, backTesteEffectFinished);
    canvasDefineHierarchy.setStyle("top", 0);
    canvasDefineHierarchy.percentHeight=0;
}

```

```

        canvasDefineHierarchy.visible = false;
        move.play([canvasGroupingsConfig],false);
    }

    private function backTesteEffectFinished(event: EffectEvent): void{
        canvasGroupingsConfig.setStyle("top", 450);
        hboxGroupings.percentHeight = 100;
        canvasAnalyticalSuperior.visible = true;
    }

    private function btnDefineDetailsClickHandler(): void{
        if(canvasThemeUnitDetails.visible == false){
            canvasMeasures.visible = false;
            canvasThemeUnitDetails.visible = true;
            showCanvasThemeUnitDetails();
            btnDefineDetails.setStyle("icon",iconMeasures);
            btnDefineDetails.toolTip = "Definir medidas da Unidade de Tema";
        }else{
            canvasThemeUnitDetails.visible = false;
            canvasMeasures.visible = true;
            showCanvasMeasures();
            btnDefineDetails.setStyle("icon",iconThemeUnitDetails);
            btnDefineDetails.toolTip = "Definir detalhamento da Unidade de Tema";
        }
    }

    private function showCanvasThemeUnitDetails(): void{
        var move: Move = new Move;
        move.duration = 750;
        move.xFrom = canvasThemeUnitDetails.width;
        move.xTo = 0;
        move.addEventListener(Event.EFFECT_END, showCanvasThemeUnitDetailsEffectFinished);
        canvasWithBorderThemeUnitDetails.verticalScrollPolicy = "off";
        canvasWithBorderThemeUnitDetails.horizontalScrollPolicy = "off";
        move.play([canvasThemeUnitDetails],false);
    }

    private function showCanvasThemeUnitDetailsEffectFinished(event: EffectEvent): void{
        canvasWithBorderThemeUnitDetails.verticalScrollPolicy = "auto";
        canvasWithBorderThemeUnitDetails.horizontalScrollPolicy = "auto";
    }

    private function showCanvasMeasures(): void{
        var move: Move = new Move;
        move.duration = 750;
        move.xFrom = -canvasMeasures.width;
        move.xTo = 0;
        move.addEventListener(Event.EFFECT_END, showCanvasMeasuresEffectFinished);
        canvasWithBorderMeasures.verticalScrollPolicy = "off";
        canvasWithBorderMeasures.horizontalScrollPolicy = "off";
        move.play([canvasMeasures],false);
    }

    private function showCanvasMeasuresEffectFinished(event: EffectEvent): void{
        canvasWithBorderMeasures.verticalScrollPolicy = "auto";
        canvasWithBorderMeasures.horizontalScrollPolicy = "auto";
    }

    //////////////////////////////////////
    //OUTROS////////////////////////////////////
    //////////////////////////////////////

    private function hboxAnalyticalElementClickHandler(event: HBoxAnalyticalElementClick): void{
        if(event.elementType == ElementTypes.THEME_UNIT){
            var themeUnit: B1ThemeUnit = B1ThemeUnit(OntologyManager.getThemeUnit(event.URI));
            if(themeUnit != _mapperModel.selectedThemeUnit){
                if(_mapperModel.selectedThemeUnit != null){
                    deselectSelectedThemeUnitHBoxAnalyticalElement();
                }
                selectThemeUnit(themeUnit);
            }
        }else if(event.elementType == ElementTypes.MEASURE){
            if(_mapperModel.selectedMeasure != null){
                var j: uint;
                for (j=0; j<contMeasures.getChildren().length; j++){
                    if(HBoxAnalyticalElement(contMeasures.getChildren()[j]).URI ==
                    _mapperModel.selectedMeasure.URI){
                        HBoxAnalyticalElement(contMeasures.getChildAt(j)).selectLinkButton(false);
                    }
                }
            }
            _mapperModel.selectedMeasure = MeasureUnit(OntologyManager.getMeasureUnit(event.parentURI, event.URI));
        }else if(event.elementType == ElementTypes.GROUPING){
            if(_mapperModel.selectedGrouping != null){
                var k: uint;
                for (k=0; k<contGrouping.getChildren().length; k++){
                    if(HBoxAnalyticalElement(contGrouping.getChildren()[k]).URI ==
                    _mapperModel.selectedGrouping.URI){
                        HBoxAnalyticalElement(contGrouping.getChildAt(k)).selectLinkButton(false);
                    }
                }
            }
        }
    }

```

```

    }
    _mapperModel.selectedGrouping = Grouping(OntologyManager.getGrouping(event.parentURI, event.URI));
    loadGroupingUnits(_mapperModel.selectedGrouping);
} else if (event.elementType == ElementTypes.GROUPING_UNIT) {
    if (_mapperModel.selectedGroupingUnit != null) {
        var i: uint;
        for (i=0; i<contGroupingUnits.getChildren().length; i++){
            if (HBoxAnalyticalElement(contGroupingUnits.getChildren()[i]).URI ==
_mapperModel.selectedGroupingUnit.URI) {
                HBoxAnalyticalElement(contGroupingUnits.getChildAt(i)).selectLinkButton(false);
            }
        }
    }
    _mapperModel.selectedGroupingUnit = GroupingUnit(OntologyManager.getGroupingUnit(event.parentURI, event.URI));
    populateContGroupingUnitHierarchy();
}
}

private function populateContGroupingUnitHierarchy(): void {
    if (_mapperModel.selectedGroupingUnit) {
        lblGroupingUnitSelected.text =
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(_mapperModel.selectedGroupingUnit.labels.values[0]).label;

        if (_mapperModel.selectedGroupingUnit.hierarchicalParent) {
            lblGroupingUnitHierarquicalParent.text =
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(_mapperModel.selectedGroupingUnit.hierarchicalParent.labels.values[0]).label;
        } else {
            lblGroupingUnitHierarquicalParent.text = "";
        }

        if (_mapperModel.selectedGroupingUnit.hierarchicalChild) {
            lblGroupingUnitHierarquicalChild.text =
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(_mapperModel.selectedGroupingUnit.hierarchicalChild.labels.values[0]).label;
        } else {
            lblGroupingUnitHierarquicalChild.text = "";
        }
    }
}

private function selectHBoxAnalyticalElement(type: ElementTypes, element: Element): void {
    var i: uint;
    if (type == ElementTypes.THEME_UNIT) {
        for (i=0; i<contThemeUnits.getChildren().length; i++){
            if (HBoxAnalyticalElement(contThemeUnits.getChildren()[i]).URI == element.URI) {
                HBoxAnalyticalElement(contThemeUnits.getChildAt(i)).selectLinkButton(true);
            }
        }
        _mapperModel.selectedThemeUnit = BThemeUnit(element);
        loadThemeUnitMeasures(_mapperModel.selectedThemeUnit);
        loadThemeUnitGroupings(_mapperModel.selectedThemeUnit);
    } else if (type == ElementTypes.GROUPING) {
        for (i=0; i<contGrouping.getChildren().length; i++){
            if (HBoxAnalyticalElement(contGrouping.getChildren()[i]).URI == element.URI) {
                HBoxAnalyticalElement(contGrouping.getChildAt(i)).selectLinkButton(true);
            }
        }
        _mapperModel.selectedGrouping = Grouping(element);
        loadGroupingUnits(_mapperModel.selectedGrouping);
    } else if (type == ElementTypes.GROUPING_UNIT) {
        for (i=0; i<contGroupingUnits.getChildren().length; i++){
            if (HBoxAnalyticalElement(contGroupingUnits.getChildren()[i]).URI == element.URI) {
                HBoxAnalyticalElement(contGroupingUnits.getChildAt(i)).selectLinkButton(true);
            }
        }
        _mapperModel.selectedGroupingUnit = GroupingUnit(element);
        populateContGroupingUnitHierarchy();
    }
}

private function deselectSelectedThemeUnitHBoxAnalyticalElement(): void {
    var i: uint;
    for (i=0; i<contThemeUnits.getChildren().length; i++){
        if (HBoxAnalyticalElement(contThemeUnits.getChildren()[i]).URI == _mapperModel.selectedThemeUnit.URI) {
            HBoxAnalyticalElement(contThemeUnits.getChildAt(i)).selectLinkButton(false);
        }
    }
    contMeasures.removeAllChildren();
    _mapperModel.selectedMeasure = null;
    contGrouping.removeAllChildren();
    _mapperModel.selectedGrouping = null;
    contGroupingUnits.removeAllChildren();
    _mapperModel.selectedGroupingUnit = null;
    populateContGroupingUnitHierarchy();
}

private function deselectSelectedGroupingHBoxAnalyticalElement(): void {
    if (_mapperModel.selectedGrouping) {
        var i: uint;

```

```

        for (i=0; i< contGrouping.getChildren().length; i++){
            if(HBoxAnalyticalElement(contGrouping.getChildren()[i]).URI == _mapperModel.selectedGrouping.URI){
                HBoxAnalyticalElement(contGrouping.getChildAt(i)).selectLinkButton(false);
                break;
            }
        }
        contGroupingUnits.removeAllChildren();
        _mapperModel.selectedGroupingUnit = null;
        populateContGroupingUnitHierarchy();
    }
}

private function hboxAnalyticalElementUpdateLabelHandler(event: HBoxAnalyticalElementUpdateLabel): void{
    var labelRequired: String = event.label;
    //Se já tem URI, é porque o recurso já existe e está apenas sendo alterado a sua label
    if(event.URI){
        if(event.elementType == ElementTypes.THEME_UNIT){
            var themeUnit: ThemeUnit = ThemeUnit(OntologyManager.getThemeUnit(event.URI));
            if(themeUnit){
                br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(themeUnit.labels.values[0]).label = event.label;
                lblSelectedThemeUnit.text = "(Unidade de Tema: " + event.label + ")";
                lblSelectedThemeUnit2.text = "(Unidade de Tema: " + event.label + ")";
            }
        }
        else if(event.elementType == ElementTypes.GROUPING){
            var grouping: Grouping = Grouping(OntologyManager.getGrouping(event.parentURI, event.URI));
            if(grouping){
                br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(grouping.labels.values[0]).label = event.label;
            }
        }
        else if(event.elementType == ElementTypes.GROUPING_UNIT){
            var groupingUnit: GroupingUnit = GroupingUnit(OntologyManager.getGroupingUnit(event.parentURI,
event.URI));
            if(groupingUnit){
                br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(groupingUnit.labels.values[0]).label = event.label;
            }
        }
        else if(event.elementType == ElementTypes.DETAIL){
            var detail: Detail = Detail(OntologyManager.getThemeUnitDetail(event.parentURI, event.URI));
            if(detail){
                br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(detail.labels.values[0]).label = event.label;
            }
        }
        else if(event.elementType == ElementTypes.MEASURE){
            var measure: MeasureUnit = MeasureUnit(OntologyManager.getMeasureUnit(event.parentURI,
event.URI));
            if(measure){
                br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(measure.labels.values[0]).label = event.label;
            }
        }
    }
    //Se não tem URI, é porque está sendo inserido um novo recurso
    else{
        var analyticalElement: AnalyticalElement;
        var hboxAnalyticalElement: HBoxAnalyticalElement;

        if(event.elementType == ElementTypes.THEME_UNIT){
            analyticalElement = createBIThemeUnit(labelRequired);
            hboxAnalyticalElement = HBoxAnalyticalElement(event.target);
            hboxAnalyticalElement.URI = analyticalElement.URI;
            hboxAnalyticalElement.setLabel(labelRequired);

            deselectSelectedThemeUnitHBoxAnalyticalElement();

            selectThemeUnit(BIThemeUnit(analyticalElement));
            hboxAnalyticalElement.selectLinkButton(true);
        }
        else if(event.elementType == ElementTypes.GROUPING){
            analyticalElement = createGrouping(labelRequired);
            hboxAnalyticalElement = HBoxAnalyticalElement(event.target);
            hboxAnalyticalElement.URI = analyticalElement.URI;
            hboxAnalyticalElement.setLabel(labelRequired);

            deselectSelectedGroupingHBoxAnalyticalElement();

            _mapperModel.selectedGrouping = Grouping(analyticalElement);
            hboxAnalyticalElement.selectLinkButton(true);
        }
    }
}

private function hboxAnalyticalElementRemoveHandler(event: HBoxAnalyticalElementRemove): void{
    trace(event.URI);
    OntologyManager.deletedURIs.push(event.URI);
    var i: uint;
    if(event.elementType == ElementTypes.THEME_UNIT){
        //remover todas as medidas e agrupamentos que estiverem nos containers, se o themUnit escolhido era o que estava
        selecionado

        if(_mapperModel.selectedThemeUnit == OntologyManager.getThemeUnit(event.URI)){
            contMeasures.removeAllChildren();
            contGrouping.removeAllChildren();
            contGroupingUnits.removeAllChildren();
        }
    }
}

```



```

        OntologyManager.removeThemeUnit(event.URI);
        for (i=0; i<contThemeUnits.getChildren().length; i++){
            if(HBoxAnalyticalElement(contThemeUnits.getChildren()[i]).URI == event.URI){
                contThemeUnits.removeChildAt(i);
            }
        }
    }else if(event.elementType == ElementTypes.MEASURE){
        OntologyManager.removeMeasureUnit(event.parentURI, event.URI);
        for (i=0; i< contMeasures.getChildren().length; i++){
            if(HBoxAnalyticalElement(contMeasures.getChildren()[i]).URI == event.URI){
                contMeasures.removeChildAt(i);
            }
        }
    }else if(event.elementType == ElementTypes.GROUPING){
        if(_mapperModel.selectedGrouping == OntologyManager.getGrouping(event.parentURI, event.URI)){
            contGroupingUnits.removeAllChildren();
        }
        OntologyManager.removeGrouping(event.parentURI, event.URI);
        for (i=0; i< contGrouping.getChildren().length; i++){
            if(HBoxAnalyticalElement(contGrouping.getChildren()[i]).URI == event.URI){
                contGrouping.removeChildAt(i);
            }
        }
    }else if(event.elementType == ElementTypes.GROUPING_UNIT){
        OntologyManager.removeGroupingUnit(event.parentURI, event.URI);
        for (i=0; i< contGroupingUnits.getChildren().length; i++){
            if(HBoxAnalyticalElement(contGroupingUnits.getChildren()[i]).URI == event.URI){
                contGroupingUnits.removeChildAt(i);
            }
        }
    }else if(event.elementType == ElementTypes.DETAIL){
        OntologyManager.removeThemeUnitDetail(event.parentURI, event.URI);
        for (i=0; i< contThemeUnitDetails.getChildren().length; i++){
            if(HBoxAnalyticalElement(contThemeUnitDetails.getChildren()[i]).URI == event.URI){
                contThemeUnitDetails.removeChildAt(i);
            }
        }
    }
}

}

//THEMES////////////////////////////////////
private function cboThemesLabelFunction(item:Object):String {
    return item.labelTheme;
}

private function btnNewThemeClickHandler(): void{
    var panelNewTheme : ConfigTheme = new ConfigTheme();
    panelNewTheme.addEventListener(ConfigThemeEvent.CONFIRM, configNewThemeBtnOKClickHandler);
    panelNewTheme.addEventListener(ConfigThemeEvent.CANCEL, configThemeBtnCancelClickHandler);
    PopupManager.addPopUp(panelNewTheme, DisplayObject(Application.application), true);
    PopupManager.centerPopUp(panelNewTheme);
}

private function btnRenameThemeClickHandler(): void{
    var panelNewTheme : ConfigTheme = new ConfigTheme();
    panelNewTheme.themeLabel = br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(_mapperModel.selectedTheme.labels.values[0]).label;
    panelNewTheme.addEventListener(ConfigThemeEvent.CONFIRM, configThemeBtnOKClickHandler);
    panelNewTheme.addEventListener(ConfigThemeEvent.CANCEL, configThemeBtnCancelClickHandler);
    PopupManager.addPopUp(panelNewTheme, DisplayObject(Application.application), true);
    PopupManager.centerPopUp(panelNewTheme);
}

private function configNewThemeBtnOKClickHandler(event: ConfigThemeEvent): void{
    PopupManager.removePopUp(FlexDisplayObject(event.target));

    var analyticalElementURI: URI_AS = OntologyManager.createURI("ano");
    var labelOnto: br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label = OntologyManager.createLabel(event.themeName);

    var theme: Theme = new Theme();
    theme.URI = analyticalElementURI;
    theme.labels.put(labelOnto.URI, labelOnto);

    OntologyManager.themes.push(theme);

    _mapperModel.selectedTheme = theme;

    //adiciona no provider do combo
    var themeObject: Object = new Object;
    themeObject.uriTheme = theme.URI;
    themeObject.labelTheme = event.themeName;
    _mapperModel.cboThemesProvider.addItem(themeObject);
    cboThemes.selectedItem = themeObject;

    contThemeUnits.removeAllChildren();
    contGrouping.removeAllChildren();
    contGroupingUnits.removeAllChildren();
    contThemeUnitDetails.removeAllChildren();
    contMeasures.removeAllChildren();
}

private function configThemeBtnOKClickHandler(event: ConfigThemeEvent): void{
    PopupManager.removePopUp(FlexDisplayObject(event.target));
    br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label(_mapperModel.selectedTheme.labels.values[0]).label = event.themeName;
}

```

```

        cboThemes.selectedItem.labelTheme = event.themeName;
        _mapperModel.cboThemesProvider.refresh();
    }

    private function configThemeBtnCancelClickHandler(event: Event): void{
        PopUpManager.removePopUp(IFlexDisplayObject(event.target));
    }

    private function deleteThemeClickHandler(): void{
        Alert.show("Essa operação irá remover o tema e todas as instâncias ligadas a ele. Deseja prosseguir?",
            "Remover o tema selecionado", Alert.OK | Alert.CANCEL, this, alertListener, null, Alert.CANCEL);
    }

    private function alertListener(eventObj: CloseEvent):void {
        if (eventObj.detail==Alert.OK) {
            OntologyManager.removeTheme(URI_AS(cboThemes.selectedItem.uriTheme));
            _mapperModel.cboThemesProvider.removeItemAt(_mapperModel.cboThemesProvider.getItemIndex(cboThemes.selectedItem));
            _mapperModel.cboThemesProvider.refresh();
            if(_mapperModel.cboThemesProvider.length>0){
                cboThemes.selectedIndex = 0;
                _mapperModel.selectedTheme = OntologyManager.getTheme(cboThemes.selectedItem.uriTheme);

                loadThemeUnits(_mapperModel.selectedTheme);
            }else{
                contThemeUnits.removeAllChildren();
                contMeasures.removeAllChildren();
                contGrouping.removeAllChildren();
                contGroupingUnits.removeAllChildren();
            }
        }
    }

    private function cboThemesChangeHandler(event: Event): void{
        var theme: Theme = Theme(OntologyManager.getTheme(URI_AS(ComboBox(event.target).selectedItem.uriTheme)));
        if(theme != _mapperModel.selectedTheme){
            _mapperModel.selectedTheme = theme;
            loadThemeUnits(_mapperModel.selectedTheme);
        }
    }

    //THEME UNITS////////////////////////////////////
    private function btnNewThemeUnitClickHandler(): void{
        if(_mapperModel.selectedTheme){
            var hboxAnalyticalElement: HBoxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.THEME_UNIT, _mapperModel.selectedTheme.URI);
            contThemeUnits.addChild(hboxAnalyticalElement);
        }else{
            Alert.show("Selecione um Tema antes", "Atenção!");
        }
    }

    //GROUPINGS////////////////////////////////////
    private function btnNewGroupingClickHandler(): void{
        if(_mapperModel.selectedThemeUnit){
            var hboxAnalyticalElement: HBoxAnalyticalElement = new HBoxAnalyticalElement(ElementTypes.GROUPING, _mapperModel.selectedThemeUnit.URI);
            contGrouping.addChild(hboxAnalyticalElement);
        }else{
            Alert.show("Selecione uma Unidade de Tema antes", "Atenção!");
        }
    }

]]</mx:Script>

<mx:Button label="Gerar Elementos da Ontologia Analítica" id="btnCreateANOInstances" click="btnCreateAllInstancesClickHandler()" tooltip="Instanciar Elementos
da Ontologia de Análise" width="286" height="40" icon="@Embed(source='/assets/mapping.gif') left="10" top="10"/>
<view:BtnLoadOntology right="68"/>
<view:BtnSaveOntology right="10"/>
<mx:HRule y="60" left="10" right="10"/>

<mx:HBox top="70" right="0" left="10" bottom="10">
    <mx:Canvas width="40%" height="100%">
        <view:TreeISO id="treeISO" dataProvider="{_mapperModel.dataSourcesTreeData}" top="26" bottom="10" cornerRadius="10"
right="10" left="0" dragEnabled="true" allowMultipleSelection="true" dropEnabled="false" dragMoveEnabled="false"/>
        <mx:Label text="Elementos da Ontologia de Fontes de Informação" fontWeight="bold" left="4" y="7"/>
    </mx:Canvas>
    <mx:Canvas width="60%" height="100%">
        <mx:Label y="7" text="Elementos da Ontologia Analítica" fontWeight="bold" left="14"/>
    </mx:Canvas>
    <mx:Canvas id="canvasWithBorderAnalyticalElements" borderStyle="solid" cornerRadius="10" left="10" right="10" bottom="10"
top="26" verticalScrollPolicy="off" horizontalScrollPolicy="off">
        <mx:Canvas id="canvasAnalyticalSuperior" width="100%" top="0" height="450">
            <mx:ComboBox y="23" id="cboThemes" left="10" dataProvider="{_mapperModel.cboThemesProvider}"
labelFunction="cboThemesLabelFunction" right="70" change="cboThemesChangeHandler(event)"/>
            <mx:Label y="5" text="Temas" fontWeight="bold" left="14"/>
            <mx:Button width="16" id="btnNewTheme" right="49" click="btnNewThemeClickHandler()"
icon="@Embed(source='/assets/new2.gif') y="27" tooltip="Adicionar tema" height="16"/>
            <mx:Button width="16" id="btnRenameTheme" right="29" click="btnRenameThemeClickHandler()"
icon="@Embed(source='/assets/rename1.gif') y="27" tooltip="Renomear tema" height="16"/>
            <mx:Button width="16" id="btnDeleteTheme" width="16" right="10"
icon="@Embed(source='/assets/remove.gif') click="deleteThemeClickHandler()" tooltip="Excluir tema" height="16"/>
        </mx:Canvas>
    </mx:Canvas>
</mx:HBox>
<mx:HRule y="64" height="1" right="10" left="10"/>

```

```

id="btnDefineDetails" click="btnDefineDetailsClickHandler()" tooltip="Definir detalhamento da Unidade de Tema" y="67" right="10"/>
<mx:Button width="18" height="18" icon="@Embed(source=/assets/details2.gif)" cornerRadius="2"
cornerRadius="2" id="btnNewThemeUnit" click="btnNewThemeUnitClickHandler()" tooltip="Adicionar unidade de tema"/>
<mx:Button y="73" text="Unidades de Tema" fontWeight="bold" left="14"/>
<mx:Button y="73" width="16" x="126" height="18" icon="@Embed(source=/assets/new.gif)"
cornerRadius="2" id="btnNewThemeUnit" click="btnNewThemeUnitClickHandler()" tooltip="Adicionar unidade de tema"/>
<mx:Canvas y="93" height="100" right="10" left="10" borderStyle="solid" cornerRadius="10">
<mx:VBox id="contThemeUnits" right="20" left="10" top="10" bottom="10"
dragEnter="contThemeUnitDragEnterHandler(event)" dragExit="contThemeUnitDragExitHandler(event)" dragOver="contThemeUnitDragOverHandler(event)"
dragStart="contThemeUnitDragStartHandler(event)" dragComplete="contThemeUnitDragCompleteHandler(event)" dragDrop="contThemeUnitDragDropHandler(event)"
backgroundColor="#FFFFFF" backgroundAlpha="0.0"/>
</mx:Canvas>

verticalScrollPolicy="off" horizontalScrollPolicy="off">
<mx:Canvas id="canvasMeasures" visible="true" width="100%" top="215" height="235"
<mx:HRule top="2" height="1" right="10" left="10"/>
<mx:Label y="8" text="Medidas" fontWeight="bold" left="14"/>
<mx:Label x="64" y="8" text="(Unidade de Tema Seleccionada)" fontWeight="bold"
id="lblSelectedThemeUnit"/>
<mx:Canvas id="canvasWithBorderMeasures" height="200" borderStyle="solid" left="10"
right="10" cornerRadius="10" top="25">
<mx:VBox id="contMeasures" right="20" left="10" top="10" bottom="10"
backgroundAlpha="0.0" dragEnter="contMeasuresDragEnterHandler(event)" dragOver="contMeasuresDragOverHandler(event)"
dragDrop="contMeasuresDragDropHandler(event)"/>
</mx:Canvas>
</mx:Canvas>

verticalScrollPolicy="off" horizontalScrollPolicy="off">
<mx:Canvas id="canvasThemeUnitDetails" visible="false" width="100%" top="215" height="235"
<mx:HRule top="2" height="1" right="10" left="10"/>
<mx:Label y="8" text="Detalhamentos" fontWeight="bold" left="14"/>
<mx:Label x="106" y="8" text="(Unidade de Tema Seleccionada)" fontWeight="bold"
id="lblSelectedThemeUnit2"/>
<mx:Canvas id="canvasWithBorderThemeUnitDetails" height="200" borderStyle="solid"
left="10" right="10" cornerRadius="10" top="25">
<mx:VBox id="contThemeUnitDetails" right="20" left="10" top="10"
bottom="10" backgroundAlpha="0.0" dragEnter="contThemeUnitDetailsDragEnterHandler(event)"
dragOver="contThemeUnitDetailsDragOverHandler(event)" dragDrop="contThemeUnitDetailsDragDropHandler(event)"/>
</mx:Canvas>
</mx:Canvas>

<mx:Canvas id="canvasGroupingsConfig" width="100%" bottom="0" top="450">
<mx:HRule top="2" height="1" right="10" left="10"/>
<mx:Button width="16" height="18" icon="@Embed(source=/assets/define_hierarchy.gif)"
cornerRadius="2" id="btnDefineHierarchy" click="btnDefineHierarchyClickHandler()" tooltip="Definir hierarquia das unidades de agrupamento" right="10" top="5"/>
<mx:HBox id="hboxGroupings" top="25" width="100%" height="100%">
<mx:Canvas id="canvasGroupings" width="100%" height="100%" verticalScrollPolicy="off">
<mx:Label text="Agrupamentos/ Filtros" fontWeight="bold" y="2" left="14"/>
<mx:Button y="1" width="16" x="152" height="18"
icon="@Embed(source=/assets/new.gif)" cornerRadius="2" id="btnNewGrouping" click="btnNewGroupingClickHandler()" tooltip="Adicionar agrupamento/ filtro"/>
<mx:Canvas borderStyle="solid" left="10" cornerRadius="10" bottom="6"
top="20" right="5">
<mx:VBox id="contGrouping" right="20" left="10" top="10"
bottom="10" backgroundAlpha="0.0" backgroundColor="#FFFFFF" dragEnter="contGroupingDragEnterHandler(event)" dragOver="contGroupingDragOverHandler(event)"
dragDrop="contGroupingDragDropHandler(event)"/>
</mx:Canvas>
</mx:Canvas>
</mx:Canvas>

verticalScrollPolicy="off" horizontalScrollPolicy="off">
<mx:Canvas id="canvasGroupingUnits" width="100%" height="100%"
<mx:Label text="Unidades de Agrupamento" fontWeight="bold" left="4"
y="2"/>
<mx:Canvas borderStyle="solid" cornerRadius="10" bottom="6" top="20"
left="0" right="5">
<mx:VBox id="contGroupingUnits" right="20" left="10"
top="10" bottom="10" backgroundAlpha="0.0" backgroundColor="#FFFFFF" dragEnter="contGroupingUnitsDragEnterHandler(event)"
dragOver="contGroupingUnitsDragOverHandler(event)" dragDrop="contGroupingUnitsDragDropHandler(event)"/>
</mx:Canvas>
</mx:Canvas>
</mx:HBox>

visible="false"> <!--height="215"-->
<mx:Canvas id="canvasDefineHierarchy" height="0" top="0" left="10" right="10" bottom="10"
y="2" left="10"/>
<mx:Label text="Definição da Hierarquia das Unidades de Agrupamento" fontWeight="bold"
<mx:HBox width="100%" height="100%" borderStyle="solid" cornerRadius="10" top="20">
<mx:Canvas width="100%" height="100%">
<mx:Label x="44" y="89" text="Unidade de Agrupamento"
Selecionada:" right="0"/>
<mx:Label x="85" y="157" text="Unidade de Agrupamento"
Filha:" right="0"/>
<mx:Label x="94" y="18" text="Unidade de Agrupamento Pai:"
right="0"/>
</mx:Canvas>
<mx:Canvas width="100%" height="100%">

```

```

        <mx:Canvas id="contGroupingUnitHierarquicalParent"
dragEnter="contGroupingUnitHierarchyDragEnterHandler(event)" dragOver="contGroupingUnitHierarchyDragOverHandler(event)"
dragDrop="contGroupingUnitHierarchyDragDropHandler(event)" y="6" height="40" borderStyle="solid" cornerRadius="10" right="6" left="0">
        <mx:Label y="10" horizontalCenter="0"
id="lblGroupingUnitHierarquicalParent" fontWeight="bold" fontSize="12"/>
    </mx:Canvas>
    <mx:Canvas id="contGroupingUnitSelected" y="77"
height="40" borderStyle="solid" cornerRadius="10" right="6" left="0">
        <mx:Label y="10" horizontalCenter="0"
id="lblGroupingUnitSelected" fontWeight="bold" fontSize="12"/>
    </mx:Canvas>
    <mx:Canvas id="contGroupingUnitHierarquicalChild"
dragEnter="contGroupingUnitHierarchyDragEnterHandler(event)" dragOver="contGroupingUnitHierarchyDragOverHandler(event)"
dragDrop="contGroupingUnitHierarchyDragDropHandler(event)" y="146" height="40" borderStyle="solid" cornerRadius="10" right="6" left="0">
        <mx:Label y="10" horizontalCenter="0"
id="lblGroupingUnitHierarquicalChild" fontWeight="bold" fontSize="12"/>
    </mx:Canvas>
    <mx:Image y="50" source="assets/down_arrow.gif"
horizontalCenter="-3"/>
    <mx:Image y="121" source="assets/down_arrow.gif"
horizontalCenter="-3"/>
</mx:Canvas>
</mx:HBox>
</mx:Canvas>
</mx:Canvas>
</mx:Canvas>
</mx:HBox>
</mx:Canvas>

```

TreelSO.as

```

package br.ufsc.inf.feco.mapper.view{

import mx.controls.Tree;

/**
 * Classe que representa a árvore que mostra o esquema de banco de dados selecionado
 * (ontologia de fontes de informação)
 */
public class TreelSO extends Tree{

    public function TreelSO(){
        super();
        labelFunction = treelLabelFunction;
    }

    private function treelLabelFunction(item: Object): String{
var xmlSchemaltem: XML = XML(item);

if(xmlSchemaltem.localName() == "table"){
return xmlSchemaltem.@source;
}else if (xmlSchemaltem.localName() == "attribute"){
return xmlSchemaltem.@source;
}
return "";
}
}
}

```

BtnLoadOntology.as

```

package br.ufsc.inf.feco.mapper.view{
import br.ufsc.inf.feco.mapper.events.GetOntologyInstancesEvent;

import flash.events.MouseEvent;

import mx.controls.Button;

public class BtnLoadOntology extends Button{

[Bindable]
[Embed(source="/assets/load.gif")]
public var iconSave:Class;

    public function BtnLoadOntology(){
        super();
        width = 56;
        height = 55;
        toolTip = "Carregar ontologia do arquivo owl";
        setStyle("icon", iconSave);
        addEventListener(MouseEvent.CLICK, btnLoadOntologyClickHandler);
    }

    private function btnLoadOntologyClickHandler(event: MouseEvent): void{
dispatchEvent(new GetOntologyInstancesEvent(GetOntologyInstancesEvent.LOAD_ALL, true));
    }
}
}

```

```
}
```

BtnSaveOntology.as

```
package br.ufsc.inf.feco.mapper.view{

    import br.ufsc.inf.feco.mapper.view.popups.SaveBIOntology;

    import flash.display.DisplayObject;
    import flash.events.Event;
    import flash.events.MouseEvent;

    import mx.controls.Alert;
    import mx.controls.Button;
    import mx.core.Application;
    import mx.managers.PopUpManager;

    public class BtnSaveOntology extends Button{

        [Bindable]
        [Embed(source="/assets/save.gif")]
        public var iconSave:Class;

        public function BtnSaveOntology(){
            super();
            width = 56;
            height = 55;
            tooltip = "Salvar ontologia em arquivo owl";
            setStyle("icon", iconSave);
            addEventListener(MouseEvent.CLICK, btnSaveOntologyClickHandler);
        }

        private function btnSaveOntologyClickHandler(event: MouseEvent): void{
            var panelSaveOntology : SaveBIOntology = new SaveBIOntology();
            panelSaveOntology.addEventListener(Event.COMPLETE, saveOntologyCompleted);
            PopUpManager.addPopUp(panelSaveOntology, DisplayObject(Application.application), true);
            PopUpManager.centerPopUp(panelSaveOntology);
        }

        public function saveOntologyCompleted(event: Event): void {
            PopUpManager.removePopUp(mx.core.IFlexDisplayObject(event.target));
            Alert.show("Ontologia salva com sucesso!", "");
        }
    }
}
```

HBoxAnalyticalElement.as

```
package br.ufsc.inf.feco.mapper.view{

    import br.ufsc.inf.feco.mapper.constants.ElementTypes;
    import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementClick;
    import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementRemove;
    import br.ufsc.inf.feco.mapper.events.HBoxAnalyticalElementUpdateLabel;
    import br.ufsc.inf.feco.mapper.view.skins.ToggleLinkButtonSkin;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;

    import flash.events.FocusEvent;
    import flash.events.KeyboardEvent;
    import flash.events.MouseEvent;
    import flash.ui.Keyboard;

    import mx.containers.HBox;
    import mx.controls.Button;
    import mx.controls.LinkButton;
    import mx.controls.TextInput;

    public class HBoxAnalyticalElement extends HBox{

        private var _elementType: ElementTypes;
        private var _parentURI: URI_AS;
        private var _URI: URI_AS;

        private var _linkButton: LinkButton;
        private var _txtRotulo: TextInput;
        private var _isEditingTxtRotulo: Boolean = false;

        private var _btnRename: Button;
        private var _btnRemove: Button;

        [Bindable]
        [Embed(source='assets/rename2.gif')]
        private var iconRename: Class;

        [Bindable]
        [Embed(source='assets/remove.gif')]
        private var iconRemove: Class;
    }
}
```

```

public function HBoxAnalyticalElement(elementType: ElementTypes, parentURI: URI_AS, uri: URI_AS = null, elementLabel: String = null){
    super();
    _elementType = elementType;
    _parentURI = parentURI;

    if(uri){
        _URI = uri;
    }

    _linkButton = new LinkButton;
    if(elementLabel){
        _linkButton.label = elementLabel;
    }
    _linkButton.setStyle("skin", ToggleLinkButtonSkin);
    _linkButton.addEventListener(MouseEvent.CLICK, linkButtonClickHandler);
    addChild(_linkButton);

    var btnToolTip: String;
    if(elementType == ElementTypes.THEME_UNIT){
        btnToolTip = "Unidade de Tema";
    }else if(elementType == ElementTypes.MEASURE){
        btnToolTip = "Medida";
    }else if(elementType == ElementTypes.GROUPING){
        btnToolTip = "Agrupamento";
    }else if(elementType == ElementTypes.GROUPING_UNIT){
        btnToolTip = "Unidade de Agrupamento";
    }else if(elementType == ElementTypes.DETAIL){
        btnToolTip = "Detalhamento";
    }else if(elementType == ElementTypes.MERGE){
        btnToolTip = "junção a tabela";
    }else{
        btnToolTip = "Elemento";
    }

    if(elementType != ElementTypes.MERGE){
        _btnRename = new Button;
        _btnRename.width = 16;
        _btnRename.height = 16;
        _btnRename.toolTip = "Renomear " + btnToolTip;
        _btnRename.setStyle("icon", iconRename);
        _btnRename.addEventListener(MouseEvent.CLICK, btnRenameClicked);
        addChild(_btnRename);
    }

    _btnRemove = new Button;
    _btnRemove.width = 16;
    _btnRemove.height = 16;
    _btnRemove.toolTip = "Remover " + btnToolTip;
    _btnRemove.setStyle("icon", iconRemove);
    _btnRemove.addEventListener(MouseEvent.CLICK, btnRemoveClicked);
    addChild(_btnRemove);

    if(!uri){
        showTextInput();
    }
}

private function linkButtonClickHandler(event: MouseEvent): void{
    dispatchEvent(new HBoxAnalyticalElementClick(HBoxAnalyticalElementClick.LINKBUTTONCLICK, _elementType, _URI, _parentURI));
    _linkButton.selected = true;
}

public function selectLinkButton(select: Boolean): void{
    _linkButton.selected = select;
}

private function btnRemoveClicked(event: MouseEvent): void{
    dispatchEvent(new HBoxAnalyticalElementRemove(HBoxAnalyticalElementRemove.REMOVE, _elementType, _URI, _parentURI));
}

private function btnRenameClicked(event: MouseEvent): void{
    if(_isEditingTxtRotulo){
        //comitar o que estava sendo editado
        updateLabel();
    }
    showTextInput();
    _txtRotulo.setFocus();
}

private function showTextInput(): void{
    _isEditingTxtRotulo = true;
    _txtRotulo = new TextInput();
    _txtRotulo.width = _linkButton.width; //_linkButton.maxWidth;
    _linkButton.width = 0;
    _txtRotulo.text = _linkButton.label;
    _txtRotulo.addEventListener(KeyboardEvent.KEY_DOWN, keyDownWizardHandler);
    _txtRotulo.addEventListener(FocusEvent.FOCUS_OUT, focusOutWizardHandler);
    _linkButton.visible = false;
    addChildAt(_txtRotulo,0);
}

```

```

private function keyDownWizardHandler(event: KeyboardEvent): void{
    if(event.keyCode == Keyboard.ENTER){
        updateLabel();
    }
}

private function focusOutWizardHandler(event: FocusEvent): void{
    updateLabel();
}

private function updateLabel(): void{
    if(!_isEditingTxtRotulo){
        _linkButton.label = _txtRotulo.text;
        dispatchEvent(new HBoxAnalyticalElementUpdateLabel(HBoxAnalyticalElementUpdateLabel.UPDATE_LABEL,
        _elementType, _URI, _parentURI, _txtRotulo.text));
        removeChildAt(0);
        _linkButton.visible = true;
        _isEditingTxtRotulo = false;
    }
}

protected override function updateDisplayList(unscaledWidth:Number, unscaledHeight:Number):void{
    super.updateDisplayList(unscaledWidth, unscaledHeight);
    if(!_isEditingTxtRotulo){
        _linkButton.width = _linkButton.measuredWidth;
    }else{
        _txtRotulo.width = 150;
        _txtRotulo.setFocus();
    }
}

public function set URI(URI: URI_AS): void{
    _URI = URI;
}

public function get URI(): URI_AS{
    return _URI;
}

public function get parentURI(): URI_AS{
    return _parentURI;
}

public function setLabel([label: String]: void{
    _linkButton.label = label;
}

public function getLabel(): String{
    return _linkButton.label;
}
}
}

```

HBoxMergeDetail.as

```

package br.ufsc.inf.feco.mapper.view{

import br.ufsc.inf.feco.mapper.events.HBoxMergeDetailModification;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBAttribute;

import flash.events.MouseEvent;

import mx.containers.Canvas;
import mx.containers.HBox;
import mx.controls.Button;
import mx.controls.Image;
import mx.controls.Label;
import mx.controls.Tree;
import mx.core.DragSource;
import mx.core.UIComponent;
import mx.events.DragEvent;
import mx.managers.DragManager;

public class HBoxMergeDetail extends HBox{

    private var _mergeURI: URI_AS;
    private var _binaryExpressionURI: URI_AS;
    private var _dbAttribute1URI: URI_AS;
    private var _dbAttribute2URI: URI_AS;
    private var _lblAttribute1: Label;
    private var _lblAttribute2: Label;

    private var _btnRemove: Button;

    [Bindable]
    [Embed(source='assets/remove.gif')]
    private var iconRemove: Class;

    [Bindable]
    [Embed(source='assets/equals.gif')]

```

```

private var iconEquals: Class;

public function HBoxMergeDetail(mergeURI: URI_AS, binaryExpressionURI: URI_AS, dbAttribute1URI: URI_AS, source: String){
    super();
    height = 25;
    percentWidth=100;
    _mergeURI = mergeURI;
    _binaryExpressionURI = binaryExpressionURI;
    _dbAttribute1URI = dbAttribute1URI;

    var canvasdbAttribute1: Canvas = new Canvas;
    canvasdbAttribute1.percentWidth = 100;
    canvasdbAttribute1.percentHeight = 100;
    canvasdbAttribute1.setStyle("borderStyle","solid");
    canvasdbAttribute1.setStyle("cornerRadius",10);
    _lblAttribute1 = new Label;
    _lblAttribute1.setStyle("fontWeight","bold");
    _lblAttribute1.setStyle("verticalCenter",0);
    _lblAttribute1.setStyle("horizontalCenter",0);
    _lblAttribute1.text = source;
    canvasdbAttribute1.addChild(_lblAttribute1);
    addChild(canvasdbAttribute1);

    var imgEquals: Image = new Image;
    imgEquals.source = iconEquals;
    addChild(imgEquals);

    var canvasdbAttribute2: Canvas = new Canvas;
    canvasdbAttribute2.percentWidth = 100;
    canvasdbAttribute2.percentHeight = 100;
    canvasdbAttribute2.setStyle("borderStyle","solid");
    canvasdbAttribute2.setStyle("cornerRadius",10);
    canvasdbAttribute2.setStyle("backgroundColor",0xFFFFFFFF);
    canvasdbAttribute2.setStyle("backgroundAlpha",0);
    _lblAttribute2 = new Label;
    _lblAttribute2.setStyle("fontWeight","bold");
    _lblAttribute2.setStyle("verticalCenter",0);
    _lblAttribute2.setStyle("horizontalCenter",0);
    canvasdbAttribute2.addChild(_lblAttribute2);
    canvasdbAttribute2.addEventListener(DragEvent.DRAG_ENTER, canvasDBAttribute2DragEnterHandler);
    canvasdbAttribute2.addEventListener(DragEvent.DRAG_OVER, canvasDBAttribute2DragOverHandler);
    canvasdbAttribute2.addEventListener(DragEvent.DRAG_DROP, canvasDBAttribute2DragDropHandler);
    addChild(canvasdbAttribute2);

    _btnRemove = new Button;
    _btnRemove.width = 16;
    _btnRemove.height = 16;
    _btnRemove.toolTip = "Remover";
    _btnRemove.setStyle("icon", iconRemove);
    _btnRemove.addEventListener(MouseEvent.CLICK, btnRemoveClicked);
    addChild(_btnRemove);
}

private function btnRemoveClicked(event: MouseEvent): void{
    dispatchEvent(new HBoxMergeDetailModification(HBoxMergeDetailModification.REMOVE, null, null, _mergeURI,
    _binaryExpressionURI, true));
}

private function canvasDBAttribute2DragEnterHandler(event: DragEvent): void{
    if(event.dragInitiator is Tree){
        var ds: DragSource = event.dragSource;
        if(ds.hasFormat("treeltems")) return;
        var items: Array = ds.dataForFormat("treeltems") as Array;
        for(var i:Number=0; i < items.length; i++){
            var item:XML = XML(items[i]);
            if((item.localName() != "attribute"))
                return;
        }
        DragManager.acceptDragDrop(UIComponent(event.currentTarget));
    }
}

private function canvasDBAttribute2DragOverHandler(event: DragEvent): void{
    if(event.dragInitiator is Tree){
        DragManager.showFeedback(DragManager.COPY);
    }
}

private function canvasDBAttribute2DragDropHandler(event: DragEvent): void{
    var ds: DragSource = event.dragSource;
    var dropTarget:Canvas = Canvas(event.currentTarget);
    var arr:Array;

    if(ds.hasFormat("items")) {
        arr = ds.dataForFormat("items") as Array;
    }else if(ds.hasFormat("treeltems")) {
        arr = ds.dataForFormat("treeltems") as Array;
    }

    var node:XML = XML(arr[0]);
    var attribute: DBAttribute;

```



```

        if(node.localName() == "attribute"){
            dispatchEvent(new HBoxMergeDetailModification(HBoxMergeDetailModification.UPDATE_SECOND_ATTRIBUTE,
node.@uriName, node.parent().@uriName, _mergeURI, binaryExpressionURI, true));
            _lblAttribute2.text = node.@source; //attribute.source;
        }
    }

    public function set dbAttribute1URI(dbAttribute1URI: URI_AS): void{
        _dbAttribute1URI = dbAttribute1URI;
    }

    public function get dbAttribute1URI(): URI_AS{
        return _dbAttribute1URI;
    }

    public function set binaryExpressionURI(binaryExpressionURI: URI_AS): void{
        _binaryExpressionURI = binaryExpressionURI;
    }

    public function get binaryExpressionURI(): URI_AS{
        return _binaryExpressionURI;
    }

    public function set mergeURI(mergeURI: URI_AS): void{
        _mergeURI = mergeURI;
    }

    public function get mergeURI(): URI_AS{
        return _mergeURI;
    }

    public function set dbAttribute2URI(dbAttribute2URI: URI_AS): void{
        _dbAttribute2URI = dbAttribute2URI;
    }

    public function get dbAttribute2URI(): URI_AS{
        return _dbAttribute2URI;
    }

    public function set lblAttribute1Text(lblAttribute1Text: String): void{
        _lblAttribute1.text = lblAttribute1Text;
    }

    public function get lblAttribute1Text(): String{
        return _lblAttribute1.text;
    }

    public function set lblAttribute2Text(lblAttribute2Text: String): void{
        _lblAttribute2.text = lblAttribute2Text;
    }

    public function get lblAttribute2Text(): String{
        return _lblAttribute2.text;
    }
}
}
}

```

ToggleLinkButtonSkin.as

```

package br.ufsc.inf.feco.mapper.view.skins{

import mx.skins.halo.LinkButtonSkin;

/**
 * Classe utilizada como skin para alguns componentes do tipo linkbutton
 */
public class ToggleLinkButtonSkin extends LinkButtonSkin{

    public function ToggleLinkButtonSkin(){
        super();
    }

    override protected function updateDisplayList(w:Number, h:Number):void {
        super.updateDisplayList(w, h);

        var cornerRadius:Number = getStyle("cornerRadius");
        var rollOverColor:uint = getStyle("rollOverColor");
        var selectionColor:uint = getStyle("selectionColor");

        graphics.clear();

        switch (name) {
            case "upSkin":
                // Draw invisible shape so we have a hit area.
                drawRoundRect(0, 0, w, h, cornerRadius, 0, 0);
                break;

            case "selectedUpSkin":
            case "selectedOverSkin":
            case "overSkin":
                drawRoundRect(0, 0, w, h, cornerRadius, rollOverColor, 1);
        }
    }
}

```

```

        break;

        case "selectedDownSkin":
        case "downSkin":
            drawRoundRect(0, 0, w, h, cornerRadius, selectionColor, 1);
            break;

        case "selectedDisabledSkin":
        case "disabledSkin":
            // Draw invisible shape so we have a hit area.
            drawRoundRect(0, 0, w, h, cornerRadius, 0, 0);
            break;
    }
}
}
}

```

ConfigTheme.mxml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<!--Janela para inserção de um novo tema-->
```

```
<mx:Panel xmlns:mx="http://www.adobe.com/2006/mxml" width="350" height="250"
    title="Inserir Novo Tema" initialize="init()" creationComplete="afterComplete()">
```

```

    <mx:Script>
        <![CDATA[
            import br.ufsc.inf.feco.mapper.events.ConfigThemeEvent;

            [Bindable]
            private var _themeLabel: String = "";

            private function init(): void{
                this.addEventListener(KeyboardEvent.KEY_DOWN, keyPressedHandler);
            }

            private function keyPressedHandler(event: KeyboardEvent): void{
                if(event.keyCode == Keyboard.ENTER)
                    dispatchEvent(new ConfigThemeEvent(ConfigThemeEvent.CONFIRM, txtThemeName.text));
                else if(event.keyCode == Keyboard.ESCAPE)
                    dispatchEvent(new ConfigThemeEvent(ConfigThemeEvent.CANCEL));
            }

            private function afterComplete(): void{
                txtThemeName.setFocus();
            }

            public function get themeLabel(): String{
                return _themeLabel;
            }

            public function set themeLabel(themeLabel: String): void{
                _themeLabel = themeLabel;
            }

            private function btnOKClickHandler(): void{
                dispatchEvent(new ConfigThemeEvent(ConfigThemeEvent.CONFIRM, txtThemeName.text));
            }

            private function btnCancelClickHandler(): void{
                dispatchEvent(new ConfigThemeEvent(ConfigThemeEvent.CANCEL));
            }

        ]]>
    </mx:Script>

    <mx:Canvas width="100%" height="100%">
        <mx:Button label="Confirmar" id="btnOK" x="79" y="163" click="btnOKClickHandler()"/>
        <mx:Button label="Cancelar" id="btnCancel" x="170" y="163" click="btnCancelClickHandler()"/>
        <mx:TextInput y="67" id="txtThemeName" horizontalCenter="0" width="200" text="{themeLabel}"/>
        <mx:Label x="66" y="46" text="Nome do Tema:"/>
    </mx:Canvas>
</mx:Panel>

```

GenerateBIOntologyInstances.as

```

package br.ufsc.inf.feco.mapper.view.popups{

    import br.ufsc.inf.feco.mapper.business.MapperService;
    import br.ufsc.inf.feco.ontologymanager.OntologyManager;

    import mx.controls.ProgressBar;
    import mx.controls.ProgressBarMode;
    import mx.rpc.events.ResultEvent;
}

```

```

/**
 * Classe visual que mostra o progresso do download da ontologia de BI (gerada pelo servidor).
 */
public class GenerateBIOntologyInstances extends ProgressBar{

    public function GenerateBIOntologyInstances(tableFactPrefix: String, measurePrefix: String){
        super();

        var serviceLocator: MapperService = new MapperService(onComplete, null);

        label = "Aguarde, carregando instâncias da ontologia BI... ";

        width = 300;
        height = 300;
        mode = ProgressBarMode.MANUAL;
        indeterminate = true;

        serviceLocator.generateThemes(tableFactPrefix, measurePrefix);
    }

    /**
     * Evento disparado no final do download.
     */
    public function onComplete(event: ResultEvent): void {
        OntologyManager.themes = event.result as Array;
        setProgress(100, 100);
    }
}
}

```

GenerateISOInstances.as

```

package br.ufsc.inf.feco.mapper.view.popups{

    import br.ufsc.inf.feco.mapper.business.MapperService;
    import br.ufsc.inf.feco.ontologymanager.OntologyManager;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBCollection;

    import mx.controls.ProgressBar;
    import mx.controls.ProgressBarMode;
    import mx.rpc.events.ResultEvent;

    /**
     * Classe visual que mostra o progresso do download da ontologia de fontes-de-informação
     * (gerada pelo servidor).
     */
    public class GenerateISOInstances extends ProgressBar{

        public function GenerateISOInstances() {
            super();

            var serviceLocator: MapperService = new MapperService(onComplete, null);

            label = "Aguarde, carregando instâncias da ontologia ISO... ";

            width = 300;
            height = 300;
            mode = ProgressBarMode.MANUAL;
            indeterminate = true;

            serviceLocator.generateDBCollectionSchema();
        }

        /**
         * Evento disparado no final do download.
         */
        public function onComplete(event: ResultEvent): void {
            OntologyManager.dbCollectionSchema = event.result as DBCollection;
            setProgress(100, 100);
        }
    }
}

```

LoadAggregateFunctionInstances.as

```

package br.ufsc.inf.feco.mapper.view.popups{

    import mx.controls.ProgressBar;
    import br.ufsc.inf.feco.mapper.business.MapperService;
    import br.ufsc.inf.feco.ontologymanager.OntologyManager;

    import mx.controls.ProgressBarMode;
    import mx.rpc.events.ResultEvent;

```

```

/**
 * Classe visual que mostra o progresso do download das funções
 * de agregação definidas na ontologia (SUM, CONT..).
 *
 */
public class LoadAggregateFunctionInstances extends ProgressBar{

    public function LoadAggregateFunctionInstances(){
        super();

        var serviceLocator: MapperService = new MapperService(onComplete, null);

        label = "Aguarde, carregando funções... ";

        width = 300;
        height = 300;
        mode = ProgressBarMode.MANUAL;
        indeterminate = true;

        serviceLocator.getAggregateFunctions();
    }

    /**
     * Evento disparado no final do download.
     */
    public function onComplete(event: ResultEvent): void {
        OntologyManager.aggregateFunctions = event.result as Array;
        setProgress(100, 100);
    }
}

```

LoadBIOInstances.as

```

package br.ufsc.inf.feco.mapper.view.popups{

    import br.ufsc.inf.feco.mapper.business.MapperService;
    import br.ufsc.inf.feco.ontologymanager.OntologyManager;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBCollection;

    import mx.controls.ProgressBar;
    import mx.controls.ProgressBarMode;
    import mx.rpc.events.ResultEvent;

    /**
     * Classe visual que mostra o progresso do download da ontologia de BI.
     *
     */
    public class LoadBIOInstances extends ProgressBar{

        public function LoadBIOInstances() {
            super();

            var serviceLocator: MapperService = new MapperService(onComplete, null);

            label = "Aguarde, carregando instâncias da ontologia... ";

            width = 300;
            height = 300;
            mode = ProgressBarMode.MANUAL;
            indeterminate = true;

            serviceLocator.getThemes();
        }

        /**
         * Evento disparado no final do download.
         */
        public function onComplete(event: ResultEvent): void {
            OntologyManager.themes = event.result as Array;
            setProgress(100, 100);
        }
    }
}

```

LoadISOInstances.as

```

package br.ufsc.inf.feco.mapper.view.popups{

    import br.ufsc.inf.feco.mapper.business.MapperService;
    import br.ufsc.inf.feco.ontologymanager.OntologyManager;
    import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBCollection;

    import mx.controls.ProgressBar;
    import mx.controls.ProgressBarMode;

```

```

import mx.rpc.events.ResultEvent;

/**
 * Classe visual que mostra o progresso do download da ontologia de fontes de informação.
 */
public class LoadISOInstances extends ProgressBar{

    public function LoadISOInstances() {
        super();

        var serviceLocator: MapperService = new MapperService(onComplete, null);

        label = "Aguarde, carregando instâncias da ontologia ISO... ";

        width = 300;
        height = 300;
        mode = ProgressBarMode.MANUAL;
        indeterminate = true;

        serviceLocator.getDBCollectionSchema();
    }

    /**
     * Evento disparado no final do download.
     */
    public function onComplete(event: ResultEvent): void {
        OntologyManager.dbCollectionSchema = event.result as DBCollection;
        setProgress(100, 100);
    }
}

```

LoadLocalesInstances.as

```

package br.ufsc.inf.feco.mapper.view.popups{

import mx.controls.ProgressBar;
import br.ufsc.inf.feco.mapper.business.MapperService;
import br.ufsc.inf.feco.ontologymanager.OntologyManager;

import mx.controls.ProgressBarMode;
import mx.rpc.events.ResultEvent;

/**
 * Classe visual que mostra o progresso do download das
 * instâncias de locais da ontologia (utilizadas para internacionalização).
 */
public class LoadLocalesInstances extends ProgressBar{

    public function LoadLocalesInstances(){
        super();

        var serviceLocator: MapperService = new MapperService(onComplete, null);

        label = "Aguarde, carregando locais... ";

        width = 300;
        height = 300;
        mode = ProgressBarMode.MANUAL;
        indeterminate = true;

        serviceLocator.getLocales();
    }

    /**
     * Evento disparado no final do download.
     */
    public function onComplete(event: ResultEvent): void {
        OntologyManager.locales = event.result as Array;
        setProgress(100, 100);
    }
}
}

```

LoadMergesInstances.as

```

package br.ufsc.inf.feco.mapper.view.popups{

import br.ufsc.inf.feco.mapper.business.MapperService;
import br.ufsc.inf.feco.ontologymanager.OntologyManager;

import mx.controls.ProgressBar;
import mx.controls.ProgressBarMode;
import mx.rpc.events.ResultEvent;

```

```

/**
 * Classe visual que mostra o progresso do download das
 * instâncias de merge da ontologia de fonte-de-informação.
 *
 */
public class LoadMergesInstances extends ProgressBar{

    public function LoadMergesInstances() {
        super();

        var serviceLocator: MapperService = new MapperService(onComplete, null);

        label = "Aguarde, carregando junções detectadas... ";

        width = 300;
        height = 300;
        mode = ProgressBarMode.MANUAL;
        indeterminate = true;

        serviceLocator.getMerges();

    }

    /**
     * Evento disparado no final do download.
     */
    public function onComplete(event: ResultEvent): void {
        OntologyManager.merges = event.result as Array;
        setProgress(100, 100);

    }

}

```

PanelHeuristicsConfigSimple.mxml

```

<?xml version="1.0" encoding="utf-8"?>

<!--Janela para definição dos prefixos que serão utilizados para reconhecer as tabelas de fato e medidas no
processo de mapeamento da ontologia ISO com BI pelo servidor -->
<mx:Panel xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute" width="530" height="456" title="Configurar Heurísticas para Mapeamento ISO -&gt; ANO"
horizontalAlign="center">
    <mx:Script>
        <![CDATA[
            import br.ufsc.inf.feco.mapper.events.PanelHeuristicsConfigExecute;

            private function btnClickHandler():void{
                dispatchEvent(new PanelHeuristicsConfigExecute(PanelHeuristicsConfigExecute.CONFIRM, false, txtTablePrefix.text,
txtMeasurePrefix.text));
            }
        ]]>
    </mx:Script>

    <mx:Label y="260" height="20" text="Heurísticas para detecção de Agrupamentos e de Unidades de Agrupamentos" fontWeight="bold" horizontalCenter="0"/>
    <mx:Label y="9" height="20" text="Heurísticas para detecção de Tabelas de Fatos e de Medidas" fontWeight="bold" horizontalCenter="0"/>
    <mx:TextArea y="278" height="77" editable="false" right="10" left="10" cornerRadius="10">
        <mx:text>*Os Agrupamentos, que representarão inicialmente as tabelas de dimensão, serão detectados automaticamente através das junções dessas
tabelas com as tabelas de fato. As Unidades de Agrupamento serão representados pelos atributos da tabela que representa o agrupamento.</mx:text>
    </mx:TextArea>
    <mx:Button label="OK" id="btnOK" horizontalCenter="0" bottom="10" click="btnClickHandler()"/>
    <mx:Canvas y="26" height="215" right="10" left="10" cornerRadius="10" borderStyle="solid">
        <mx:RadioButtonGroup id="rdgFactTableDetectionType"/>
        <mx:RadioButtonGroup id="rdgMeasuresDetectionType"/>
        <mx:Canvas y="10" height="190" width="400" horizontalCenter="0">
            <mx:Label text="Detecção das Tabelas de Fato:" horizontalCenter="0"/>
            <mx:Label text="Detecção das Medidas (Fatos):" horizontalCenter="0" y="100"/>
            <mx:TextInput id="txtTablePrefix" x="219" y="27" width="163" text="fato"/>
            <mx:TextInput id="txtMeasurePrefix" x="219" y="126" width="163" text="TOT"/>
            <mx:Label x="62" y="29" text="Prefixo (FATO, FT, etc)"/>
            <mx:Label x="42" y="128" text="Prefixo (TOT, TOTAL , etc)"/>
        </mx:Canvas>
    </mx:Canvas>
</mx:Panel>

```

SaveBIOntology.as

```

package br.ufsc.inf.feco.mapper.view.popups{

    import br.ufsc.inf.feco.mapper.business.MapperService;

    import mx.controls.ProgressBar;
    import mx.controls.ProgressBarMode;
    import mx.rpc.events.ResultEvent;

    /**

```

```

* Classe visual que mostra o progresso do upload da
* ontologia de BI para o servidor.
*
*/
public class SaveBIOntology extends ProgressBar{

    public function SaveBIOntology(){
        super();

        var serviceLocator: MapperService = new MapperService(onComplete, null);

        label = "Aguarde, salvando a ontologia... ";

        width = 300;
        height = 300;
        mode = ProgressBarMode.MANUAL;
        indeterminate = true;

        serviceLocator.saveThemes();
    }

    /**
     * Evento disparado no final do upload.
     */
    public function onComplete(event: ResultEvent): void {
        trace("OK");
        //OntologyManager.themes = event.result as Array;
        setProgress(100, 100);
    }
}
}

```

Mapper_CommonsFlex

OntologyManager.as

```

package br.ufsc.inf.feco.ontologymanager{

import br.ufsc.inf.feco.ontologymanager.semantic.ontology.Namespace_AS;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.URI_AS;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.AggregateFunction;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.BinaryExpression;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.BinaryExpression$Type;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.Locale;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.UnaryExpression;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.AnalyticalElement;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.BIThemeUnit;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Detail;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Grouping;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.GroupingUnit;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.MeasureUnit;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.Theme;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.analytical.ThemeUnit;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBAttribute;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.DBCollection;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.source.Merge;
import br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label;

import mx.utils.UIDUtil;

import org.granite.collections.IMap;

/**
 * Representa o controlador do módulo de gerenciamento de ontologias do cliente
 *
 */
public class OntologyManager{

    //representa um esquema de banco de dados
    private static var _dbCollectionSchema: DBCollection = null;

    //armazena as junções utilizadas (ontologia de fontes de informação)
    private static var _merges: Array = null;

    //armazena os temas utilizados (ontologia de BI)
    private static var _themes: Array = null;

    //armazena as URIs dos elementos deletados (será necessário na persistência no servidor)
    private static var _deletedURIs: Array = new Array;

    //armazenas os locais da ontologia (internacionalização)
    private static var _locales: Array = null;

    //armazena as funções de agregação da ontologia (SUM, CONT, etc...)
    private static var _aggregateFunctions: Array = null;

```

```

/**
 * Cria uma instância da classe URI.
 *
 * @param nsPrefix: String - prefixo do namespace.
 *
 * @return URI_AS.
 */
public static function createURI(nsPrefix: String): URI_AS{
    var namespaceURI: Namespace_AS = new Namespace_AS();
    namespaceURI.ns = "http://www.stela.org.br/ekp#";
    namespaceURI.prefix = "ekp";

    var uri: URI_AS = new URI_AS;
    uri.name = "_" + UIDUtil.createUID();
    uri.ns = namespaceURI;
    uri.prefixedName = "ekp-" + uri.name;

    return uri;
}

public static function createLabel(strLabel: String): br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label{
    var labelOnto: br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label = new
br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label;
    labelOnto.locale = Locale[_locales[0]];
    labelOnto.label = strLabel;
    labelOnto.URI = createURI("vio");

    return labelOnto;
}

public static function getAggregateFunctionInstance(type: String): AggregateFunction{
    var i: uint;
    for(i=0; i<_aggregateFunctions.length; i++){
        if(AggregateFunction[_aggregateFunctions[i]].functionType == type){
            return AggregateFunction[_aggregateFunctions[i]];
        }
    }
    return null;
}

/**
 * Associa um rótulo a um Elemento analítico.
 *
 * @param element: AnalyticalElement - o Elemento analítico alvo.
 * @param label: String - o rótulo desejado.
 *
 * @see #createLabel(strLabel: String)
 */
private static function setAnalyticalElementLabel(element: AnalyticalElement, label: String): void{
    var labelOnto: br.ufsc.inf.feco.ontologymanager.semantic.ontology.application.visual.Label = OntologyManager.createLabel(label);
    element.labels.put(labelOnto.URI, labelOnto);
}

/**
 * Retorna a instância de um Tema, dada uma URI.
 *
 * @param themeURI: URI_AS - a URI do Tema desejado.
 *
 * @return Theme
 */
public static function getTheme(themeURI: URI_AS): Theme{
    var theme: Theme;
    for (var t: uint = 0; t < _themes.length; t++) {
        theme = _themes[t];
        if (theme.URI == themeURI)
            return theme;
    }
    return null;
}

/**
 * Retorna a instância de um Tema, dado seu rótulo.
 */
public static function getThemeByLabel(label: String): Theme {
    var theme: Theme;

    for (var t: uint = 0; t < _themes.length; t++) {
        theme = _themes[t];
        if ( (theme.labels.values[0] as Label).label == label)
            return theme;
    }

    return null;
}

/**
 * Remove a instância de um Tema, dada uma URI.
 */
public static function removeTheme(themeURI: URI_AS): Boolean{

```



```

        var theme: Theme;

        var t: uint;
        for (t = 0; t < _themes.length; t++) {
            theme = _themes[t];
            if (theme.URI == themeURI){
                _themes.splice(t,1);
                return true;
            }
        }
        return false;
    }
}

/**
 * Cria uma Uidade de Tema, dado um rótulo.
 */
public static function createBIThemeUnit(label: String): BIThemeUnit{
    var element: BIThemeUnit = new BIThemeUnit();
    var elementURI: URI_AS = OntologyManager.createURI("ano");
    element.URI = elementURI;
    setAnalyticalElementLabel(element, label);
    return element;
}

/**
 * Reorna uma Uidade de Tema, dada uma URI.
 */
public static function getThemeUnit(themeUnitURI: URI_AS): ThemeUnit {
    var theme: Theme;
    var themeUnit: ThemeUnit;
    for (var t: uint = 0; t < _themes.length; t++) {
        theme = _themes[t];
        for (var tu: uint = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            if (themeUnit.URI == themeUnitURI)
                return themeUnit;
        }
    }
    return null;
}

/**
 * Remove uma Uidade de Tema, dada uma URI.
 */
public static function removeThemeUnit(themeUnitURI: URI_AS): Boolean {
    var theme: Theme;
    var themeUnit: ThemeUnit;

    var t: uint;
    for (t = 0; t < _themes.length; t++) {
        theme = _themes[t];
        var tu: uint;
        for (tu = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            if (themeUnit.URI == themeUnitURI){
                theme.removeThemeUnit(themeUnitURI);
                return true;
            }
        }
    }
    return false;
}

/**
 * Cria uma instância de MeasureUnit (medida), dado um atributo do BD.
 */
public static function createMeasureUnitByDBAttribute(dbAttribute: DBAttribute): MeasureUnit{
    var measureUnit: MeasureUnit;
    var measureURI: URI_AS;
    var measureLabel: String;
    var aggregateFunction: AggregateFunction
    var unaryExpression: UnaryExpression;

    measureUnit = new MeasureUnit();

    measureURI = OntologyManager.createURI("ano");
    measureUnit.URI = measureURI;
    measureLabel = "MU.";
    measureLabel += dbAttribute.source;
    setAnalyticalElementLabel(measureUnit, measureLabel);

    aggregateFunction = getAggregateFunctionInstance("SUM");
    if(aggregateFunction)
        measureUnit.aggregateFunctions.put(aggregateFunction.URI, aggregateFunction);
    else if (OntologyManager.aggregateFunctions.length > 0){
        aggregateFunction = AggregateFunction(OntologyManager.aggregateFunctions[0]);
        measureUnit.aggregateFunctions.put(aggregateFunction.URI, aggregateFunction);
    }

    unaryExpression = new UnaryExpression();

```

```

        unaryExpression.URI = OntologyManager.createURI("");
        unaryExpression.informationSource = dbAttribute;
        measureUnit.unaryExpression = unaryExpression;

        return measureUnit;
    }

    /**
     * Retorna uma instância de MeasureUnit (medida), dada a URI da unidade de tema e a sua URI.
     * Obs.: A URI da unidade de tema é utilizada para acelerar a busca da medida.
     */
    public static function getMeasureUnit(themeUnitURI: URI_AS, measureUnitURI: URI_AS): MeasureUnit{
        var theme: Theme;
        var themeUnit: BThemeUnit;
        var measureUnits: IMap;
        var measureUnit: MeasureUnit;

        var t: uint;
        for(t = 0; t < _themes.length; t++){
            theme = _themes[t];
            var tu: uint;
            for(tu = 0; tu < theme.themeUnits.length; tu++){
                themeUnit = theme.themeUnits.values[tu];
                if (themeUnit.URI == themeUnitURI){
                    measureUnits = themeUnit.measureUnits;
                    var mu: uint;
                    for(mu = 0; mu < measureUnits.length; mu++){
                        measureUnit = measureUnits.values[mu];
                        if (measureUnit.URI == measureUnitURI)
                            return measureUnit;
                    }
                }
            }
        }
        return null;
    }

    /**
     * Remove uma instância de MeasureUnit (medida), dada a URI da unidade de tema e a sua URI.
     *
     */
    public static function removeMeasureUnit(themeUnitURI: URI_AS, measureUnitURI: URI_AS): Boolean{
        var theme: Theme;
        var themeUnit: BThemeUnit;
        var measureUnits: IMap;
        var measureUnit: MeasureUnit;

        var t: uint;
        for(t = 0; t < _themes.length; t++){
            theme = _themes[t];
            var tu: uint;
            for(tu = 0; tu < theme.themeUnits.length; tu++){
                themeUnit = theme.themeUnits.values[tu];
                if (themeUnit.URI == themeUnitURI){
                    measureUnits = themeUnit.measureUnits;
                    var mu: uint;
                    for(mu = 0; mu < measureUnits.length; mu++){
                        measureUnit = measureUnits.values[mu];
                        if (measureUnit.URI == measureUnitURI){
                            themeUnit.removeMeasure(measureUnitURI);
                            return true;
                        }
                    }
                }
            }
        }
        return false;
    }

    /**
     * Retorna uma instância de agrupamento, dado o seu rótulo.
     *
     */
    public static function createGrouping(label: String): Grouping{
        var element: Grouping = new Grouping();
        var elementURI: URI_AS = OntologyManager.createURI("ano");
        element.URI = elementURI;
        setAnalyticalElementLabel(element, label);
        return element;
    }

    /**
     * Retorna um array com todos os agrupamentos de uma unidade de tema.
     */
    public static function getGroupings(themeUnitURI: String): Array {
        var theme: Theme;
        var themeUnit: ThemeUnit;
        var groupings: IMap;
        var grouping: Grouping;

        for (var t: uint = 0; t < _themes.length; t++) {
            theme = _themes[t];

```

```

        for (var tu: uint = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            if (themeUnit.URI.prefixName == themeUnitURI)
                return themeUnit.groupings.values.toArray();
        }
    }

    return null;
}

/**
 * Retorna uma instância de Grouping (agrupamento), dada a URI da unidade de tema e a sua URI.
 * Obs.: A URI da unidade de tema é utilizada para acelerar a busca da medida.
 */
public static function getGrouping(themeUnitURI: URI_AS, groupingURI : URI_AS): Grouping{
    var theme: Theme;
    var themeUnit: ThemeUnit;
    var groupings: IMap;
    var grouping: Grouping;

    var t: uint;
    for (t = 0; t < _themes.length; t++) {
        theme = _themes[t];
        var tu: uint;
        for (tu = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            if (themeUnit.URI == themeUnitURI){
                groupings = themeUnit.groupings;
                var g: uint;
                for (g = 0; g < groupings.length; g++) {
                    grouping = groupings.values[g];
                    if (grouping.URI == groupingURI){
                        return grouping;
                    }
                }
            }
        }
    }

    return null;
}

/**
 * Remove uma instância de Grouping (agrupamento), dada a URI da unidade de tema e a sua URI.
 */
public static function removeGrouping(themeUnitURI: URI_AS, groupingURI : URI_AS): Boolean {
    var theme: Theme;
    var themeUnit: ThemeUnit;
    var groupings: IMap;
    var grouping: Grouping;

    var t: uint;
    for (t = 0; t < _themes.length; t++) {
        theme = _themes[t];
        var tu: uint;
        for (tu = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            if (themeUnit.URI == themeUnitURI){
                groupings = themeUnit.groupings;
                var g: uint;
                for (g = 0; g < groupings.length; g++) {
                    grouping = groupings.values[g];
                    if (grouping.URI == groupingURI){
                        themeUnit.removeGrouping(groupingURI);
                        return true;
                    }
                }
            }
        }
    }

    return false;
}

/**
 * Cria uma instância de GroupingUnit (unidade de agrupamento), dado um DBAttribute (atributo de um BD).
 */
public static function createGroupingUnitByDBAttribute(dbAttribute: DBAttribute): GroupingUnit{
    var groupingUnit: GroupingUnit;
    var groupingUnitURI: URI_AS;
    var groupingUnitLabel: String;
    var filterDetail: Detail;
    var groupingDetail: Detail;
    var filterDetailLabel: String;
    var groupingDetailLabel: String;

    groupingUnit = new GroupingUnit();
    groupingUnitURI = OntologyManager.createURI("ano");
    groupingUnit.URI = groupingUnitURI;
    groupingUnitLabel = "GU.";
    groupingUnitLabel += dbAttribute.source;
    setAnalyticalElementLabel(groupingUnit, groupingUnitLabel);
}

```

```

        filterDetailLabel = "DT.";
        filterDetailLabel += dbAttribute.source;
        filterDetail = createDetail(filterDetailLabel);
        filterDetail.attribute = dbAttribute;
        groupingUnit.filterDetail = filterDetail;

        groupingDetailLabel = "DT.";
        groupingDetailLabel += dbAttribute.source;
        groupingDetail = createDetail(groupingDetailLabel);
        groupingDetail.attribute = dbAttribute;
        groupingUnit.groupDetail = groupingDetail;

        return groupingUnit;
    }

/**
 * Retorna uma instância de GroupingUnit (unidade de agrupamento), dado a URI do agrupamento e a sua URI.
 * Obs.: a URI do agrupamento é utilizada apenas para acelerar o processo de busca.
 */
public static function getGroupingUnit(groupingURI: URI_AS, groupingUnitURI : URI_AS): GroupingUnit {
    var theme: Theme;
    var themeUnit: ThemeUnit;
    var groupings: IMap;
    var grouping: Grouping;
    var groupingUnits: IMap;
    var groupingUnit: GroupingUnit;

    var t: uint;
    for (t = 0; t < _themes.length; t++) {
        theme = _themes[t];
        var tu: uint;
        for (tu = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            groupings = theme.themeUnits.values[tu].groupings;
            var g: uint;
            for (g = 0; g < groupings.length; g++) {
                grouping = groupings.values[g];
                if (grouping.URI == groupingURI) {
                    groupingUnits = grouping.groupingUnits;
                    var gu: uint;
                    for (gu = 0; gu < groupingUnits.length; gu++) {
                        groupingUnit = groupingUnits.values[gu];
                        if (groupingUnit.URI == groupingUnitURI)
                            return groupingUnit;
                    }
                }
            }
        }
    }
    return null;
}

/**
 * Remove uma instância de GroupingUnit (unidade de agrupamento), dado a URI do agrupamento e a sua URI.
 */
public static function removeGroupingUnit(groupingURI: URI_AS, groupingUnitURI : URI_AS): Boolean {
    var theme: Theme;
    var themeUnit: ThemeUnit;
    var groupings: IMap;
    var grouping: Grouping;
    var groupingUnits: IMap;
    var groupingUnit: GroupingUnit;

    var t: uint;
    for (t = 0; t < _themes.length; t++) {
        theme = _themes[t];
        var tu: uint;
        for (tu = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            groupings = themeUnit.groupings;
            var g: uint;
            for (g = 0; g < groupings.length; g++) {
                grouping = groupings.values[g];
                if (grouping.URI == groupingURI) {
                    groupingUnits = grouping.groupingUnits;
                    var gu: uint;
                    for (gu = 0; gu < groupingUnits.length; gu++) {
                        groupingUnit = groupingUnits.values[gu];
                        if (groupingUnit.URI == groupingUnitURI) {
                            grouping.removeGroupingUnit(groupingUnitURI);
                            return true;
                        }
                    }
                }
            }
        }
    }
    return false;
}

```

```

/**
 * Cria uma instância de Detail (detalhamento), dado o seu rótulo.
 */
public static function createDetail(label: String): Detail{
    var element: Detail = new Detail();
    var elementURI: URI_AS = OntologyManager.createURI("ano");
    element.URI = elementURI;
    setAnalyticalElementLabel(element, label);
    return element;
}

/**
 * Retorna uma instância de Detail (detalhamento), dado a URI da unidade de tema e a sua URI.
 * Obs.: a URI da unidade de tema é utilizada apenas para acelerar o processo de busca.
 */
public static function getThemeUnitDetail(themeUnitURI: URI_AS, detailURI : URI_AS): Detail {
    var theme: Theme;
    var themeUnit: ThemeUnit;
    var details: IMap;
    var detail: Detail;

    var t: uint;
    for (t = 0; t < _themes.length; t++) {
        theme = _themes[t];
        var tu: uint;
        for (tu = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            if (themeUnit.URI == themeUnitURI){
                details = themeUnit.details;
                var g: uint;
                for (g = 0; g < details.length; g++) {
                    detail = details.values[g];
                    if (detail.URI == detailURI)
                        return detail;
                }
            }
        }
    }
    return null;
}

/**
 * Remove uma instância de Detail (detalhamento), dado a URI da unidade de tema e a sua URI.
 */
public static function removeThemeUnitDetail(themeUnitURI: URI_AS, detailURI: URI_AS): Boolean {
    var theme: Theme;
    var themeUnit: ThemeUnit;
    var details: IMap;
    var detail: Detail;

    var t: uint;
    for (t = 0; t < _themes.length; t++) {
        theme = _themes[t];
        var tu: uint;
        for (tu = 0; tu < theme.themeUnits.length; tu++) {
            themeUnit = theme.themeUnits.values[tu];
            if (themeUnit.URI == themeUnitURI){
                details = themeUnit.details;
                var g: uint;
                for (g = 0; g < details.length; g++) {
                    detail = details.values[g];
                    if (detail.URI == detailURI){
                        themeUnit.removeDetail(detailURI);
                        return true;
                    }
                }
            }
        }
    }
    return false;
}

////////////////////////////////////
//Ontologia ISO////////////////////////////////////
////////////////////////////////////

/**
 * Cria uma instância de Merge (junção).
 */
public static function createMerge(): Merge{
    var element: Merge = new Merge();
    element.URI = OntologyManager.createURI("map");
    return element;
}

/**
 * Cria uma instância de Expressão Unária.
 */
public static function createUnaryExpression(): UnaryExpression{
    var element: UnaryExpression = new UnaryExpression();
    element.URI = OntologyManager.createURI("map");
    element.isBracketed = false;
}

```

```

        return element;
    }

    /**
     * Cria uma instância de Expressão Binária.
     */
    public static function createBinaryExpression(type: BinaryExpression$Type): BinaryExpression{
        var element: BinaryExpression = new BinaryExpression();
        element.URI = OntologyManager.createURI("map");
        element.expressionOperator = type;
        element.isBracketed = false;
        return element;
    }

    /**
     * Retorna a instância de uma Merge, dada a sua URI.
     */
    public static function getMerge(mergeURI: URI_AS): Merge{
        var merge: Merge;
        var j: uint;
        for (j=0; j< merges.length; j++){
            merge = Merge(merges[j]);
            if(merge.URI == mergeURI)
                return merges[j];
        }
        return null;
    }

    /**
     * Remove a instância de uma Merge, dada a sua URI.
     */
    public static function removeMerge(mergeURI: URI_AS): Boolean{
        var merge: Merge;
        var j: uint;
        for (j=0; j< merges.length; j++){
            merge = Merge(merges[j]);
            if(merge.URI == mergeURI)
                merges.splice(j,1);
        }
        return false;
    }

    /**
     * Retorna a instância de uma Expressão Binária, dada a sua URI e a URI da Merge da qual ela faz parte.
     */
    public static function getBinaryExpression(binaryExpressionURI: URI_AS, mergeURI: URI_AS): BinaryExpression{
        return getBinaryExpressionAux(getMerge(mergeURI).relation, binaryExpressionURI);
    }

    private static function getBinaryExpressionAux(binaryExpression: BinaryExpression, binaryExpressionURI: URI_AS): BinaryExpression{
        var binaryExpressionReturn: BinaryExpression = null;

        if(binaryExpression.URI == binaryExpressionURI){
            return binaryExpression;
        }

        if(binaryExpression.firstExpression is BinaryExpression) {
            binaryExpressionReturn = getBinaryExpressionAux(BinaryExpression(binaryExpression.firstExpression),
binaryExpressionURI);
            if (binaryExpressionReturn != null)
                return binaryExpressionReturn;
        }

        if(binaryExpression.secondExpression is BinaryExpression) {
            binaryExpressionReturn = getBinaryExpressionAux(BinaryExpression(binaryExpression.secondExpression),
binaryExpressionURI);
            if (binaryExpressionReturn != null)
                return binaryExpressionReturn;
        }

        return binaryExpressionReturn;
    }

    //////////////////////////////////////
    //GETS E SETS////////////////////////////////////
    //////////////////////////////////////

    public static function get dbCollectionSchema(): DBCollection {
        return _dbCollectionSchema;
    }

    public static function set dbCollectionSchema(dbCollectionSchema: DBCollection): void {
        _dbCollectionSchema = dbCollectionSchema;
    }

    public static function get themes(): Array {
        return _themes;
    }

    public static function set themes(themes: Array): void {
        _themes = themes;
    }

```

```

    }

    public static function get merges(): Array {
        return _merges;
    }

    public static function set merges(merges: Array): void {
        _merges = merges;
    }

    public static function get deletedURIs(): Array {
        return _deletedURIs;
    }

    public static function get locales(): Array {
        return _locales;
    }

    public static function set locales(locales: Array): void {
        _locales = locales;
    }

    public static function get aggregateFunctions(): Array {
        return _aggregateFunctions;
    }

    public static function set aggregateFunctions(aggregateFunctions: Array): void {
        _aggregateFunctions = aggregateFunctions;
    }
}
}
}

```

Mapper_BusinessService (servidor – Java)

MapperService.java

```

package br.ufsc.inf.feco.mapper.service;

import info.stela.utils.ConnectionMgr;

import java.sql.SQLException;
import java.util.Arrays;
import java.util.Iterator;
import java.util.Map;
import java.util.StringTokenizer;

import org.apache.commons.lang.StringUtils;

import br.org.stela.semantic.SemanticMetadata;
import br.org.stela.semantic.collection.SemanticCollection;
import br.org.stela.semantic.collection.SemanticList;
import br.org.stela.semantic.exception.SemanticException;
import br.org.stela.semantic.impl.ontology.JenaSemanticResourceFactory;
import br.org.stela.semantic.knowledgebase.KnowledgeBase;
import br.org.stela.semantic.knowledgebase.exception.KnowledgeBaseException;
import br.org.stela.semantic.manager.ApplicationInstanceCreator;
import br.org.stela.semantic.manager.OntologyManager;
import br.org.stela.semantic.manager.OntologyManagerFactory;
import br.org.stela.semantic.ontology.SemanticInstance;
import br.org.stela.semantic.ontology.SemanticResource;
import br.org.stela.semantic.ontology.SemanticResourceFactory;
import br.org.stela.semantic.ontology.SemanticStatement;
import br.org.stela.semantic.ontology.URI;
import br.org.stela.semantic.ontology.application.AggregateFunction;
import br.org.stela.semantic.ontology.application.ClassURI;
import br.org.stela.semantic.ontology.application.Element;
import br.org.stela.semantic.ontology.application.Element2Instance;
import br.org.stela.semantic.ontology.application.Locale;
import br.org.stela.semantic.ontology.application.PropertyURI;
import br.org.stela.semantic.ontology.application.analytical.BIThemeUnit;
import br.org.stela.semantic.ontology.application.analytical.Grouping;
import br.org.stela.semantic.ontology.application.analytical.MeasureUnit;
import br.org.stela.semantic.ontology.application.analytical.Theme;
import br.org.stela.semantic.ontology.application.analytical.ThemeUnit;
import br.org.stela.semantic.ontology.application.source.Collection;
import br.org.stela.semantic.ontology.application.source.DBCollection;
import br.org.stela.semantic.ontology.application.source.InformationSource;
import br.org.stela.semantic.ontology.application.source.Merge;
import br.org.stela.semantic.repository.DBSourceRepository;

public class MapperService {

    private static OntologyManager om = null;

    private static DBSourceRepository repository = null;

```

```

private static KnowledgeBase kb = null;

private static SemanticResourceFactory factory = null;

private static String dbSchema = "cvlattesdw"; //por enquanto está fixo

private static final String IGNORED_CHARACTER = "_";
private static final String PREFIX = "ekp";
private static final String NAMESPACE = "http://www.stela.org.br/ekp#";

static {
    try {
        String appConfigAlias = "CVLATTESDW"; //alias no arquivo de configuração de acesso ao banco
        String ontologyModule = "";

        //faz conexão no repositório e retorna um DBSourceRepository.
        //DBSourceRepository representa a ontologia ISO
        repository = createSourceRepository(appConfigAlias);

        //KnowledgeBase representa o arquivo OWL (utilizada pelo Jena)
        kb = createKnowledgeBase(ontologyModule);
        om = new OntologyManager(kb);
        factory = new JenaSemanticResourceFactory();

    } catch (SemanticException e) {
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

//faz conexão no repositório e retorna um DBSourceRepository
public static DBSourceRepository createSourceRepository(String alias) throws Exception{
    DBSourceRepository repository = null;
    if (StringUtils.isBlank(alias) == false)
        //DBSourceRepository representa a ontologia ISO
        repository = new DBSourceRepository(ConnectionMgr.getConnection(alias));
    else{
        repository = new DBSourceRepository(ConnectionMgr.getConnection());
    }
    return repository;
}

public static KnowledgeBase createKnowledgeBase(String module) throws SemanticException{
    KnowledgeBase knowledgeBase = OntologyManagerFactory.getOntologyManager(module).getKnowledgeBase();
    return knowledgeBase;
}

public static DBCollection generateDBCollectionSchema(){
    populateSOntologyInstances();
    return getDBCollectionSchema();
}

public static void populateSOntologyInstances(){
    try {
        //Popula ontologia ISO
        ApplicationInstanceCreator creator = new ApplicationInstanceCreator(factory, repository, kb);
        creator.createInstances(dbSchema, null);
        om.initializeMapping();
    } catch (SemanticException e) {
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public static DBCollection getDBCollectionSchema(){
    Map<String, Element> mapURI_Element = om.getApplicationOntology().getOntology().get(ClassURI.DB_COLLECTION);
    DBCollection db = (DBCollection) mapURI_Element.values().toArray()[0];
    return (DBCollection)db.getParent();
}

public static Theme[] generateThemes(String tableFactPrefix, String measurePrefix) {
    populateBIOntologyInstances(tableFactPrefix, measurePrefix);
    return getThemes();
}

public static Theme[] getThemes(){
    Theme[] arrTheme = null;
    try {
        Map<String, Element> mapURI_Element = om.getApplicationOntology().getOntology().get(ClassURI.THEME);
        arrTheme = new Theme[mapURI_Element.size()];
        mapURI_Element.values().toArray(arrTheme);
    } catch (Exception e) {
        e.printStackTrace();
        throw new RuntimeException(e);
    }
    return arrTheme;
}

```



```

}

public static Merge[] getMerges(){
    Merge[] merges = null;
    try {
        Map<String, Element> mapURI_Element = om.getApplicationOntology().getOntology().get(ClassURI.MERGE);
        merges = new Merge[mapURI_Element.size()];
        mapURI_Element.values().toArray(merges);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return merges;
}

public static Locale[] getLocales(){
    Locale[] locales = null;
    try {
        Map<String, Element> mapURI_Element = om.getApplicationOntology().getOntology().get(ClassURI.LOCALE);
        locales = new Locale[mapURI_Element.size()];
        mapURI_Element.values().toArray(locales);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return locales;
}

public static AggregateFunction[] getAggregateFunctions(){
    AggregateFunction[] aggregateFunctions = null;
    try {
        Map<String, Element> mapURI_Element = om.getApplicationOntology().getOntology().get(ClassURI.AGGREGATE_FUNCTION);
        aggregateFunctions = new AggregateFunction[mapURI_Element.size()];
        mapURI_Element.values().toArray(aggregateFunctions);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return aggregateFunctions;
}

public static void populateBIOntologyInstances(String tableFactPrefix, String measurePrefix){

    try {

        String sql = null; //SQL optional para recuperar as tabelas desejadas
        SemanticCollection<? extends Collection> tables = null;

        if (StringUtils.isBlank(sql))
            tables = repository.getCollections(null, tableFactPrefix + "%");
        else{
            tables = repository.getCollections(sql);
        }

        SemanticInstance theme = createTheme("Theme.Default", kb, factory);
        while ( tables.hasNext() ){
            Collection table = tables.next();
            SemanticInstance themeUnit = createThemeUnit(tableFactPrefix, measurePrefix, table, factory, theme, kb);
            if ( themeUnit != null )
                theme.addProperty(PropertyURI.HAS_THEME_UNIT.getURI(), themeUnit );
        }

        kb.addSemanticInstance(theme);

        om.initializeMapping();

    } catch (SemanticException e) {
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public static boolean saveThemes(Theme[] themes, URI[] deletedURIs){
    for(int i=0; i< themes.length; i++){
        Theme theme = themes[i];

        Element2Instance elementToInstance = new Element2Instance(factory);
        SemanticInstance instance = elementToInstance.toInstance(theme);
        try {
            kb.addSemanticInstance(instance, true);
            SemanticList<URI> sl = new SemanticList<URI>(Arrays.asList(deletedURIs));
            kb.removeSemanticInstances(sl);
            kb.save();
            om.initializeMapping();
        } catch (KnowledgeBaseException e){
            e.printStackTrace();
        } catch (SemanticException e) {
            e.printStackTrace();
        }
    }

    System.out.println("ThemesSaved!!!");
    return true;
}

```

```

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//Métodos pra criar as instâncias da ontologia analitica (BI)////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

private static SemanticInstance createTheme(String themeName, KnowledgeBase kb, SemanticResourceFactory factory) throws SemanticException {
    SemanticInstance theme = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE + themeName),
factory.createSemanticClass(ClassURI.THEME.getURI()));
    theme.addProperty(PropertyURI.HAS_LABEL.getURI(), createLabel(themeName, factory));
    return theme;
}

/**
 * @param tokenizer
 * @param builder
 */
private static String nameElement(StringTokenizer tokenizer, StringBuilder builder) {
    if (tokenizer.countTokens() == 0)
        builder.append( StringUtils.capitalize( StringUtils.lowerCase(tokenizer.nextToken()) ) );
    else{
        while (tokenizer.hasMoreTokens()){
            builder.append(StringUtils.capitalize( StringUtils.lowerCase(tokenizer.nextToken()) ));
        }
    }
    return builder.toString();
}

/**
 * @param table
 * @param factory
 * @param theme
 * @throws SemanticException
 */
private static SemanticInstance createThemeUnit(String tableFactPrefix, String measurePrefix, Collection table, SemanticResourceFactory factory, SemanticInstance
theme ,KnowledgeBase kb) throws SemanticException {
    SemanticInstance themeUnit = null;
    if ( table.getSource().startsWith(tableFactPrefix)){
        StringTokenizer tokenizer = new StringTokenizer(table.getSource(), MapperService.IGNORED_CHARACTER);
        String themeUnitName = nameElement(tokenizer, new StringBuilder("TU."));
        themeUnit = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE + themeUnitName),
factory.createSemanticClass(ClassURI.BI_THEME_UNIT.getURI()));
        themeUnit.addProperty(PropertyURI.HAS_LABEL.getURI(), createLabel(themeUnitName, factory));

        //Medidas
        Iterator<InformationSource> it = table.getChild().values().iterator();
        while ( it.hasNext() ){
            SemanticInstance measureUnit = createMeasureUnits(measurePrefix, factory, it.next(), kb);
            if ( measureUnit != null )
                themeUnit.addProperty(PropertyURI.HAS_MEASURE_UNIT.getURI(), measureUnit);
        }

        SemanticCollection<? extends Merge> merges = repository.getImportedMerges(table);
        //Agrupamentos
        while ( merges.hasNext() ){
            Merge merge = merges.next();

            SemanticInstance grouping = createGrouping( factory, merge, kb );
            SemanticInstance groupingTable = kb.getSemanticInstance(merge.getSourceCollection().getURI());
            SemanticCollection<? extends SemanticStatement> childStatements =
groupingTable.getSemanticStatements(PropertyURI.HAS_CHILD.getURI().getURI());
            while ( childStatements.hasNext() ){
                SemanticInstance groupingUnit = createGroupingUnit( factory, childStatements.next().getObjectValue() ,
kb);
                grouping.addProperty(PropertyURI.HAS_GROUPING_UNIT.getURI(), groupingUnit);
            }
            themeUnit.addProperty(PropertyURI.HAS_GROUPING.getURI(), grouping);
        }
    }
    return themeUnit;
}

private static SemanticInstance createGrouping( SemanticResourceFactory factory, Merge merge, KnowledgeBase kb) throws SemanticException {
    SemanticInstance grouping = null;
    StringTokenizer tokenizer = new StringTokenizer(merge.getSourceCollection().getSource(), MapperService.IGNORED_CHARACTER);
    String groupName = nameElement(tokenizer, new StringBuilder("Grp."));

    grouping = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE + groupName),
factory.createSemanticClass(ClassURI.GROUPING.getURI()));
    grouping.addProperty(PropertyURI.HAS_LABEL.getURI(), createLabel(groupName, factory));
    return grouping;
}

private static SemanticInstance createGroupingUnit(SemanticResourceFactory factory, SemanticResource attribute, KnowledgeBase kb) throws SemanticException {
StringBuilder("GU.");

    SemanticInstance groupingUnit = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE + groupingUnitName),
factory.createSemanticClass(ClassURI.GROUPING_UNIT.getURI()));
    SemanticInstance detail = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE + "Dt." + groupingUnitName),
factory.createSemanticClass(ClassURI.DETAIL.getURI()));
}

```

```

        detail.addProperty(PropertyURI.HAS_ATTRIBUTE.getURI(), attribute );
        detail.addProperty(PropertyURI.HAS_LABEL.getURI(), createLabel(groupingUnitName, factory));
        groupingUnit.addProperty(PropertyURI.HAS_GROUP_DETAIL.getURI(), detail);
        groupingUnit.addProperty(PropertyURI.HAS_FILTER_DETAIL.getURI(), detail);
        groupingUnit.addProperty(PropertyURI.HAS_LABEL.getURI(), createLabel(groupingUnitName, factory));
        return groupingUnit;
    }

    /**
     * @param factory
     * @param themeUnit
     * @param it
     * @throws SemanticException
     */
    private static SemanticInstance createMeasureUnits(String measurePrefix, SemanticResourceFactory factory, InformationSource attribute, KnowledgeBase kb)
    throws SemanticException {
        SemanticInstance measureUnit = null;
        if ( attribute.getSource().startsWith(measurePrefix) ) {
            StringTokenizer tokenizer = new StringTokenizer(attribute.getSource(), MapperService.IGNORED_CHARACTER);
            String measureName = nameElement(tokenizer, new StringBuilder("MU.")).getURI();

            measureUnit = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE + measureName),
            factory.createSemanticClass(ClassURI.MEASURE_UNIT.getURI()));
            measureUnit.addProperty(PropertyURI.HAS_LABEL.getURI(), createLabel(measureName, factory));

            measureUnit.addProperty(PropertyURI.HAS_AGGREGATE_FUNCTION.getURI(), kb.getSemanticInstance( new
            URI(SemanticMetadata.EKP_ONTOLOGY_URI + "#SUM") ));
            measureUnit.addProperty(PropertyURI.HAS_AGGREGATE_FUNCTION.getURI(), kb.getSemanticInstance( new
            URI(SemanticMetadata.EKP_ONTOLOGY_URI + "#COUNTING") ));

            SemanticInstance unaryExpression = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE +
            "UE." + measureName), factory.createSemanticClass(ClassURI.UNARY_EXPRESSION.getURI()));

            unaryExpression.addProperty(PropertyURI.HAS_INFORMATION_SOURCE.getURI(), kb.getSemanticInstance(attribute.getURI()) );
            unaryExpression.addProperty(PropertyURI.IS_BRACKETED.getURI(), Boolean.FALSE );

            measureUnit.addProperty(PropertyURI.HAS_UNARY_EXPRESSION.getURI(), unaryExpression );
        }
        return measureUnit;
    }

    /**
     * @param name
     * @param factory
     * @return
     * @throws SemanticException
     */
    private static SemanticInstance createLabel(String name, SemanticResourceFactory factory) throws SemanticException {
        SemanticInstance label = factory.createSemanticInstance(new URI(MapperService.PREFIX, MapperService.NAMESPACE + "Lb." + name),
        factory.createSemanticClass(ClassURI.LABEL.getURI()));
        label.addProperty(PropertyURI.LABEL.getURI(), name);
        label.addProperty(PropertyURI.HINT.getURI(), "Hint " + name);
        return label;
    }
}

```